UNIVERSITÉ DE NICE - SOPHIA ANTIPOLIS UFR Sciences

École Doctorale STIC

Thèse

pour obtenir le titre de

Docteur en Sciences

de l'Université de Nice - Sophia Antipolis

Spécialité : INFORMATIQUE

présentée et soutenue par

Céline HUDELOT

Équipe d'accueil : ORION – INRIA Sophia-Antipolis

Towards a Cognitive Vision Platform for Semantic Image Interpretation; Application to the Recognition of Biological Organisms

Thèse dirigée par Monique THONNAT

Soutenue publiquement le vendredi 29 avril 2005
devant le jury composé de :

Mme	Annie	CAVARERO	I3S, Université de Nice	Présidente
Mme	Isabelle	BLOCH	ENST/ CNRS	Rapporteur
M.	Bernd	NEUMANN	Université de Hamburg, Allemagne	Rapporteur
M.	Regis	CLOUARD	Universite de Caen, France	Examinateur
M.	Paul	BOISSARD	INRA Sophia Antipolis	Examinateur
Mme	Monique	THONNAT	INRIA Sophia Antipolis	Directrice

Acknowledgments

First, I would like to express my sincere thanks to Monique Thonnat, supervisor of my thesis, for her advice, her support and her encouragement during the years of my research. She has directed my works while letting me a lot of autonomy. I have really appreciated all our discussions, the scientific but also the no scientific ones.

I sincerely thank Professor Bernd Neumann of the Universität of Hamburg (Germany) and Professor Isabelle Bloch for having accepted to be the two reviewers of my thesis and for their good comments and advice on my dissertation.

The following of this section will be in French.

Je remercie Régis Clouard et Annie Cavarero de m'avoir fait l'honneur d'être membres du jury de thèse.

Je remercie tout particulièrement Paul Boissard non seulement pour avoir accepté de prendre part au jury, pour son suivi et ses précieuses remarques tout le long de ma thèse mais aussi pour cette collaboration entre l'INRA et l'INRIA dont il est l'initiateur et sans laquelle mes travaux de thèse n'auraient pu se faire.

Cette thèse s'est déroulée grâce à la coopération entre trois instituts. Je remercie les responsables du SCRADH pour avoir cru en nos travaux.

Je remercie également l'ensemble du personnel de l'URIH de Sophia Antipolis et plus particulièrement Guy et Philippe avec qui la traque des champignons et autres petites bêtes a toujours été une partie de plaisir.

Un grand merci également à tous les membres de l'équipe Orion de l'INRIA; les anciens: Alain, mon maître nageur et bien plus, Nico D le Bédéphile, Irène, Tinh et Binh; ceux qui ont partagés avec moi ces trois années et qui sont devenus bien plus que des collègues de travail (même si certains m'ont détesté sur leur vélo un certain été de canicule en Bourgogne): Alberto qui restera mon éternel cobureau, Magali, Ben, Fred, Christophe. Je remercie aussi Sabine et Nicolas M avec qui travailler a été très enrichissant, François l'adepte des débats, et puis bien sûr Cathy pour tous les services rendus. Merci aussi à Julien, Florent, Vincent, Bernard, Annie, Valery, Gabriele, Florence, Marcos et Alexander, les derniers arrivés.

J'en profite également pour remercier le SEMIR pour son efficacité et la qualité de son service.

Ces années de thèse auraient été beaucoup plus difficiles à supporter sans la présence des amis qui m'ont accompagné de près ou de loin et m'ont permis de surmonter les moments difficiles et les moments de doute: la bande d'Orion bien sûr mais aussi Olivier, Nico S le plus attendrissant des grincheux, Damiana, Fabien G, Sylvain, Guillaume et Ludo avec qui j'ai découvert cette merveilleuse région qu'est la Côte d'Azur, Alex, Thomas et Séverine, Guillaume, et puis tous ceux que j'oublie. Un grand merci aussi à ceux qui sont restés mes amis malgré un éloignement géographique et mes longues périodes de silence: Pti Guy, Gael, Charlotte, Fred, Sophie et tous les autres.

Il y a une absente dans cette longue énumération que je tiens à remercier tout particulièrement. Un énorme merci à Marie Claude pour son amitié à toute épreuve et pour tous ces moments partagés. Les dîners, dans la cafeteria non éclairée de l'INRIA pendant la période de rédaction, resteront les souvenirs les plus marquants de cette fin de thèse.

Et puis bien sûr, mes pensées les plus tendres vont à Baptiste pour sa présence et son soutien pendant cette dernière année de thèse. Merci de m'avoir permis de "tenir debout" et d'arriver au bout...

Enfin, je tiens à exprimer mes plus vifs et plus sincères remerciements à l'ensemble de ma famille : mes parents, Marie Paule et Pascal, qui ont toujours respecté mes choix, m'ont accompagné et m'ont donné les moyens financiers mais surtout humains de les accomplir; mes soeurs et plus chères amies, Hélène et Émilie, mon frère, le discret Joseph mais aussi Laurent, Adil, la famille Penault pour leur accueil à Strasbourg, les musiciens de la Roquette et tous les autres...

Pour terminer, j'aimerais reprendre une phrase de ma mère parlant de cette année scolaire comme d'une année importante car deux de ses filles accoucheraient: la première d'une thèse portée pendant plus de 3 ans et la deuxième d'un enfant. Je trouve que cette métaphore est ce qu'il y a de plus parlant. Je dédie cette thèse au petit Louis, mon neveu.

Abstract

Abstract

This work deals with cognitive vision and in particular semantic image interpretation. One of the main challenges of cognitive vision is to develop a flexible, adaptable system, capable of performing complex image analysis tasks and of extracting information from various scenes and images. The objective of this thesis is to cope with this challenging problem by the design of a reusable and generic cognitive vision platform for the complex problem of semantic image interpretation. We are interested in both the cognitive issues and the software engineering ones involved in the design of such a platform. The proposed cognitive vision platform is a unified environment which proposes generic and reusable tools for the design of complete semantic image interpretation systems.

The semantic image interpretation problem is complex and can be divided into three more-tractable sub-problems: (1) the semantic interpretation, (2) the problem of the mapping between high level representations of physical objects and the sensor data extracted from images, (3) the image processing problem. To manage and separate the different sources of knowledge and reasoning, we propose a distributed architecture based on the cooperation of three Knowledge Based Systems (KBS). Each KBS is highly specialized for the corresponding sub-problem of semantic image interpretation. For each sub-problem, we define a dedicated engine and a unified knowledge representation model.

The implementation of the cognitive vision platform has been made with the LAMA platform, a software platform for the development of knowledge based systems, designed in the ORION team.

To validate our cognitive vision platform, we have chosen a real world application: the early diagnosis of plant diseases. In particular, we have studied the rose leaf diseases in greenhouses. This work has been made in cooperation with INRA (French National Institute for Research in Agronomy).

Keywords

Semantic Image Interpretation, Cognitive Vision, Artificial Intelligence, Software Engineering, Knowledge Based Systems, Early Plant Disease Diagnosis, Integrated Pest Management

Résumé

Ces travaux de thèse ont pour but de faire des avancées dans le domaine de la vision cognitive en proposant une plate forme fonctionnelle et logicielle pour le problème complexe de l'interprétation sémantique d'images. Nous nous sommes focalisés sur la proposition de solutions génériques et indépendantes de toute application. Plus qu'une solution à un problème spécifique, la plate forme proposée est une architecture minimale qui fournit des outils réutilisables pour la conception de systèmes d'interprétation sémantique d'images.

Le problème de l'interprétation sémantique d'images est un problème complexe qui peut se séparer en 3 sous problèmes plus faciles à résoudre en tant que problèmes indépendants: (1) l'interprétation sémantique, (2) la gestion des données visuelles pour la mise en correspondance des représentations abstraites haut niveau de la scène avec les données image issues des capteurs et (3) le traitement d'images. Nous proposons une architecture distribuée qui se base sur la coopération de trois systèmes à base de connaissances (SBCs). Chaque SBC est spécialisé pour un des sous problèmes de l'interprétation d'images. Pour chaque SBC nous avons proposé un modèle générique en formalisant la connaissance et des stratégies de raisonnement dédiées. De plus, nous proposons d'utiliser deux ontologies pour faciliter l'acquisition de la connaissance et permettre l'interopérabilité entre les trois différents SBCs.

Un travail d'implémentation de la plate forme de vision cognitive a été fait à l'aide de la plate forme de développement de systèmes à base de connaissances LAMA conçue par l'équipe ORION.

Les solutions proposées ont été validées sur une application concrète et difficile: le diagnostic précoce des pathologies végétales et en particulier des pathologies du rosier de serre. Ce travail a été effectué en coopération avec l'INRA (Institut National de la Recherche Agronomique).

Mots-clefs

Interprétation sémantique d'images, Vision cognitive, Intelligence artificielle, Génie logiciel, Systèmes à base de connaissances, Ontologie, Diagnostic précoce des pathologies végétales, Protection integrée des cultures.

Contents

List of Figures

Chapter 1

Introduction

This thesis deals with the problem of semantic image interpretation. Semantic image interpretation is a problem of visual perception, i.e. the perception of the real world by visual sensors (human visual system, digital camera,...). We tackle this problem from the point of view of the **building of automatic image interpretation systems**.

1.1 Problem Overview

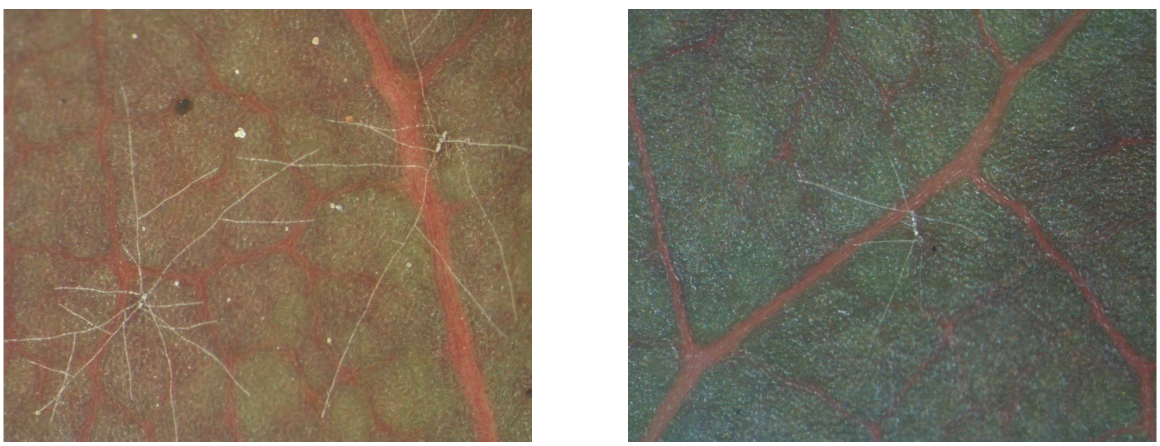


Figure 1.1: Two examples of images to interpret

The problem of semantic image interpretation can be simply illustrated with the examples of the figure 1.1. What does it mean to perform semantic image interpretation? When we look at the images of the figure 1.1, we have to answer to the following question: what are the semantic contents of these images? According to the knowledge of the interpreter, different answers and different interpretations are possible. For the image at left, possible interpretations could be:

1. 2 white thin line networks on a textured complex background,
2. an abnormality on a vegetable leaf,
3. a microscopic fungi on a vegetable leaf,
4. two very early infections of powdery mildew on a young rose leaf.

The image on the right is quite similar to the left image and could have approximately the same interpretations than the previous one. Nevertheless, to illustrate a step further the multiplicity of possible interpretations, other possible interpretations for the right image could be:

1. a thick red line network on a green background;
2. a network of roads in an aerial image;
3. a network of roads in a forest area.

Without any other information, we could consider that all these interpretations are correct. The analysis of these images shows that a unique solution does not exist for the problem of semantic image interpretation. Nevertheless, the different interpretations enable to identify fundamental issues. The identified issues and their repercussions on the design of automatic image interpretation systems are:

- **Importance of the a priori knowledge.**
  The answer to the interpretation problem highly depends on the a priori knowledge level of the interpreter. Image semantics is not inside the image. Indeed, without any knowledge of plant disease symptoms, it is impossible to interpret thin white line networks (figure 1.1) as an early infection of powdery mildew. Therefore, semantic image interpretation is an **intensive knowledge based process**. From the point of view of the design of semantic image interpretation systems, it implies to make available to the system such kind of more or less sophisticated knowledge.

- **Importance of the contextual information.**
  Without knowing that images of figure 1.1 are microscopic biological images, the aerial interpretations are completely valid. By contextual information, we refer to all the non visual, additional information, that may influence the way a scene is perceived. This contextual information can be of various types. For instance, the term *biological* refers to the application domain context and the term *microscopic* refers to the image acquisition context. Thus, the role of the context is significant for visual interpretation. For semantic image interpretation systems, representing and using context information in an appropriate way, can improve system efficiency and system performance.

- **Importance of the high level goal of the interpretation.**
  Semantic image interpretation is a goal-oriented task. Interpretation results depend on the observer goal. Taking into account this goal enables to focus the analysis only on what is relevant for the goal. For instance, in the previous images (fig 1.1), if the biological goal is to confirm or invalidate the presence of fungi on rose leaves, it is not necessary to search for other rose diseases on images. The detection and the recognition of all objects of the scene are not necessarily useful to build interpretation results. The interpretation strategy of a semantic image interpretation system has to be goal-oriented.

- **Importance of the notion of scene.**
  By the notion of scene, we mean existence of multiple objects and of spatial relations between objects. The relations between objects and their environment are important. We extensively use spatial relations between objects and their environment to detect and recognize them. As a consequence, semantic image interpretation is

"above the level of single object recognition" [Neumann and Weiss, 2003]. Therefore, knowledge of the scene and spatial reasoning mechanisms are necessary parts of an image interpretation system.

- **Importance of an intelligent extraction of objects from images.**
  The two images of the figure 1.1 show us that for a same class of objects (the powdery mildew), there are various appearances and various image backgrounds. The use of a specialized image processing program is not sufficient. The extraction of objects from images (**i.e. the image processing problem**) has to be managed in an intelligent way. The image processing process has to be flexible and is has to be able to **adapt itself** to changing conditions of the environment.

Building image interpretation systems consists in endowing computers with a visual system enabling them to understand their environment. During the last five decades, there has been intensive and maturing research concerning automatic image interpretation systems. This intensive research has given birth to several philosophies for the design of image interpretation systems. Many image interpretation systems with different motivations and different models was built. Nevertheless, a current statement is that most of them are brittle technologies. They are often restricted to the needs of a particular application. After three decades, there is still no generic automatic image interpretation system able to deal with different tasks and images, like the human vision apparatus.

The research field of **cognitive vision** emerges from this statement. Cognitive vision criticizes the lack of robustness, the lack of adaptability and the application dependence of traditional image interpretation systems. According to the ECVision[1] road map [Vernon, 2004], **cognitive vision** refers to *the attempt to achieve more robust, resilient and adaptable computer vision systems by endowing them with cognitive faculties: the ability to learn, adapt, weight alternative solutions, and even the ability to develop new strategies for analysis and interpretation.* Therefore, one of the main challenges in semantic image interpretation is to develop a flexible, adaptable system, capable of performing complex image analysis tasks, of extracting information from varying scenes or images and of performing high level semantic interpretation.

1.2 Our Objective

Our objective is to make advances in the emerging field of cognitive vision by the design of a **reusable and generic cognitive vision platform** for the complex problem of semantic image interpretation. Our aim is not to design yet another application dependent image interpretation system but to propose generic and reusable tools for the design of such systems.

The proposed cognitive vision platform is a unified environment for the design of complete image interpretation systems. Cognitive vision refers to the fact that the resulting image interpretation systems must exhibit an intelligent and robust behavior for the resolution of the global semantic image interpretation problem. We are interested in both **cognitive issues** and **software engineering** ones involved in the design of the platform. In particular, we have focused our attention on the property of **re-usability** and **convenience**.

The cognitive vision platform is defined as :

[1]The European research Network for Cognitive Computer Vision Systems, www.ecvision.org

- A minimal functional architecture for semantic image interpretation systems. This architecture defines what are the task oriented functional modules which are necessary parts of a semantic image interpretation system, what are their role and their interactions.
- The formalization and the explicitation of the different types of knowledge and of the different types of reasoning involved in the global problem of semantic image interpretation.
- A generic and convenient environment of development for the design of image interpretation systems for various applications.

The objective of this work is twofold:

- First, the definition and the design of a cognitive vision platform for semantic image interpretation.
- Second, the validation of the proposed platform by a real world application: the recognition of biological organisms in their natural environment.

1.3 Context of the Study

This work takes place in the **Orion team** at INRIA Sophia Antipolis. Orion is a multidisciplinary team at the frontier of computer vision, knowledge based systems, and software engineering. Therefore, our work benefits from a great experience in the domain of image understanding and complex object recognition and in the domain of software engineering for the reuse of intelligent systems.

In particular, several works of the Orion team have proved the efficiency of the use of explicit expertise to solve complex image analysis problems. The tackled problems were the automation of the use of an image processing library by program supervision [Moisan and Thonnat, 1995], [Moisan and Thonnat, 2000] and the automation of the object recognition task [Thonnat, 2002]. In [Ossola, 1996], a knowledge based approach for the design of automatic object recognition systems was proposed. This approach is based on the knowledge explicitation and on the reasoning formalization.

Moreover, the reuse of intelligent systems is another active research domain of the Orion team. Problem solving methods have been used to design engines, independent of any specific applications, but dedicated to a particular task. To implement the proposed cognitive vision platform, the LAMA software platform was used [Moisan, 1998]. The LAMA platform provides a unified environment to design not only knowledge bases but also inference engines.

1.4 The Proposed Approach

The complex problem of semantic image interpretation can be divided into three more tractable sub-problems:

- **The image processing problem**, for the extraction and the numerical description of objects of interest from images.
- **The problem of the mapping** between the *qualitative* high level representations of the scene and the *numerical* information extracted from images.

- **The semantic interpretation problem**, i.e. the understanding of the scene using domain concepts, using the domain terminology.

Each sub-problem is a problem as such, involving its proper expertise. To manage and to separate the different sources of knowledge and the different reasoning strategies, we propose a minimal distributed architecture based on the **cooperation of three knowledge based systems**. Each KBS (Knowledge Based System) is highly specialized for the corresponding sub-problem of semantic image interpretation. The architecture is composed of an **image processing program supervision KBS**, a **visual data management KBS** specialized in symbol grounding and spatial reasoning and a **semantic interpretation KBS**. This architecture is **problem-oriented**: the global semantic image interpretation problem is broken down into sub-problems, and each sub-problem is assigned to a particular part of the system.

We are interested in providing a unified environment for the design of semantic image interpretation systems, i.e. a set of reusable tools. We study the three different sub-problems under the software and knowledge engineering points of view. For each sub-problem, we propose a model for the specific problem-solving mechanism and a set of generic concepts to model the knowledge involved in the sub-problem. Then, we build a cognitive vision platform by the integration of the different sub-problem models and by providing means for their interaction and their interoperability. Finally, we choose a complex semantic image interpretation problem, i.e. the early diagnosis of rose leaves, to test and validate the cognitive vision platform with a real world application.

1.5 Dissertation Structure

This dissertation contains the following parts.

The **chapter 2** is a state of the art on image interpretation systems. Due to the great number of works dealing with image interpretation systems, this state of the art is not exhaustive. We focus on the works that we have considered as relevant for our work. We first present some philosophical issues for the building of image interpretation systems. Then, we detail several automatic image interpretation systems with the analysis of their advantages and drawbacks. We also introduce the emerging field of cognitive vision in which our work takes part.

In the **chapter 3**, we present our global approach for the semantic image interpretation problem: a generic and reusable cognitive vision platform for the design of semantic image interpretation systems. In this chapter, we define what we want, what the requirements of the cognitive vision platform are and what are they for. Then, we give a global overview of the proposed cognitive vision platform.

In the **chapter 4**, we briefly present the ontological engineering and its contributions for the proposed cognitive vision platform. In particular, we focus on two ontologies built for the interoperability of the different modules of the platform: **a visual concept ontology** and **an image processing ontology**. The detailed description of these ontologies is given.

Then, the **chapter 5** is dedicated to the detailed description of the cognitive vision platform. For each sub-problem component, we propose a model for the dedicated knowledge base and a dedicated problem solving mechanism.

In the **chapter 6**, a real world application, i.e. the early diagnosis of rose diseases, is used to test and validate our platform. We first present and describe the biological problem and its objectives. Then, we show how to solve it using the proposed cognitive vision platform. The resulting image interpretation system dedicated to rose diseases and called ROSESIM is presented.

Finally, the **chapter 7** is a feedback on our contributions, i.e. the analysis of the expected requirements for the cognitive vision platform and how the proposed cognitive vision platform answers to these requirements. Future works and perspectives to improve the cognitive vision platform are presented.

Chapter 2

State of the Art on Image Interpretation

Introduction

Image interpretation and in particular semantic image interpretation consists in extracting the semantics of sensor data. It means the understanding and the semantic interpretation of image contents just like humans do. Semantic image interpretation is a problem of visual perception, i.e. the perception of our environment by visual sensors. Visual perception is the act of sensing a scene (its visible objects, structures and events), of recognizing it and of describing it with symbols. While humans perform visual perception effortlessly and robustly, visual perception is still a major challenge for artificial vision systems. According to [Trivedi and Rosenfeld, 1989], the research in visual perception is classified in:

- **neurophysiology** which studies the biological mechanisms of the human or animal vision,
- **perceptual psychology** which tries to understand the psychological aspects of the perception task,
- **artificial vision** which studies the computational and algorithmic aspects involved by the problems of image acquisition, processing and interpretation.

Other scientific domains as **cognitive sciences** or **linguistics** also deal with the research in visual perception.

We deal with the visual perception problem under the artificial vision point of view, i.e. **the building of automatic image interpretation systems**. The goal is to endow computers with a visual system enabling them to understand their environment from sensor data. Sensor data can be static images or sequences of images. The results of semantic interpretation can be object categorization but also event, situation or scenario recognition. Semantic image interpretation results can be used for different purposes like making decision (diagnosis problem), like monitoring issues (visual surveillance, health care monitoring), and so on. In the framework of our work, we are only interested in the semantic interpretation of 2D static images. Despite about thirty years of research, the building of automatic image interpretation systems is a difficult problem which was and which is still the basis of many research activities in both computer vision and artificial intelligence. The different scientific communities previously mentioned have a mutual influence on their research. Nevertheless, the aim of artificial vision is not to reproduce the mechanisms of human vision but rather to use its proper mechanisms to be close to the results and the performance of human vision.

In [Marr, 1982], David Marr proposes the first complete methodology for the design and the building of artificial vision systems. The works of David Marr are fundamental and the basis of many important works in image interpretation. David Marr did not believe in the usefulness of a priori knowledge in the interpretation process. His approach is a reconstructive approach. As a consequence, opposite philosophical approaches were proposed for the building of vision systems. The first section of this chapter is dedicated to a brief review on the main different approaches for the design of interpretation systems.

There is a great diversity of image interpretation systems with many motivations, models and building approaches. Nevertheless, a general statement on "old" or recent image interpretation systems is they are brittle technologies: they lack of robustness, of flexibility and of adaptation to various contexts and conditions of use. The emerging research field of **cognitive vision** encapsulates this attempt to achieve more robust, resilient and adaptable vision systems. It proposes to endow vision systems with cognitive faculties. **Cognitive vision** is a combination of computer vision and cognitive sciences. As an emerging discipline, **cognitive vision** is not yet well defined. Nevertheless, we try to give a brief survey on this emerging discipline in the second part of this chapter.

Then, we present some image interpretation systems that we have judged as the most significant in the state of the art.

2.1 "Philosophical" Approaches for the Building of Image Interpretation Systems

The aim of this section is to illustrate the differences between the different approaches of artificial vision: from the *traditional approach* of David Marr to the *purposive vision*. The main differences lie in the definition of the vision problem and in the manner to solve the problem.

2.1.1 The Marr Paradigm or the Reconstructive Approach of Vision

In the beginning of the eighties, David Marr proposes, in [Marr, 1982], a computational theory of human vision. This theory is the first complete methodology for the design of information systems. Marr's paradigm had and still has a great influence on the research in artificial vision. It is restricted to the 3D interpretation of single and static scenes. Marr introduces three levels to understand the running of and to build an information system:

- **the computational theory**: it describes what the system is supposed to do, what types of information it provides from other input information and what types of computations are needed.
- **the representations and the algorithms**: they represent the software point of view, i.e. how the computational theory can be carried out? What are the structures to represent information (input and output data) and what are the algorithms which manipulate them ?
- **the implementation**: it represents the hardware point of view, i.e. what about the physical realization of the algorithms. It also includes the programs and the hardware implementation.

According to Marr's paradigm, a vision system is a succession of bottom up (data-driven) processes which enable to transform information from an abstraction level to an higher abstraction level. Classically, the succession of processes is : **segmentation**, **reconstruction** and **recognition**. These three steps turn the image signal into a symbolic description of the scene. Marr identifies three representation levels associated with this succession of processes:

- the primal sketch: it aims at capturing the significant intensity changes in an image (regions, edges, intensity variations),
- the 2.5 D sketch: it is a midway between 2D and 3D representations. It reconstructs the relative distance from the viewer of the surfaces detected in the scene (depth map),
- the 3D representation: it represents the complete description of the scene in a viewer independent manner. A transition to an object centered coordinate system is made.

The main point of the methodology proposed by David Marr is the **hierarchical structure** of the processes and the representations. It advocates a set of relatively independent modules. The paradigm of Marr provides a nice theoretical framework for the understanding and the building of vision systems. However, important criticisms have also been made on this theory:

- the impossibility to reconstruct an exact representation of the interpreted scene,
- the sequential ordering of the approach,
- the lack of a priori knowledge, i.e. the approach of David Marr does not take into account knowledge about the scene and as a consequence, a semantic interpretation is not possible,
- the lack of the goal point of view in the process of vision : Marr's approach does not take into account the action of perception , i.e. the purpose under the task of visual perception

2.1.2 Active Vision

The main idea of the active vision paradigm introduced by Aloimonos in [Aloimonos et al., 1987] is that the visual perception activity is an exploratory activity. It underscores the fact that the observer is active and in interaction with its environment. Consequently, the active viewer is an additional source of information in the visual perception and interpretation process. The observer is able to acquire images from different points of views by the control of its visual sensor motions. The advocates of the active vision criticize the passive point of view of the traditional David Marr's approach and state that many fundamental problems of vision[1] are ill-posed[2] with this theory. In [Aloimonos et al., 1987], the author is interested in the motions of the sensors. To take into account these motions enables to introduce additional constraints to solve the ill-posed problems.

This approach is obviously inspired from the faculty of adaptation of the human vision with the motions of the head, the eyes and the pupils. The contribution of this approach

[1] shape from shading, shape from texture, shape from motion

[2] A problem is ill-posed when it does not satisfy one of the following criteria : to have a solution, to guarantee an unique solution, to depend continuously on initial data

is incontestable. However, current criticisms claim that it is more dynamic than active (the observer is moving but is not active). Indeed, it neglects the control of the sensors. Moreover, these works stayed too much theoretical and experimental validations were rarely made.

2.1.3 Active Perception

The notion of active perception was introduced by Bajcsy in [Bajcsy, 1988]. The visual perception and interpretation is defined as a problem of control. The aim is to plan control strategies to improve the knowledge of the system on its environment and for an intelligent acquisition of data. Active perception is defined as the study of the modeling of control strategies of visual perception. The modeling affects both the sensors and the processing modules. The modeling is divided into:

- local models: they represent the parameters of the different processing modules (sensor parameters, parameters of image processing algorithms). These parameters enable the prediction of the behavior and/or the results of the processing modules.

- global models: they are the parameters which represent the interaction between the different modules, i.e. how the different modules are merged (supervisor). The main idea is the introduction of a retroactive loop in the system. This retroactive loop enables to the system the acquisition of data only when they are needed by the system.

The active perception approach is interesting because it takes into account, in an explicit manner, not only the representations but also the processes which work on these representations. The perception strategy consists in searching for the succession of actions to obtain a maximum of information with a minimal cost. Works on active perception emphasize the three following important points:

- the explicit representation of both knowledge and reasoning using knowledge representations,

- the notion of reasoning process, the notion of control,

- the importance of a retroactive loop: i.e. the processing of data only when they are needed.

2.1.4 Animate Vision

Animate vision was proposed by Ballard in [Ballard, 1992]. This approach is based on the study of the purposive motions of the human eye during a visual task. Ballard considers the visual perception and interpretation in the context of an action. A 3D representation of the real world is not needed. Similarly to active vision, this approach considers vision as an ill-posed problem. The aim of animate vision is to add constraints by the information provided by the controlled motion of the sensors. The aim of this approach is to control the motion of the sensor to achieve focus of attention and gaze control tasks. This method aims at reducing the complexity of visual perception and interpretation tasks. An animate vision system can shift the sensors, change the focus and the angle of vision. Animate vision uses an exocentric coordinate system centered on the object. Concerning the implementation of the system, Ballard uses an active binocular head.

The main important idea of animate vision is the notion of strategy of visual search by the setting up of mechanisms of gaze control and focus of attention in the image. These mechanisms enable to analyze only the relevant parts of images.

2.1.5 The Purposive Vision

The purposive vision was introduced by Alomoinos in [Aloimonos, 1990]. It emphasizes the task oriented point of view of the visual perception and interpretation processes. It stresses the dependency between action (the goal of the visual perception task) and perception. The ability of complete reconstruction is not necessary. The aim is to derive only task relevant representations of images and to derive the processing modules and the implementation that correctly fit these representations.

In this approach, the basic idea is to break the initial problem into sub-problems. The work consists of the definition of the processing modules dedicated to each sub-problem and of the definition of a supervisor which manages the different modules. As explained in [Tsotsos, 1994], the break of a global problem into sub-problems and their grouping in a general module enable to improve visual perception and interpretation tasks.

Purposive vision is a very important approach. It gave birth to a wide range of works and applications. Some fundamental notions have inspired the research in cognitive vision. In particular, the key points are :

- the task oriented point of view of the visual perception and interpretation,
- the notion of minimalist systems: to achieve only the relevant tasks to reach the desired goal,
- the breaking up of complex tasks into more tractable sub-tasks.

The main criticism of purposive vision is its application dependence. Most of the vision systems built with the paradigm of purposive vision are highly application dependent.

2.1.6 The Direct Approach or Ecological Vision

The theory of ecological vision is based on the works of J.J. Gibson [Gibson, 1979]. This approach is opposite to the reconstructive approach. This approach underlines the relation between the system and its environment. It stresses the importance of the environment, of the nature of the light and of the goal of invariants in vision. Ecological vision assumes that luminous rays directly contain all the information needed for the recognition of the real world. This approach refutes the use of a priori knowledge and minimizes the importance of information processing and internal representations. Recent trends on appearance based vision are based on this theory. According to the ecological vision, the motion of the observer by involving a change of the optical flow enables to perceive the world. Moreover object function has a great importance on the visual perception. This point of view define the theory of the *affordance*. According to this theory, the semantics associated with object is relative to their functions.

2.1.7 Discussion on the Different Approaches

The different methodologies for the design of artificial vision systems have been the origin of various debates in the artificial vision research community. A set of these debates can be found in [cvg, 1994]. The differences of methodologies are partially explained by the influence of human vision theories in the proposed methodologies.

From a general manner, we make the distinction between two main points of views: the *passive* or traditional approach and the *active approach* which gathers the active vision, the active perception and the purposive vision.

The active approach is totally the opposite of the passive approach concerning the interaction of the system with its environment. The passive approach does not take care about the different characteristics of the environment whereas the active approach considers that a strong interaction between the system and its environment is needed. An interesting synthesis of these different methodologies can be found in [Sandakly, 1995].

2.2 Towards Cognitive Vision

Cognitive vision is not a new methodology for the building of vision systems. Cognitive vision is an emerging research field which encapsulates a general attempt to achieve more robust, flexible, resilient and adaptable vision systems by endowing them with cognitive faculties. Cognitive vision arises from the statement that existing statistical or knowledge based vision systems are brittle. Problems such as the re-usability in a wide range of fields, the environmental influence and noise are still major challenges for vision systems.

The emerging research field of cognitive vision does not advocate a unique approach for the building of vision systems but stresses a set of requirements that must be fulfilled by a cognitive vision system. In particular, a cognitive vision system should be able:

- to learn from experiences,
- to adapt itself to various and sometimes unforeseeable conditions,
- to choose between alternative solutions,
- to develop new strategies for analysis and interpretation.

The ultimate goal of cognitive vision research is a general-purpose system with the robustness and the resilience of the human visual system.

The discipline of cognitive vision gathers several various scientific fields like computer vision, pattern recognition, artificial intelligence, machine learning, cognitive sciences and knowledge engineering (among others). Cognitive vision is expected to be one of the major research areas in vision. The aim of this section is to give an overview of the emerging field of cognitive vision: i.e. its definition and its major challenges.

2.2.1 Definition and Major Challenges

Cognitive vision is an *emerging discipline in a pre-paradigmatic state* [Vernon, 2004]. As a consequence, a definition of cognitive vision which entirely satisfies all the scientific fields involved does not yet exist. Recently, researches and interesting thoughts on cognitive vision in the active research network ECVision[3] give birth to a research road map [Vernon, 2004]. This road map is a twenty year research plan on cognitive vision. It provides a good introduction on what is cognitive vision in a neutral manner and in what is the scientific foundations of cognitive vision. It enumerates a list of major scientific and methodological challenges.

[3]The European research Network for Cognitive Computer Vision Systems, www.ecvision.org

2.2.1.1 A Definition of Cognitive Vision

The ECVision road map [Vernon, 2004] states about the following working definition of cognitive vision: *A cognitive vision system can achieve, in an intelligent way, the four levels of generic computer functionalities of detection, localization, recognition and understanding.* The functionality of understanding refers to *the ability to comprehend the role, the context and the purpose of a recognized entity and its categorization on some basis other than visual appearance alone.*

As in the purposive vision paradigm, cognitive vision also stresses the dependency between action and perception. *A cognitive vision system has a purposive goal-directed behavior*, i.e. contrary to traditional vision systems concerned with obtaining a description or a reconstruction of the physical world, a cognitive vision system has to take into account the purpose and the intent associated to the observed entity with respect with its goal.

A cognitive vision system must be in interaction with its environment and *it can engage in adapting itself to unforeseen changes of the visual environment. It can also anticipate the occurrence of objects or events.*

To achieve these capabilities, a cognitive system is endowed with cognitive faculties: i.e. ability of knowing, ability of understanding, ability of reasoning and ability of learning things.

- **"Knowing"** refers to the *memory*: i.e. to store knowledge which is either provided a priori, learned from experiences or derived from existing knowledge. It concerns knowledge about the environment, about itself and about its relationship with the environment. This issue is strongly linked to the problem of **knowledge representation**.

- **"Understand"** refers to the recognition and the categorization of objects, situations or events across visual appearance.

- **"Reasoning"** refers to the process of using knowledge and cognitive process to explain things and solve problems. It can consist of making inferences of what is already known to explain an observation or to make predictions. A cognitive vision system makes deliberations about objects and events in the environment.

- **"Learning"** refers to the importance of the experience to cover real world problems. A system whose goal is to perform complex tasks under real world conditions must be able to learn from experience and adapt itself to unexpected changes. As emphasized in criticisms on the approach of Marr, a vision system can not be based on the hypothesis of a "closed world": it is unlikely that all the relevant knowledge involved in vision tasks can be acquired and provided a priori to the system. The automatic generation of new representations and models are needed. Moreover, learning capability is a mean for a continuous adaptation to the changing environment.

2.2.1.2 Major Challenges of Cognitive Vision

The ECVision road map proposes to carried out the research in cognitive vision in the context of seven major scientific and methodological challenges. The scientific challenges are:

1. *The advancement of method for continuous learning.*
   Cognitive vision systems are shaped by their experiences and learning is an important component of cognitive vision. As real world environments are not stationary, it must

be an open-ended process: i.e. the system must be able to continuously learn from observations and adapt itself to its environment. Continuous learning methods are required.

2. *The establishment of minimal architectures.*
   It refers to the identification of the minimal set of visual information processing modules that are needed to perform the complete semantic interpretation process. In particular, a minimal architecture must provide solutions to achieve the four functionalities of cognitive vision : detection, localization, recognition and understanding.

3. *Goal achievement.*
   Visual perception and interpretation are goal directed processes [Aloimonos, 1990] and a cognitive vision system is engaged in a goal directed behavior.

4. *Generalization.*
   It refers to the transferability of competences or skills from one context to another one.

The methodological challenges are concerned with :

1. *The utilization and advancement of system engineering methodologies.*
   Software engineering properties of cognitive vision systems are as much important as cognitive ones. Indeed, due to their requirements, cognitive systems will exhibit a high degree of system complexity. System complexity, system maintenance, system re-usability are important software engineering properties to take into account in the design of a cognitive vision system.

2. *The development of complete systems with well defined competences.*
   This challenge focuses on the construction of complete visually-enabled cognitive systems.

3. *The creation of research tools*

Our objective of a reusable cognitive vision platform for the design of semantic image interpretation systems takes parts in the challenge of the establishment of minimal architectures for cognitive vision systems. Moreover, as we are interested in re-usability and convenience, software engineering is a big part of our work.

2.3 Image Interpretation Systems

During the past 3 decades, a wide range of image interpretation systems have been designed with different philosophies, models and motivations. They can be classified according to several criteria:

- Their application dependence: i.e. if they have been designed for a particular application or with a re-usability purpose.
- Their level of knowledge modeling and representation.
- Their control strategy.
- Their functional architecture.
- The philosophical approach of their design: i.e passive, active or purposive vision.

- The methodological approach of their design.

The methodological approach to build image interpretation systems is inherently linked to the philosophical ones. Moreover, it has influence on the four first criteria, i.e. the genericity, the information representation, the control and the architecture. It also represents some trends in image interpretation. In particular, we have distinguished four methodological approaches:

- **Knowledge based vision** which is based on the explicit representation of a priori knowledge. It was the current trend in the eighties.
- **Decision theory based vision** which models the image interpretation as a problem of control.
- **Case based vision** which uses case based reasoning to solve image interpretation.
- **Appearance based vision** based on the learning of appearance based models of objects on 2D images. Appearance based vision techniques are widely used for categorization problems : i.e. the recognition of object class and not of object instance.

Our aim is not to make an exhaustive and detailed survey on image interpretation systems. We only present interpretation systems which are the most relevant in the literature and the most related to our work.

2.3.1 Knowledge Based Vision

A good survey on knowledge based interpretation systems can be found in [Crevier and Lepage, 1997]. Their principle is to interpret the scene in terms of a priori models representing knowledge of the world. Knowledge and knowledge representation is an important component of knowledge based vision systems. The knowledge involved in an interpretation process is divided into:

- **declarative knowledge**: it refers to the numerical and symbolic representations which describe the different entities known to the system. It represents knowing *"that"*.
- **procedural knowledge**: it represents knowing *"how"'*, i.e. the knowledge which describes how to extract and how to manage entities.

Moreover, the knowledge used in image interpretation can be classified into the following three types:

- **Scene domain knowledge**.
  This knowledge includes the description of intrinsic properties and mutual relations among objects in the scene. It is described in terms of the terminology used in the real world.
- **Image domain knowledge**.
  This knowledge is used to extract image primitives and features. It is described in terms of the terminology defined in the image domain.
- **Knowledge about the mapping between the scene and the image.**
  This type of knowledge is used to transform image features into scene features and vice versa.

We present some relevant knowledge based interpretation systems under the point of view of their architecture, their knowledge representation and their control strategy.

2.3.1.1 The VISIONS System

System Description

The VISIONS (Visual Integration by Semantic Interpretation of Natural Scenes) system [Hanson and Riseman, 78] , which was extended in SCHEMA [Draper B., 1996] is the result of a long term work at the University of Massachusetts. The aim was the building of a general integrated knowledge based interpretation system independent of any application. The goal of the VISIONS system is the construction of a symbolic representation of the three dimensional world depicted in an image. This symbolic representation includes the labeling of objects and the determination of their location in the space. A commercial product called **KBVision** was developed. In practice, the VISIONS system was only applied on images of outdoor static scenes.

1. **System architecture.**
   According to Marr's paradigm [Marr, 1982], the authors propose an image interpretation system based on a representation in three levels and based on the interaction between these levels.

   The functional architecture of the VISIONS system is composed of:

   - The low level which is dedicated to the extraction of image primitives. Low level processes manipulate pixel data. They produce intermediate symbolic events called tokens such as regions and lines with their attributes. It uses a specific library of image processing programs.
   - The intermediate level provides tools to organize the tokens into more abstract structures that can be associated with object instances. This intermediate level called ISR [Brolio et al., 1989] (Intermediate Symbolic Representation) is one of the main characteristics of the VISIONS system. It is a representation system and a management system for the use of the intermediate (symbolic) representation. ISR is based on database management methodology. It is an active interface between high level inference processes and image data. ISR provides tools for classification based on features, perceptual grouping, spatial access (e.g. the detection and the verification of neighborhood relations between objects) and constraint based graph matching between graphs of data and graphs of models.
   - The high level contains a semantic network of *schemas*. A *schema* is an expert system representing an object of the scene and dedicated to its recognition. The knowledge in a schema is not limited to the descriptions of objects; it includes information about how each object can be recognized. Schemas also control the invocation and execution of the low-level and intermediate-level routines with the goal of forming hypotheses about objects in the scene.

2. **Knowledge representation**
   ISR provides a frame based formalism for the representation of tokens.
   In the high level, we have mentioned that knowledge is organized into object descriptions called schemas. Schemas are themselves organized into relational networks. Each schema, which represents a particular object in the scene, has an associated

procedural component which represents the knowledge of how to recognize the particular object. Each schema has its own blackboard to store the local data concerning the particular object and schemas communicate asynchronously with each other by the mean of a global blackboard.

3. **Interpretation strategy**
   The interpretation strategy of VISIONS is the detection of the different objects present in the scene by the activation of the different corresponding schemas. The semantic network is used to order the activation of the different schemas. Each schema follows a strategy based on an hypothesize and test cycle called Object Hypothesis Maintenance.

Discussion
The VISIONS system separates the different types of knowledge and proposes a knowledge model for the three types of knowledge. The main characteristic of the VISIONS system lies in the knowledge representation with *schemas*. A model of knowledge is proposed for the three levels. Nevertheless, the knowledge representation by *schemas* is also one of the weakness of the system. Indeed, it does not enable the separation between knowledge and reasoning. This kind of representation does not enable the reuse of the knowledge and the reasoning which is generic for object recognition. From a knowledge acquisition point of view, the creation of schemas is a hard work. Moreover, from the maintenance and evolution point of view, the introduction of a new object implies the creation of a new schema. Machine learning techniques were proposed by Draper in [Draper and Hanson, 1991] to reduce this knowledge acquisition bottleneck. The SCHEMA system was upgraded with a learning module: the Schema Learning System described in [Draper et al., 1989].

Another important characteristic of this interpretation system is the management of the intermediate data. ISR is another main characteristic of the VISIONS system. It highlights the importance of the management of symbolic intermediate data in a global interpretation process and it proposes an application independent module to solve this problem.

2.3.1.2 The SIGMA system

Description of the system
The system SIGMA was developed by Matsumaya at the University of Kyoto [Matsuyama and Hwang, 1990].

1. **System architecture**
   From the software and functional architectural points of view, the system SIGMA is composed of a hierarchy of three reasoning systems:

   - The Low Level Vision Expert(**LLVE**).
     The LLVE contains the knowledge and the reasoning needed to perform image segmentation and feature extraction. It reasons about which method is the most effective based on knowledge of image processing techniques.
   - The Model Selection Expert(**MSE**).
     The MSE contains the knowledge of the mapping of the image data into high level objects of the scene.
   - The Geometric Reasoning Expert (**GRE**).
     The GRE is the central reasoning module of the system. Its knowledge source is

a symbolical world model representing structures and spatial relations among objects. Its reasoning called *evidence accumulation* integrates top-down and bottom-up processes.

The different expert systems communicate between each other by requests.

2. **Knowledge representation**
   The main knowledge concept is an object class (inspired from the frame based and object oriented knowledge representation scheme). An object class represents an abstraction of an object of the application domain. It not only describes properties and structures of object but also stores inferential knowledge, implemented by rules, on how to recognize it (rules for spatial reasoning).

   The LLVE uses two types of knowledge to conduct automatic image segmentation:

   - knowledge of fundamental concepts in image segmentation (e.g. types of image features and types of image processing operators) represented by a network representing the type structure in image segmentation,
   - knowledge about image processing techniques (e.g. how to combine the operators effectively) represented by a set of production rules.

3. **Interpretation strategy**
   The interpretation strategy loops on a two phase cycle: a model driven phase followed by a data driven phase. This loop is preceded by an initial segmentation of images which consists in a model driven extraction of image primitives by the LLVE.

   The interpretation cycle consists in :

   - Generation of hypotheses by rule activation (spatial reasoning about objects);
   - Evidence accumulation to test the different hypotheses, to check their consistency and to group them;
   - Solving the situation of conflicts.

Discussion
SIGMA proposes a clear and well formalized and modular architecture for image interpretation systems. The key points are:

- The architecture of the SIGMA system enables the separation of the three types of knowledge involved in image understanding : the scene domain knowledge, the image domain knowledge and the knowledge about the mapping between the scene and the image.

- The knowledge based segmentation expert and a top down goal directed segmentation deal with the imperfection of segmentation.

- An active reasoning process based on the use of evidence accumulation for spatial reasoning and object oriented knowledge representation.

Some criticisms can be made on the weakness of the spatial reasoning encapsulated in rules linked to semantic objects. The recent and growing body of research about spatial reasoning can help to solve this weakness. The knowledge acquisition is also a big problem in SIGMA.

2.3.1.3 MESSIE I and MESSIE II : Multi Expert System for Scene Interpretation and Evaluation

System Description

MESSIE-I and MESSIE-II ([Sandakly and Giraudon, 1994],[Sandakly, 1995]) were designed in the framework of works on scene interpretation in the PASTIS team at INRIA. They propose a definition, an implementation and the validation of a generic scene interpretation architecture.

1. **System architecture**
   The functional architecture of MESSIE II is composed of three hierarchical abstraction levels:

   - The scene level: it contains a scene representation with the semantic modeling of the scene objects. It also contains a generic scene interpretation specialist.
   - the semantic object level : it contains specialists for each semantic object (their detection strategy) and perceptual grouping specialists.
   - The image level which contains image primitive extraction specialists to adapt and tune image processing algorithms.

   The architecture of MESSIE II is a hierarchical blackboard with a control based on requests and events. These two mechanisms enable an opportunistic control of the interpretation process : the request mechanism is used when reasoning is goal driven and event mechanism when reasoning is data driven.

2. **Knowledge representation**
   In MESSIE II, the representation of semantic objects is generic. A semantic object is modeled from four view points: geometric, radiometric, spatial context and functionality. These view points are application and sensor independent. Moreover, the authors have identified and modeled a set of contextual objects needed for the interpretation : sensor, field of view and scene.

   Each semantic object is associated with an object specialist in the semantic object level. Each semantic object specialist has its own detection strategy. This detection strategy enables an efficient and easy management of low level feature extraction. The management of uncertainty and imprecision of extracted data and models are managed using the fuzzy set formalism and the possibility theory.

3. **Interpretation strategy**
   MESSIE II proposes a generic interpretation strategy implemented in two steps: a detection step where semantic object hypotheses are made by object specialists and a validation step to confirm or reject these hypotheses. The principle used for the detection step is that salient objects are first looked for and then used to support the detection of smaller objects.

The MESSIE II system has been validated with two applications: satellite image interpretation using multi-sensor fusion and 3D indoor scene interpretation.

Discussion

This work proposes a neat software and functional architecture for scene analysis. The key points are:

- The separation of the descriptive knowledge (the semantic object model) and the procedural knowledge (semantic object specialist).

- The representation of semantic object from four view points is generic and interesting.
- An opportunistic reasoning by the two mechanisms to trigger a specialist : requests and events.
- The validation of the architecture by two real applications.

Nevertheless, we criticize the notion of semantic object specialist. Indeed whatever is the abstraction level of a specialist, its representation is structured in the same way. Different reasonings for different levels are not possible. Moreover, MESSIE II is an intensive knowledge based system and the knowledge acquisition is a hard work. As a consequence, the use of the MESSIE system for various applications requires a long time design by the modeling of each semantic object and by the construction of semantic object specialists.

2.3.1.4 The Use of Semantic Networks: the AIDA System

System Description
AIDA is a knowledge based system for scene interpretation [Liedtke et al., 1997] .

1. **System architecture**
   The AIDA architecture is a network with different abstraction levels. Each abstraction level contains a set of concepts which are linked to concepts of the same level or of other levels.

2. **Knowledge representation**
   The representation of semantic, structural, topological and temporal knowledge about objects expected in the scene is made with semantic nets. The knowledge representation formalism is close to the knowledge formalism of the ERNEST system [Niemann et al., 1990]. The nodes of the semantic net model the objects of the scene and their appearance in images. The different abstraction levels are represented as different layers in the semantic net. Two classes of nodes are distinguished: concepts are generic models of objects and instances are realizations of the corresponding concept in the observed scene. They are used to distinguish the symbolic meaning of the objects and their visual appearance in the image. The relations between objects are described by edges or links forming the semantic net. It exists five types of links: specialization, composition, instantiation, concretisation and modeling.

3. **Interpretation strategy**
   The aim of the system is to use the prior knowledge represented in the semantic net to generate a symbolic description of the scene observed in a single image or a sequence of multi-sensor and/or multi-temporal images. The symbolic scene description consists in instances. The interpretation strategy is application independent and separated from the knowledge. It is explicitly represented by a set of generic rules which define the action to perform during the use of the semantic net. They are divided into rules for hypothesis generation, rules for instantiation, rule for specialization and rule for multiple bindings.
   It consists in an iterative combination of top-down and bottom-up processes. The system first generates hypotheses about scene objects and their properties which are consecutively verified in the image data. The end user initiates the interpretation process by the selection of a concept (a node in the semantic network). It represents the goal to achieve. Then, the system tries to instantiate this concept in image data with a top-down, bottom-up cycle. The top down analysis consists in the search of

all the related concepts using the links of the semantic network. Then in a bottom up analysis, all the selected concepts are instantiated. Each possible interpretation is documented by a search node N which contains all concepts and instances with their current interpretation state. Competing interpretations are represented in a search tree and selected with a A*-algorithm. An inference engine controls the execution of the rules.

Discussion
The main advantage of the AIDA system is the generic interpretation strategy and the representation by semantic nets. Moreover, in the AIDA system, the integration of the temporal dimension is proposed. Nevertheless, the semantic network does not enable the separation of the different types of expertise and the knowledge acquisition is an intensive work.

2.3.1.5 Cooperation of Knowledge Based Systems for Natural Complex Object Recognition

Description
The Orion team has much expertise in the use of explicit knowledge to solve complex image analysis problems. Knowledge based approaches were proposed to automate the use of an image processing library (program supervision techniques) and to automate the problem of single object recognition [Thonnat, 2002]. In [Ossola and Thonnat, 1995b], [Ossola and Thonnat, 1995a], a cooperative architecture based on two knowledge based systems is presented.

1. **System architecture**
   The architecture of the system is a distributed architecture composed of two knowledge based systems.

   - The first system dedicated to image processing enables to process an image and to describe the object it contains. To process images in an intelligent way, i.e. to dynamically construct the process with respect to variable conditions, program supervision techniques are used.
   - The second system interprets the data describing an object in order to classify it.

   Two KBS shells OCAPI [Clement and Thonnat, 1993a] and CLASSIC [Granger, 1985] are used to build the two knowledge based systems. The global architecture of the system is depicted in figure 2.1 (from [Ossola and Thonnat, 1995b]).

2. **Knowledge representation**
   This architecture provides dedicated predefined structures to represent the knowledge of each module.

   - The data-interpreter imposes predefined structures for knowledge.
     Prototypes describe the different object classes. They are frames organized in a **prototype tree** reflecting the specialization relations between the different object classes. **Inference rules** enable data abstraction, essentially the conversion of numerical features to symbolic descriptions. They are organized in rule bases attached to prototypes and fuzzy predicates in rules enable to manage imprecise and uncertain knowledge.

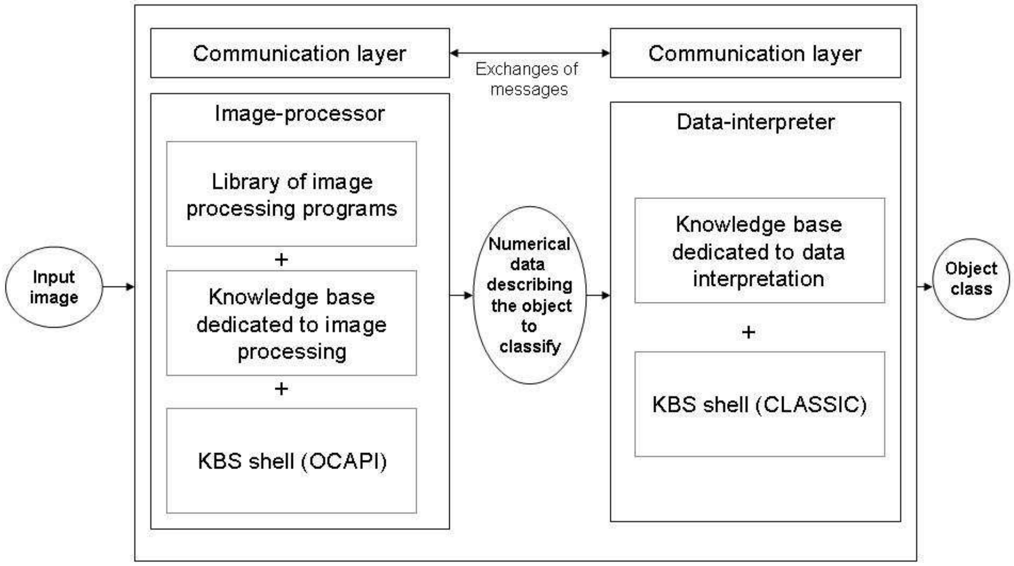


Figure 2.1: The global architecture of the object recognition system

- The main component of the knowledge structures of the image-processor are frames for descriptive knowledge (goals and operators) and rules for inferential knowledge. The image processing knowledge is principally structured in **goals** which are image processing functionalities, **operators** which contain the specific knowledge to solve a goal, and **rules** to represent the knowledge of how to initialize, choose, evaluate and adjust operators.

3. **Interpretation strategy**
   Each module has its proper reasoning strategy. The reasoning of the data-interpreter is based on the depth first tree traversal of the **prototype tree** through a three phase cycle: **data abstraction** (using inference rules), **matching** with a prototype and **refinement of the classification**. The reasoning of the image-processor follows a cycle in four phases: **planning**, **execution** of programs, **evaluation** of the results, and **repair**.

   The communication layer enables to each module to send messages to the other. These messages enable top-down and bottom-up strategies. The global process is not sequential. The data-interpreter can get new data from the image-processor during its classification process.

Discussion

The distributed architecture presented here has several advantages:

- The exploitation of the different expertise directly. The use of KBS shells enables easy construction of knowledge bases by experts: image processing experts for the image-processor and application domain experts for the data-interpreter. It enables the capitalization of their expertise.

- The separation of not only the different types of knowledge but also of the different types of reasoning involved in the object recognition problem (planning for image processing and classification for object categorization).
- From the software engineering point of view, knowledge based systems are easy to maintain by modifying the knowledge base and they are extensible by adding pieces of knowledge.
- Using program supervision techniques enables an intelligent and adaptable management of the image processing problem.

Nevertheless, the proposed architecture has also some weaknesses.

- The domain knowledge modeling does not enable the representation of spatial relations. The proposed architecture is dedicated to single object recognition so that we could think that spatial relations are not useful. Nevertheless, if we want to manage complex composite objects with interrelated sub-parts, they are essential.
- The problem of the mapping between sensor data and symbolic data is not treated as a problem as such. It is encapsulated in inference rules of the data-interpreter. As a consequence, they are application dependent. Moreover, from a domain knowledge acquisition point of view, these rules are difficult to build by domain experts.
- Another problem lies in the evaluation of image processing results. The interpretation must rely on the results provided by the image-processor.

2.3.1.6 Interpretation of Spatial Structures

In this section, we present some works which considerably use spatial relations to manage the interpretation process. Indeed, even if it is commonly admitted that spatial relations play an important role in the recognition of structures, they are often restricted to be properties of semantic objects in semantic image interpretation systems. Nevertheless, the management of, and reasoning on spatial relations is essential when a scene object can not be discriminated with its proper characteristics.

In [Le Ber and Napoli, 2002], an interesting object-based representation of spatial structures and relations is presented. The authors present a system for representing and classifying spatial structures and relations. The aim is to recognize landscape patterns from satellite data. In the system, a priori knowledge is represented as (1) classes which denote patterns of landscape and as (2) spatial relations which denote qualitative topological spatial relations. Spatial relation instances are linking classes together. They are organized into a specialization hierarchy, organized in Galois lattices. The lattice representation facilitates the verification of relations on the images and the inference of relations. These lattices are used in order to recognize spatial structures on maps based on satellite images.

In [Matsakis et al., 2001], a system which generates linguistic descriptions of natural scenes is described. The authors propose a new family of fuzzy directional relations that rely on the computation of histograms of forces. These relations are the inputs of a fuzzy rule base that produces linguistic descriptions.

In [Colliot et al., 2004], the role of spatial relationships for the model-based recognition of structures is emphasized. They propose to model spatial relationships in the fuzzy set framework [Bloch and Ralescu, 2003]. The application is the recognition of brain internal structures in MR images. A priori knowledge of the structures is modeled using a data structure called *synthetic hierarchical graph*, i.e. an attributed hierarchical graph. Nodes are brain structures and edges, relations between brain structures. Spatial relations are represented in a fuzzy set framework to handle imprecision. The fuzzy sets represent the degree of satisfaction of the relation with respect to a reference object as described in [Bloch et al., 2003]

2.3.1.7 Use of Conceptual Spaces for a Cognitive Vision Architecture

In [Chella et al., 1997], a cognitive architecture for artificial vision is presented. The authors aim at an autonomous intelligent system. The originality of their approach lies in the existence of a conceptual level based on the notion of conceptual space [Gärdenfors, 2000]. A conceptual space is a metric space in which entities are characterized by a number of quality dimensions (color, spatial coordinates, size,..). The dimensions of conceptual space represent qualities of the environment independently of any linguistic formalism or description. This representation enables the modeling of natural concepts (real physical objects) as convex regions in the conceptual space and it enables reasoning as concept formation, induction and categorization [Gärdenfors, 2000].

1. **System architecture and representation**
   The system architecture is based on the three cognitive level representation proposed by Gärdenfors [Gärdenfors, 2000]:

   - The *sub-symbolic level* in which information is strictly related to sensory data.
   - The *conceptual level* in which the information is characterized by a multiple dimension metric space. In the proposed architecture, the dimensions of the conceptual space are the parameters of the 3D geometric primitives which compose the scene. The representation of a great variety of familiar shapes (principally human artifacts) is made according to the schemas of Constructive Solid Geometry (CSG). A *knoxel* represents a generic point in the conceptual space, i.e a vector $k = (a_x, a_y, a_z, \varepsilon_1, \varepsilon_2, p_x, p_y, p_z, \varphi, \vartheta, \psi)$. a_x, a_y, a_z are the lengths of the super-quadric axes, $\varepsilon_1, \varepsilon_2$ are form factors (longitude, latitude), p_x, p_y, p_z the three center coordinates and φ, ϑ, ψ are the three orientation parameters.
   - The *linguistic level* in which information is expressed by a symbolic language. The role of the linguistic system is to provide a concise description of the perceived scene in terms of a high level language suitable for symbolic knowledge based reasoning. Description logics are used as representation scheme. The **terminological component** contains the descriptions of the concepts relevant for the represented domains. The **assertional component** stores the assertions describing the perceived scene. This representation is useful to maintain a distinction between the generic conceptual knowledge, independent of the perceived scene and the assertions concerning the current perceived scene.

2. **Interpretation strategy**
   The interpretation strategy is driven by focus of attention mechanisms: the exploration process of the perceived scene is driven by linguistic and associative expectations.

Discussion

The originality of this work is to link together research in computer vision with research in symbolic models of knowledge representation and reasoning. The design of the proposed architecture is based on two cognitive hypotheses: the existence of a conceptual representation level and the active role of focus of attention mechanisms in the interpretation process. Moreover, conceptual spaces provide a geometric treatment of knowledge which bridges the gap between the symbolic and sub-symbolic approaches. As such, they can be used for the study of the problem of creating and maintaining in time the connection between symbols and sensor data that refer to the same physical objects: the anchoring problem [Coradeschi, 1999]. A computational framework for anchoring based on conceptual spaces is presented in [Chella et al., 2002].

2.3.1.8 Conclusion

In this section, we have presented a set of interesting knowledge based image interpretation systems. The study is not exhaustive. Our choice is essentially motivated by the presentation of complete interpretation systems, i.e. which deal with the three sub-problems of image interpretation (image processing, Signal/symbol matching and interpretation) or to systems directly related to our work. The analysis of the different interpretation systems enables us to draw some conclusions on the building of knowledge based interpretation systems:

- A first characteristic is the existence, for all these systems, of at least three different semantic levels: the low level, the intermediate level and the high level level. These levels refer to the abstraction level of the handled data and knowledge. They reflects the different data transformations useful for an interpretation system.

- This existence of semantic levels does not automatically imply the building of systems into three different modules [Ossola et al., 1996]. An important question concerns the mapping between the scene and the image. This problem is rarely considered as a problem as such. It is always encapsulated in the high level problem through different forms. Interesting works in the intermediate level are the ISR approach of the VISIONS system and the use of conceptual spaces. Other very interested works on this intermediate level are found in the Robotics community. They refer to the Anchoring problem [Coradeschi, 1999].

- Modularity is an important characteristic for image interpretation systems. Modularity makes easier their design and their building but also their running and their maintenance.

- Blackboard architectures are commonly used as software architecture. In a blackboard architecture, specialists communicate through a shared fact base. Blackboard architectures favor a high modularity and data sharing. Nevertheless, a drawback of blackboard systems is they do not enable the representation of different types of reasoning. Whatever is the level of reasoning, it usually consists in the activation of the rules linked to knowledge sources according to requests or events.

- The choice of the knowledge representation language is important: semantic networks, production rules, frame based systems, object based systems, description logics, hybrid systems ...

- The representation of spatial relations and spatial reasoning is essential but rarely studied as a specific problem for image interpretation.

- Owing to the representation of a high level knowledge, knowledge based vision enables to perform semantic interpretation

Several majors criticism have been made concerning knowledge based vision systems. In [Draper B., 1996], Draper states that knowledge based systems are ad-hoc. He identifies a set of open problems in knowledge based vision : (1) the knowledge engineering problem, i.e. the knowledge acquisition bottleneck when large amount of knowledge is needed, (2) the problem of the management of vision procedures (i.e. image processing programs) and the (3) system integration problem. He proposes to cope with these problems by modeling the image interpretation problem as a problem of control : i.e. to find the best sequences of vision procedures to satisfy a goal. Related works are presented in the following section. The general assessment of knowledge based vision systems is their lack of robustness and their lack of flexibility for varying conditions of the environment. They are not able to adapt themselves to unforeseeable conditions. Approaches based on a priori knowledge are sufficient with a close world assumption.

2.3.2 Image Interpretation Systems Based on Decision Theory

This section deals with the use of decision theory as a basis for intelligent image interpretation. We describe a set of works which aims at building image interpretation systems which have properties of robustness, application independence and easy maintenance. The main idea is to reduce as much as possible the role of the human expertise in the building of interpretation systems by machine learning techniques. The notion of second generation interpretation systems refers to these works. This notion was introduced in the beginning of the nineties based on a negative assessment of knowledge based vision systems dressed by Draper in [Draper B., 1996]. In this paper, it was argued that knowledge based vision is too ad-hoc and too dependent on human expertise during their design. The principle of second generation interpretation systems is to model image interpretation as a problem of control over a space of vision operators [Draper, 2003]. These systems explicitly model the control process itself, typically either as a Bayes net or as a Markov model, and they use domain independent learning mechanisms for the automatic acquisition of control strategies.

2.3.2.1 Bayes Net Approach

In [Rimey and Brown, 1994], the TEA-1 selective vision system is described. It gathers evidence to answer a question about a visual scene. The evidence is used by Bayesian networks to update a belief about the answer to the question and determine whether more evidence is needed, what kind of evidence to collect and where to seek it.

Four Bayesian networks are used to represent the domain and the task knowledge. A *PART-OF network* encodes the structure of the scene, the *expected area net* models the geometric relationships between the different objects, the *IS-A network* is used to classify the objects into classes and the *task net* encodes the task specific knowledge.

The results of visual processing are fed as evidence into the PART-OF network expected area and IS-A networks and the beliefs associated with the different nodes are updated. The relevant probabilities are then combined into a package that is used as evidence by the task net. Inference is then performed on the task net, which yields an answer to the question underlying the task in the form of a probability distribution. The system then decides whether it needs to acquire more data or whether it is satisfied with the current solution. Bayesian networks are also used in recent works [Buxton and Gong, 1995].

2.3.2.2 Markov Model Approach

ADORE and MR ADORE

ADORE (ADaptive Object REcognition) described in [Draper et al., 2000] is an object recognition system with a Markov based control system. Its goal is to avoid knowledge engineering by approaching object recognition as a supervised learning task. In ADORE, the object recognition is formulated as a Markov Decision Problem (MDP), where the optimal vision operator is selected based on the current state of the system. Typically, a Markov decision problem is defined by a set of states, a set of actions and a control policy that map states onto actions. In ADORE, states are data tokens produced by vision procedures (region, intensity image, active contours, set of line segments). Actions are vision procedures which change the state of the system by producing new data tokens from the current data tokens. Control policies map data tokens onto vision procedures and therefore, they select the next action in the recognition process. The ADORE system is divided into two components: an off line learning system and a run time execution monitor. The control policies are learned by a supervised approach with an off line learning system based on the training of Q-functions. In the run-time execution, the system implements the learned control policies by iteratively applying vision procedures to data. ADORE was applied for the recognition of different types of houses in aerial images.

MR ADORE (Multi Resolution ADaptive Object REcognition) [Levner et al., 2003] goes a step further than ADORE in the avoiding of human knowledge engineering. The aim is the automatic building of image interpretation systems. MR ADORE investigates methods and techniques that minimize the need for human intervention while maximizing the performance and portability of interpretation systems.

In [Peng and Bhanu, 1998], an adaptive integrated image segmentation and object recognition system is proposed. The authors stress the importance of the adaptability to real world changes of the segmentation problem, in order to improve the interpretation process. The authors criticize the general open loop approach for image interpretation. They propose to use the model matching confidence degree as feedback to influence the image segmentation process. A team of stochastic learning automata is used to represent both global and local image segmentation. Reinforcement learning is used. It closes the loop between model matching and image segmentation. The main advantage of reinforcement learning is that it requires only knowledge of the goodness of the system performance rather than details on the algorithm. As a consequence, the proposed method is independent of any specific image segmentation algorithm.

2.3.2.3 Computational reflection and Control System theory

In [Robertson, 2000], Paul Robertson proposes an architecture for self-adaptive image interpretation based on the computational reflection and control system theory. The proposed architecture named GRAVA was applied for multiple applications : face recognition in [Robertson and Laddaga, 2002] and aerial image understanding in [Robertson, 2000]. These works take part in the DARPA program named *"Automatic Software Composition"* dealing with **self adaptive softwares**: i.e. softwares which auto-evaluate their behavior and which change it according to the evaluation. Thereby, self adaptive softwares have abilities to:

- **monitor** the state of the computation,
- **diagnose** problems,

- **make changes** to correct deviations from the desired behavior.

Robertson states that one of the reasons for the brittleness of image interpretation systems is their inability to predict their environment. He proposes a self adaptive image interpretation system that is capable of automatically monitoring its own performance and adjusting to varying situations. The computational reflection enables the operators to change their own semantics in order to adapt to varying situations. This approach provides adaptability and ease of maintenance. Knowledge is associated with operators to enhance them with the capability to monitor and evaluate their own performance.

The interpretation strategy is modeled as a feedback control system. It uses operators and parameter settings (models) in order to infer an interpretation for the environment. This information is used as feedback in order to improve performance by choosing better operators and models.

2.3.2.4 Conclusion

The modeling of image interpretation as a problem of control is a good way to achieve more robust, more flexible and more adaptable image interpretation systems. Nevertheless, they are too "image-based" from our point of view. To perform semantic image interpretation, i.e. to give a high level meaning of the content of images, knowledge about what is the semantics is needed.

2.3.3 Case Based Reasoning for Image Interpretation

Case Based Reasoning (CBR) is a problem solving approach which solves new problems by adapting previously successful solutions to similar problems. CBR traces its roots to the cognitive studies of Roger Schank and his students at Yale University in the early 1980s. Case based reasoning is a flourishing paradigm for reasoning and learning in artificial intelligence [Aamodt and Plaza, 1994]. Case based reasoning is a cyclical process which consists in:

- Retrieving the most similar case;
- Reusing the case to attempt to solve the problem;
- Evaluating and revising the proposed solution if necessary;
- Retaining the new solution as a part of a new case.

Case based reasoning is not well established in the image interpretation community or cognitive vision community even if it is an interesting approach for the representation of memory and the use of existing experience. We review some interesting image interpretation systems based on CBR.

In [Grimnes and Aamodt, 1996] a two layer case based reasoning architecture for medical image understanding is proposed. The architecture is based on acquisition of radiological knowledge. A low level case based reasoner works on a segment case base which contains individual image segments. These cases with labels are indexes for another case based reasoner which works on an organ interpretation case base.

In [Perner, 2001], Petra Perner stresses the attractiveness of case based reasoning for image interpretation. She proposes a generic hierarchical architecture based on three case based reasoning modules. The three levels correspond to the three traditional abstraction levels. A dedicated case based reasoning module is used at all the levels of image interpretation.

- **Case based reasoning for image segmentation**
  The case based approach is used for algorithm parameter learning. A case contains an image, contextual information (as image acquisition information) and algorithm parameters. Finding the best segmentation for the current image is done by retrieving similar cases in the case base. Similarity is computed using non-image and image information. The evaluation is done by a measure of dissimilarity between the original image and the segmented image. If the evaluation is bad, the learning module is activated to build a new case. Some interesting works can also be found in [Ficet-Cauchard et al., 1999].

- **Case based reasoning for Signal to Symbol Mapping**
  The author states the fact that a combination of low level features (image data) is often necessary to express a symbolic feature. In her architecture, the signal to symbol mapping is modeled as a mapping of several low level features to a symbolic feature by a phase of selection and the creation of a mapping function using an induced decision tree. The case based mechanism for this level is not described in the paper.

- **Case based for Interpretation**
  The high level interpretation is done by the comparison with cases of the fact base. At the interpretation level, a case contains a symbolic structural description of the image (attribute value pair, feature based representation, attributed graph, semantic network, ...) and the solution of the interpretation. The interpretation consists in the retrieval of the most similar case by case comparison. This comparison is dependent on the symbolic structural representation. In the case of a graph, it consists in a sub-graph isomorphism. To manage large scale case base, the case base is indexing using decision tree induction and incremental conceptual clustering.

Moreover, a case acquisition tool and tools for maintenance were added to the architecture. This architecture was applied to different medical applications : the recognition of air bone fungi in [Perner et al., 2003a] and the recognition of cells [Perner, 2001].

2.3.3.1 Conclusion

Case based reasoning is a convenient method to add robustness and flexibility in image interpretation systems. Nevertheless, the choice of an adequate representation is an application dependent problem. The main advantages of case based reasoning systems lie in the easiness of their reasoning strategies.

2.3.4 Appearance Based Vision

As our work is not based on appearance based vision, we briefly define what appearance based vision is. The appearance based vision paradigm emerged in the beginning of the nineties. The principle of appearance based vision is to recognize objects directly from images without a priori knowledge. Object representations, which only use information from images, are called appearance based models. Appearance based vision systems are generally composed of a learning module and of an execution module. The state of the art on appearance based vision concerns a lot of works including:

- The representation by appearance based models based on global descriptors, local descriptors, geometric invariants,...

- The different learning techniques to build the appearance based models.
- The mapping techniques between models and images.

A very good introduction to appearance based vision is found in [Forsyth and Ponce, 2002].

2.3.5 Some Works on Cognitive Vision

In this section, we present some works on Cognitive Vision. They are a representative but not an exhaustive view of the current state and trends of the research in cognitive vision. Particularly, there are a lot of European projects. Due to the wide range of scientific disciplines involved in research in cognitive vision, some of these projects are far from the scope of our objective. Some links on these projects can be found on the ECVision web site[4]. We only present some results on the CogVis project that is the most related with the scope of our work.

CogVis: Cognitive Vision

The CogVis project is a European Union funded collaborative project to study the design of cognitive vision systems. The objective of the CogVis project is to provide methods and techniques that enable construction of cognitive vision systems that can perform task oriented categorization and recognition of objects and events in the context of an embodied agent. The functionality will enable construction of mobile agents that can interpret the action of humans and interact with the environment for tasks such as fetch and delivery of objects in a realistic domestic setting. A cognitive vision system is defined as a system that uses visual information to achieve:

- Recognition and categorization of objects, structures and events.
- Interpretation and reasoning about scene and events.
- Learning and adaptation.
- Control and attention.

The four points are the four work packages of the CogVis project.

Concerning the first problem, i.e. the recognition and categorization of objects, structures and events, most of the results are related to appearance based modeling of objects and scene and their learning with supervised or unsupervised machine learning techniques [Stocaj, 2003], [Leibe and Schiele, 2003], [Schiele, 1997].

Our work is more related to some interesting CogVis results on the interpretation and reasoning about scene and events. The aim of this work package is to develop conceptual structures for high level knowledge and reasoning processes. In [Neumann and Weiss, 2003], Neumann is interested in multi object scene interpretation. He highlights the difference between the multi object scene interpretation problem and the object recognition and categorization problem:

- The structures to represent and to interpret consist in several spatially and temporally related components (object, object configurations, situations, occurrences, episodes).

[4]The European research Network for Cognitive Computer Vision Systems, www.ecvision.org

- The scene interpretation exploits the high level knowledge and the contextual information.
- The interpretation results are described in qualitative terms, omitting geometric details.

A framework for multi object scene interpretation based on logic based conceptual models is proposed. The main conceptual entity is aggregate : e.g. a set of parts which compose a concept and the constraints they satisfy. The high level knowledge is described by taxonomic and compositional aggregate hierarchies. The interpretation process is modeled as a partial description in terms of instances of concepts in the knowledge base. It is a stepwise process. The different interpretation steps are aggregate instantiation, instance specialization, instance expansion and instance merging. Different cognitive situations can be treated with these interpretations steps. The representation scheme is the description logic ALCF(D).

Some interesting works deal with the incorporation of Qualitative Spatial Reasoning (QSR) into practical machine vision systems. In [Cohn A.G, 2002], an approach for building cognitive vision systems using qualitative spatial temporal representations automatically inferred from image data is presented. The interpretation problem is modeled as an abduction problem: the system searches for explanations, phrased in terms of the learned spatio-temporal event descriptors, to account for the video data.

2.4 Conclusion

In this chapter, we have presented different approaches for the building of interpretation systems and some interesting image interpretation systems. Our aim is not to state that an approach is better than another one. Indeed, although the building of image interpretation systems has represented a major point of interest during the last decades, a general purpose system with the robustness and the resilience of the human visual system still does not exist. We think that the building of such a system could not be achieved by only one of the approaches presented above but by an intelligent cooperation between all these approaches. It is the main idea of cognitive vision.

In the previous chapter, we have identified a set of important issues for semantic image interpretation among which is the importance of a priori knowledge. We are interested in semantic image interpretation and not just in object recognition and categorization. We think that a priori knowledge about the content of the image is needed. As a consequence, our approach is knowledge based. The scope of our work concerns application domains with much well formalized expertise.

Interesting works on spatial relation representations and on spatial reasoning and works on the intermediate level ([Brolio et al., 1989] and [Coradeschi, 1999]) are good ways to improve knowledge based vision systems. Moreover, to cope with the identified drawbacks of knowledge based vision systems, we are also interested not only in the cognitive issues involved in the building of systems but also in the software engineering ones.

Chapter 3

Proposed Approach : A Cognitive Vision Platform for Semantic Image Interpretation

3.1 Towards A Cognitive Vision Platform for Semantic Image Interpretation

3.1.1 Our Objectives

We aim at making advances in the field of automatic semantic image interpretation by the design of a generic and reusable cognitive vision platform dedicated to the building of semantic image interpretation systems. This work takes place in the emerging field of cognitive vision [Vernon, 2004]. Rather than yet another image interpretation system dedicated to a specific application, the cognitive vision platform is a structured set of application independent and reusable tools for the design of complete semantic image interpretation systems.

By **complete** semantic image interpretation systems we mean systems capable of intelligent and robust behavior in the resolution of all the sub-problems involved in semantic image interpretation. As explained in [Vernon, 2004], such systems achieve functionalities of detection, description, recognition and understanding. We think that considering the system in full is important. It is not enough to focus on the development of complex intelligent techniques for just one of the previous functionalities. We aim at defining a minimal architecture for semantic image interpretation. This minimal architecture defines the necessary parts of a semantic interpretation system, what role they play and how they interact between them.

We are interested in both the cognitive issues and the software engineering ones involved in the design of semantic image interpretation systems. We propose a system design approach based on the reuse of existing and experienced components. The cognitive vision platform is a generator for semantic image interpretation systems: i.e. a generic environment composed of generic tools that can be reused for different applications.

Our goal is :

- To identify and propose a model for the different sub-problems involved in the global semantic image interpretation problem.

- To build a cognitive vision platform by the integration of the identified sub-problem models and by the definition of their interaction.

- To validate the proposed platform by the building of a semantic image interpretation system for a real world application.

Our work takes part in the identification of minimal architectures for cognitive vision systems defined by the ECVISION road map [Vernon, 2004] as one of the priority topics and challenges for research in cognitive vision.

In the framework of this thesis, we restrict our work with some strong hypotheses:

- We are interesting in the interpretation of **2D scenes** representing natural objects in their natural environment.
- There is a **unique** image acquisition system.
- The application domain has **a well defined expertise** and application domain experts exist.

3.1.2 Some Requirements for a Cognitive Vision Platform

According to the emerging definition of cognitive vision [Vernon, 2004] and with the motivation of designing a generic and reusable platform, we identify a set of properties or general requirements that the platform must satisfy. Actually, some of these requirements must be satisfied by the cognitive vision platform itself and others are requirements for the semantic image interpretation systems designed with the platform. The set of these requirements is:

1. **Re-usability**
   The property of re-usability is required for both the cognitive vision platform and semantic image interpretation systems built with the platform.

   From the point of view of the cognitive vision platform, the property of re-usability refers to the ability to design the different platform components in a manner so that they can be reused for a wide range of applications without significant modifications. The basic idea is that rather than building semantic image interpretation systems from scratch for each application, they can be built from reusable ready made parts. These reusable ready made parts are the components of the platform. This property enables to considerably reduce the development cost of systems. We are interested not only from the software engineering point of view but also from the point of view of knowledge and experience sharing for a specific problem.

   From the point of view of semantic image interpretation system, the property of re-usability refers to the ability of the system to be used in different contexts in a convenient way.

   This property of re-usability can be carried out by :

   (a) **Application independence**
   The cognitive vision platform must be independent of any application domains. It must provide generic tools. Indeed, most of the work that can be found in the literature about image interpretation systems is generally motivated by application domain needs [Hanson and Riseman, 78], [Matsuyama and Hwang, 1990]. These works have been successful but none of them are totally application independent. Only few works have been interested in a complete generic architecture for image interpretation [Sandakly and Giraudon, 1995]. A complete application and sensor independent architecture means the independence on both:

- the scene modeling (knowledge representation),
- the reasoning strategies.

As a consequence, this requirement means the identification of what knowledge and what reasoning is generic for all the sub-tasks of semantic image interpretation. Another consequence is the separation between the reasoning strategies and the a priori knowledge used by them. To achieve this requirement we have to make good use of research on system and knowledge reuse: in particular ontological engineering and problem solving methods.

(b) **Flexibility, adaptability and robustness**
Semantic image interpretation systems operate in a dynamic environment where conditions can change in an unforeseeable manner. Traditional automatic image interpretation systems have tended to be very brittle and to have poor performances in unconstrained environments whereas the human vision system has a high robustness in performance even in natural environments. Dealing with environment changes is a major problem of cognitive vision.
The property of **flexibility** is useful to accommodate such varying operating conditions. To cope with such varying operating conditions, the cognitive vision platform must have the property of **flexibility**. The property of **flexibility** means that different alternatives, i.e. different algorithms and tuning parameters, must be provided for each sub-task involved in the global semantic image interpretation process.

As a consequence of **flexibility**, semantic image interpretation systems must have the quality of **self adaptation**. Systems must be able to adapt themselves according to the current context (context awareness), to the varying operating conditions. They must exhibit a high level of autonomy by the automatic selection of the different algorithms and tuning parameters to choose according to the current context.

At last, systems must maintain a level of sensitivity to parameter variations and must ensure a desired quality of performance. It represents a property of **robustness**. The property of **robustness** refers to the invariance of system performance to changing conditions.

2. **Maintenance**
The property of maintenance is an important software engineering quality which measures the extent to which a system can be modified at the lowest possible cost. This property of maintenance is linked to the property of **evolution** of the platform. This property of maintenance can be carried out by :

(a) **Modularity**
Modularity is a classical software engineering property which measures the extent to which a system is divided into components, called modules, which have a high internal cohesion and a low level coupling between each other. It contributes to system maintenance.

As mentioned before, the image semantic interpretation problem can be divided into different sub-problems which involve different types of knowledge and reasoning. For example, the knowledge and the strategies used to classify an object are different than those which are used when we choose the best algorithm to detect it. The discipline of software engineering suggests that systems whose components are designed according to function or specific task to solve are easier to design, build, and maintain. This functional or problem specific decomposition leads to a need of modularity.

The modularity of the platform enables to combine and specify the different modules in a way corresponding to the particular requirements of an application. Nevertheless, modularity implies the problem of information exchange and sharing between the different modules.

3. **Convenience**
   The property of **convenience** refers to the easiness of use. For the cognitive vision platform, this property means the easiness of use of the platform by the different experts involved in semantic image interpretation (application domain expert, image processing expert and visual data management expert). From the point of view of a system, this property represents the easiness of use of the system by a end user. This property of convenience can be achieved by:

   (a) **Interactivity at the right level**
   From the point of view of the cognitive vision platform, this property refers to the **separation of the different expertise**. Indeed, experts of a specific domain are the best persons to deal with their domains. For example, a biologist expert is not aware to the image processing algorithms to extract information of image. Nevertheless, they are the best persons to describe biological objects and their discriminative properties. Therefore, in addition to the modularity of the platform, the cognitive vision platform must provide tools to enable specific problem experts to contribute to their domain. The cognitive vision platform is a tool for the cooperation of different expertise.

   From the point of view of the semantic image interpretation system, this property means that domain experts are the best persons to perform the assessment of the system.

   (b) **System autonomy**
   This property refers to the ability of the system to provide to the end user sufficient basic elements to achieve his high level goal. This property is linked to the property of flexibility, adaptability and robustness previously defined.

4. **Goal directed behavior**
   This property reflects the difference between seeing (the sense) and looking (using the sense for a specific purpose). It emphasizes the dependency between action and perception. According to the purposive paradigm of computer vision [Aloimonos, 1990], vision must be considered within the high level task to accomplish. The complete image interpretation system should be aware of what it processes: visual information relevant for the high level goal.
   From the control point of view, it means the combination of top-down (model-driven) and bottom-up (data-driven) processes.
   From the knowledge point of view, it means to focus the knowledge modeling on the relevant needs of the task to accomplish.

As we can see, most of these properties are highly dependent. This list of requirements appears as a catalogue. They were the main lines for the design of the proposed cognitive vision platform. The following section will give an overview of the different steps of the design and of the resulting platform. We will insist on how the proposed platform answers to the previous requirements.

3.2 Overview of the Proposed Cognitive Vision Platform

In this section, we present a brief overview of our approach. We have defined in the previous section the expected requirements for the cognitive vision platform (**What we want? For what?**). The following section focuses on the global methodology used to answer to these requirements. The detailed description of our solution will be the topics of the two next chapters.

3.2.1 A Distributed and Problem Oriented Architecture

Semantic image interpretation refers to the task of understanding and assigning a high level meaning of the contents of an image. As mentioned before, the complex problem of semantic image interpretation can be divided into three more tractable sub-problems (fig. 3.1). For each sub-problem, the abstraction level of data is different. An illustration of the data abstraction level of the three sub-problems can be found on figure 3.2. These three sub-problems are:

1. **the image processing problem** for the extraction and the numerical description of objects of interest from images.

2. the problem of the **mapping** between the **qualitative** high level representations of physical objects in the scene (*a white fly*, fig 3.2) and the **numerical** information extracted from images (*a region with its numerical descriptors*, fig 3.2). This problem refers to the symbol grounding problem: the lack of coincidence between what is perceived by the system (image data) and the interpretation that a user associates with it. This mapping often needs an intermediate representation(*a symmetrical white surface with an elongated heart like shape*, fig 3.2). As we deal with real visual scenes which imply the management of multiple objects and of their spatial configuration, this complex problem can not be restricted to an algorithm or part of a recognition algorithm. It involves complex reasoning such as **spatial reasoning**, uncertainty reasoning to deal with the imprecision of extracted data and with the imprecision of qualitative high level representations and generic data reasonings (grouping, splitting). By the following, we will refer to this complex problem as the **visual data management problem**.

3. **The semantic interpretation problem**, i.e. the understanding of the scene using the application domain terminology. For the example of the figure 3.2, it consists in assigning to the image its biological meaning : *a white fly on the underside of a rose leaf.*

Each sub-problem is a problem as such, involving its proper expertise. To manage and separate the different sources of knowledge and reasoning, we deepen an approach developed in our team and described in [Ossola et al., 1996]. This approach proposed a distributed architecture based on the cooperation of two knowledge based systems: an image processing **program supervision module** and a **classification module**. The use of **program supervision** techniques for image processing problems enables to process images in an intelligent way, i.e. the image processing is able to adapt itself to different image contexts. Image processing program supervision techniques are good techniques to insure **adaptability**, **flexibility** of image processing systems and to insure **re-usability** of image processing programs and techniques.

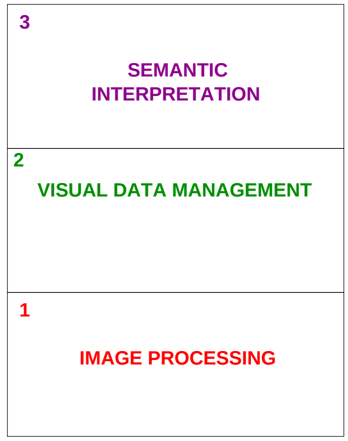


Figure 3.1: The global problem of semantic image interpretation can be divided into three more tractable sub-problems

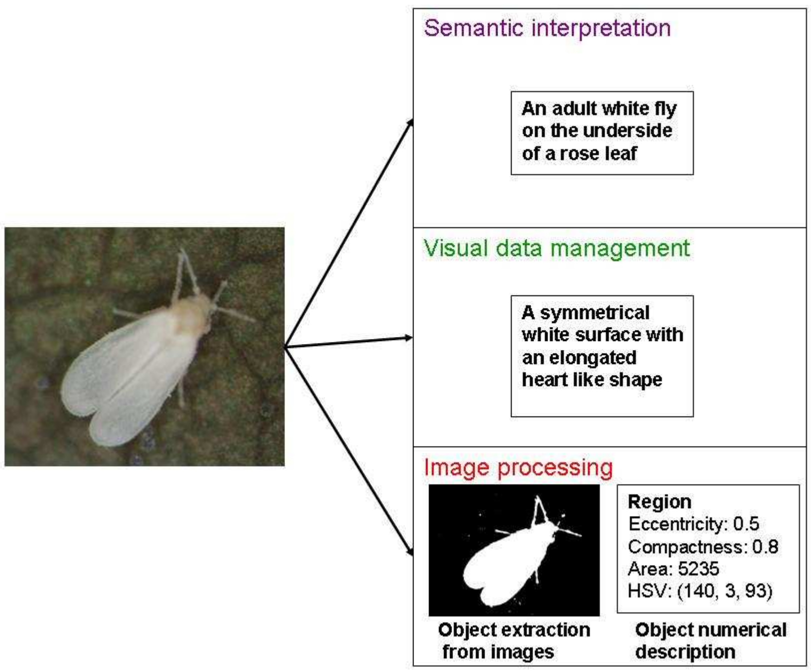


Figure 3.2: Illustration of the three abstraction levels of data corresponding to the sub-problems of semantic image interpretation. The image is a microscopic biological image

Nevertheless, contrary to this previous architecture which encapsulates the symbol grounding problem in the classification module through domain dependent data abstraction criteria, we think it is a problem as such. It requires its proper module. Moreover, this previous architecture missed a spatial reasoning component. So, we deepen the previous approach by defining a distributed architecture based on the **cooperation of three knowledge based systems**. Each KBS (Knowledge Based System) is highly specialized for the corresponding sub-problem of semantic image interpretation. The architecture is composed of an **image processing program supervision KBS**, a **visual data management KBS** specialized in symbol grounding and spatial reasoning and a **semantic data interpretation KBS**. This architecture is **problem-oriented**: the global semantic image interpretation problem is broken down into sub-problems, and each sub-problem is assigned to a particular part of the global system.

The **distribution** of the architecture into three modules reduces the complexity of the global semantic image interpretation problem. This distributed approach enables to separate the different sources of knowledge and types of reasoning involved in the global semantic image interpretation problem.
Like many other interpretation system architectures, this architecture reflects the three well known Marr's abstraction levels of computer vision [Marr, 1982] and the common admitted knowledge structuring in a hierarchy of three levels: the **low level** (image processing level), the **intermediate** level (knowledge about the mapping between image processing and domain knowledge) and the **high level** (domain knowledge). These three levels refer to the levels illustrated in the figure 3.2. Nevertheless, whereas Marr's paradigm made only the distinction between the different level data types (pixel, images primitives, symbolic data), our architecture aims at separating and formalizing the specific knowledge and the specific reasoning for each problem.

Moreover, this distributed architecture enables **modularity** which is an important software engineering property and one of the main requirements previously identified. Indeed, each module is autonomous and can be designed independently of the other modules. With such an architecture, different modules can be modified, added or deleted without altering the behavior of the rest of the system and their cooperation can be designed corresponding to the requirements of a specific application.

Blackboard systems are well known and well used to achieve a centralized communication and control between different modules. Moreover, the use of blackboard systems was successful for image interpretation systems [Sandakly and Giraudon, 1995]. Nevertheless, a weakness of blackboard systems is to not make explicit the reasoning involved and used to solve specific problems. A blackboard architecture does not enable the explicit modeling of different types of reasoning. Contrary to them, our distributed architecture enables not only to make **independent** and to separate the different types of internal data and knowledge representation but also to specify adapted reasoning strategies for each sub-problem of semantic image interpretation.

A modular architecture implies a problem of communication and information sharing between modules. To solve this problem, we have made good use of recent progress in knowledge and information sharing and reuse by the use of ontologies. In particular, we define two ontologies:

- **a visual concept ontology** which results from a parallel work led by our team. It was introduced in [Maillot et al., 2003a]. It is a terminological ontology which can be defined as a set of concepts that are commonly used by humans to describe static objects and scenes. It is shared by the **semantic interpretation module** and the **visual data management module**.

- **an image processing ontology** which is also a terminological ontology. It is defined as a set of concepts used by image processing experts to describe common image processing problems and to describe results or input data of an image processing process. It is shared by the **image processing module** and the **visual data management module**.

The ontological engineering, the complete description of these two ontologies and their goals for the cognitive vision platform are the subjects of the chapter 4.

As a consequence, our proposition for a generic architecture for semantic image interpretation systems results in the cooperation through specific ontologies of three highly specialized knowledge based systems. A global overview of this architecture can be found in fig. 3.3.

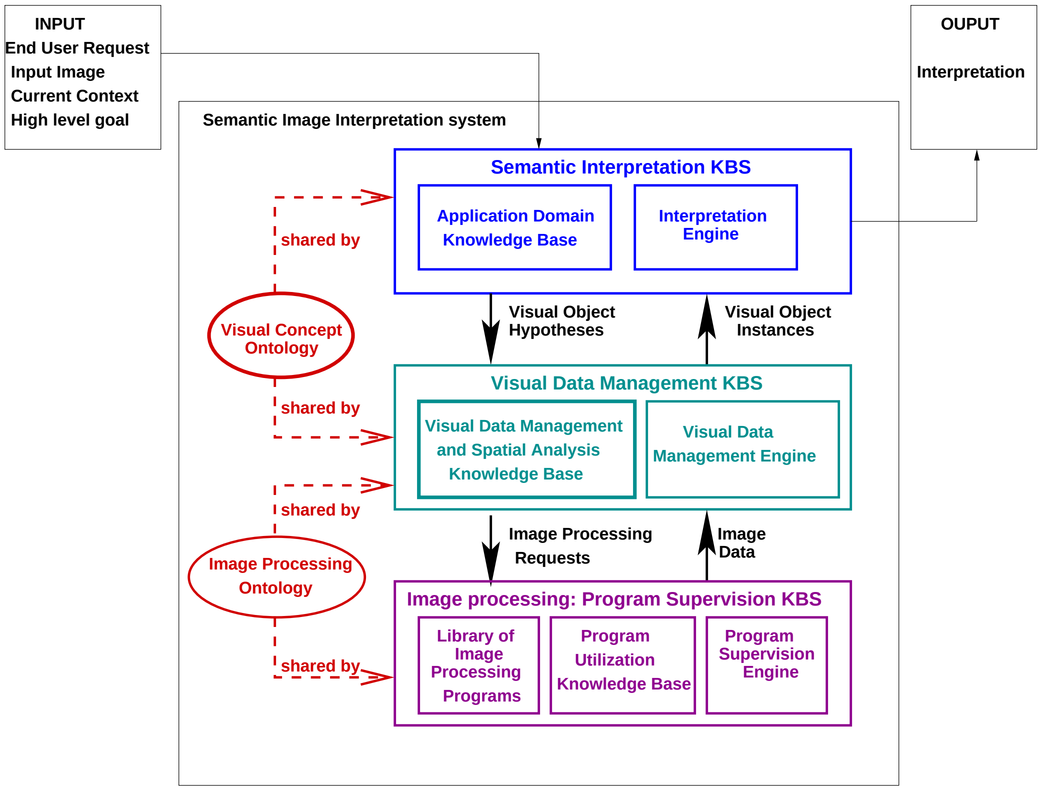


Figure 3.3: The proposed generic architecture for semantic image interpretation. A semantic image interpretation system takes as input an end user request and outputs an interpretation

3.2.2 Why the cooperation of three Knowledge Based Systems?

As already mentioned, the semantic interpretation of a visual scene is highly dependent on prior knowledge and experience of the viewer. Vision is an intensive knowledge based process. Nevertheless, knowledge based vision systems are considered by a part of the community in vision as insufficient and obsolete [Draper B., 1996]. Recent works aim at totally removing the role of the human expertise in the conception of image interpretation systems [Levner et al., 2003] (see chapter 2, section 2.3.2).

Contrary to their point of view and according to [Thonnat et al., 1998a], we are convinced that explicit expertise can help to solve complex problems of semantic image interpretation, in particular the use of an image processing library and the automation of object recognition. So that, our approach is based on knowledge based systems. Knowledge based systems are the successors of old expert systems based on inference engines and production rules for the representation of knowledge. They are artificial intelligence tools which enable to simulate the problem solving behavior of human experts. They consist of an engine, a knowledge base and a fact base. They possess some advantages including:

- The separation of knowledge and of the way this knowledge is used (the reasoning). This separation between the control knowledge and the domain knowledge enhances the **re-usability**.
- Some capitalization of expertise.
- Emulation of the strategy of experts of the corresponding task.
- Making explicit knowledge, in a manner close to the domain expertise, to make easier the **interaction with the end user and with the expert**.
- Easiness of **evolution** and **maintenance**.

However, the knowledge acquisition process and the design of knowledge bases are often time consuming. Contrary to some criticisms which have been made towards knowledge based vision [Draper B., 1996], we are convinced that knowledge based vision is not obsolete. Our point of view is that there was for many years a lack of cooperation between software engineering research and computer vision research.

To design the cognitive vision platform, we have focused our approach on recent progress in the field of reuse for knowledge and software engineering. We propose to go further in our contribution than the description of a generic semantic image interpretation system. We aim at designing a generic platform for the building of semantic image interpretation systems. The cognitive platform is a unified environment, i.e. a set of reusable tools for the building of image interpretation systems.

3.2.3 A Cognitive Vision Platform : a Unified Environment to Design Semantic Image Interpretation Systems

3.2.3.1 Re-usability through Problem Solving Methods and Ontologies

We have already mentioned that we are interested in the software engineering issues involved in the design of image interpretation systems, in particular **re-usability**, **maintenance** and **evolution**. In particular, ontologies and problem solving methods have proved to be successful methods for knowledge and software engineering [Benjamins et al., 1999].

- **Problem solving methods and knowledge based system shells**
  KBS generators are efficient solutions for **re-usability**. The notion of KBS shell (or KBS generators) emerged in the late 80's. They refer to generic Problem Solving Methods (PSMs). PSMs describe the reasoning process of a knowledge based system in an implementation and domain independent manner. It consists in abstract descriptions of the steps that must be taken to perform a particular task. In particular, knowledge based system shells have proved to make easier the development process of knowledge based systems. KBS shells allow on one hand to focus the knowledge models used by the tools on the particular needs of the task, and on the other hand to provide uniform formalisms, common to all knowledge bases belonging to the same task. Knowledge based system shells allow the design of engines and knowledge base models, independent of specific applications but dedicated to a specific problem. The principle of knowledge based system shells is shown in fig. 3.4.

  In our case, the particular problems are:

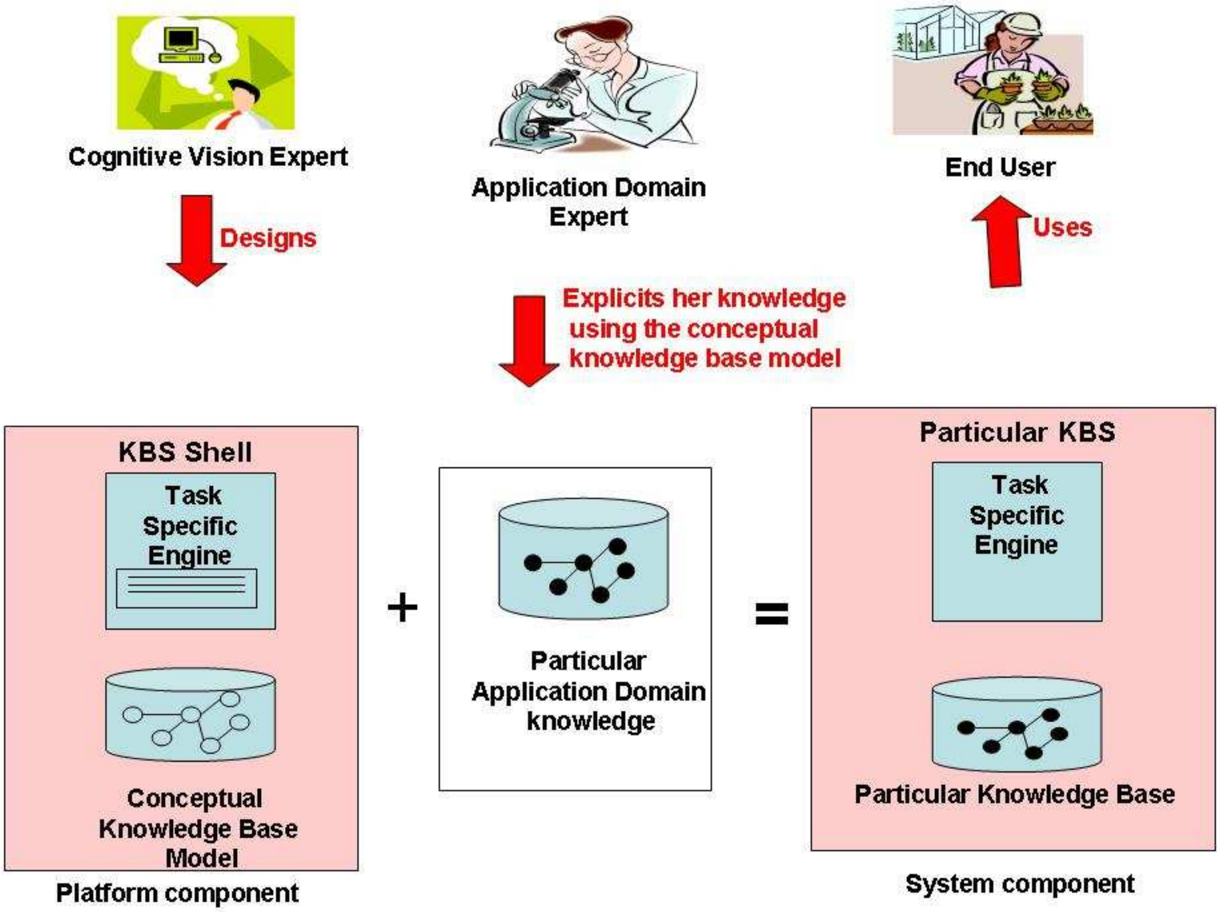


Figure 3.4: Principle of the design of an application particular knowledge based system with a specific KBS shell. Illustration with the building of an application domain knowledge based system. The role of the different persons is illustrated

- Semantic interpretation;
- Visual data management, i.e symbol grounding and spatial reasoning;
- Intelligent image processing through program supervision.

The design of such KBS shells implies to rely on models of both knowledge and reasoning mechanisms (problem solving methods) involved in the three particular sub-problems of semantic image interpretation.

- **Ontological engineering**
During the last decade, ontologies have been identified as suitable components to enable knowledge and information reuse and sharing. As problem solving methods, they are another class of **reusable** components which were investigated within the knowledge based system community for intelligent system architecture [Studer et al., 1996]. A complete definition and the benefits of ontologies can be found in the next chapter.

In our case, we have shown the benefits of the use of ontologies to manage the **interoperability** of the different platform modules. As the modularity of the proposed platform enables to reduce the complexity of the global problem and makes easier the evolution and the management of the system, it also produces a need of information (knowledge and data) sharing between the different architecture components. Ontologies can be seen as common corpus, common interchange formats to allow the communication between the different modules.

Moreover, ontologies have been proved to be efficient tools to reduce the bottleneck of the knowledge acquisition process. An ontology based domain knowledge acquisition process has been defined in our team [Maillot et al., 2003a]. Based on the domain independent visual concept ontology previously described, this acquisition process has two main advantages:

- The knowledge acquisition can be done by experts of the domain and not by a knowledge engineer. Indeed, the knowledge acquisition was traditionally carried out by a knowledge engineer, most of the time totally ignorant of the domain problem.
- As a predefined corpus used to guide the domain knowledge acquisition, it reduces the semantic gap between domain concepts and lower level concepts (in particular image concepts)

Thus, we propose a generic cognitive vision platform based on three KBS shells (generators) and on two specific ontologies : a visual concept ontology and an image processing ontology. A global overview of the platform can be found in figure 3.5. The cognitive vision platform can be used in a cooperative way by three experts, i.e. an application domain expert, a visual data management expert and an image processing expert, to build a complete semantic image interpretation system for the domain application. The methodology and the role of the different experts is depicted in figure 3.6.

3.2.4 Towards a Minimal System for Semantic Image Interpretation

One of the requirements of the cognitive vision platform is its convenience. We have shown that the modularity of the proposed platform enables to separate the different sub-problems. It enables to the different experts of the three tractable sub-problems to only be aware to the part of the platform which concerns their expertise. Moreover, we want to minimize the development cost from an application to another one. We propose to build a minimal semantic image interpretation (fig 3.7) with:

- A basic visual data management knowledge base. It was built according to the visual data management knowledge base model and it contains generic visual data management knowledge, independent of any applications. The skeleton of this minimal visual data management knowledge base is provided by the visual concept ontology and the image processing ontology. To cope with a particular application, this knowledge base can be incrementally augmented.

- A knowledge base of the utilization of very generic image processing programs (generic image processing functionalities as image segmentation, feature extraction, ...) and a library of image processing programs. This knowledge base can be incrementally augmented and specialized for a particular application with more precise image processing programs.

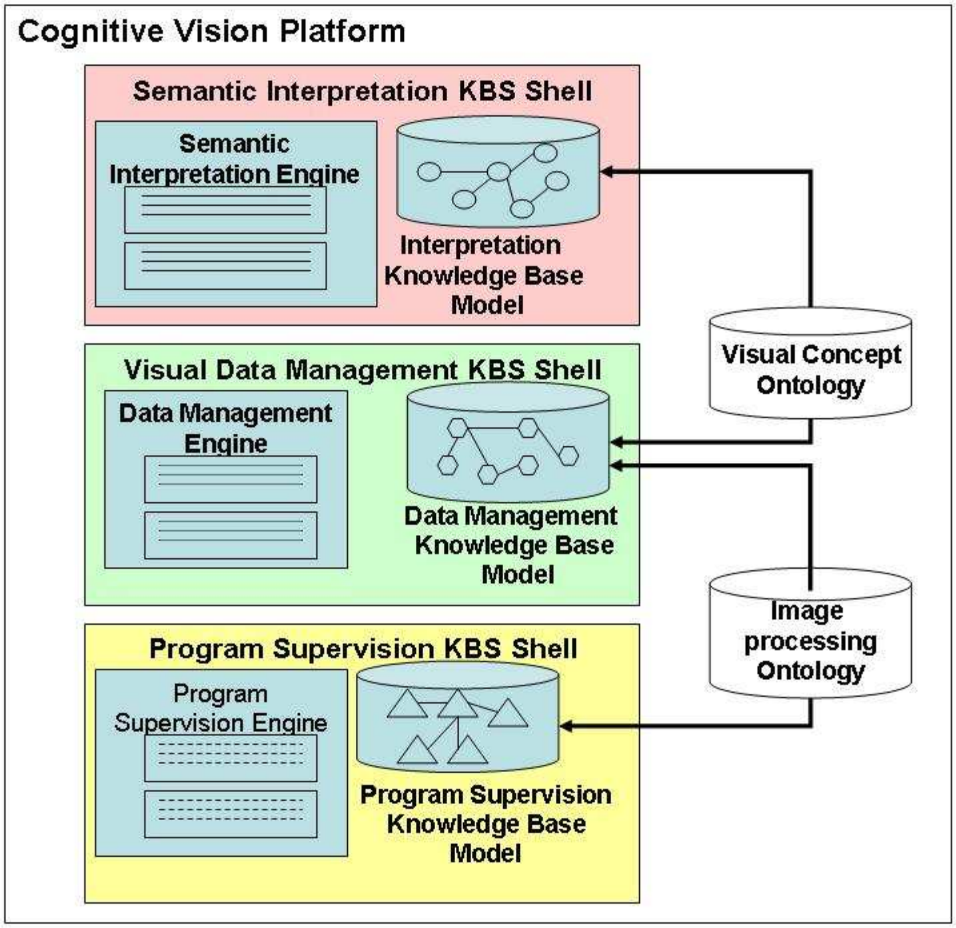


Figure 3.5: Cognitive Vision Platform global overview

3.2.5 Methodology of Use of the Minimal System for the Design of Particular Systems

The principle of use of the minimal system is described in fig.3.8. We can see that the knowledge acquisition is reduced to the particular application specific knowledge: specific domain application knowledge and, if needed, specific application visual data management knowledge (addition of specific or alternative functionalities) and specific application image processing knowledge base (addition of specific image processing functionalities).

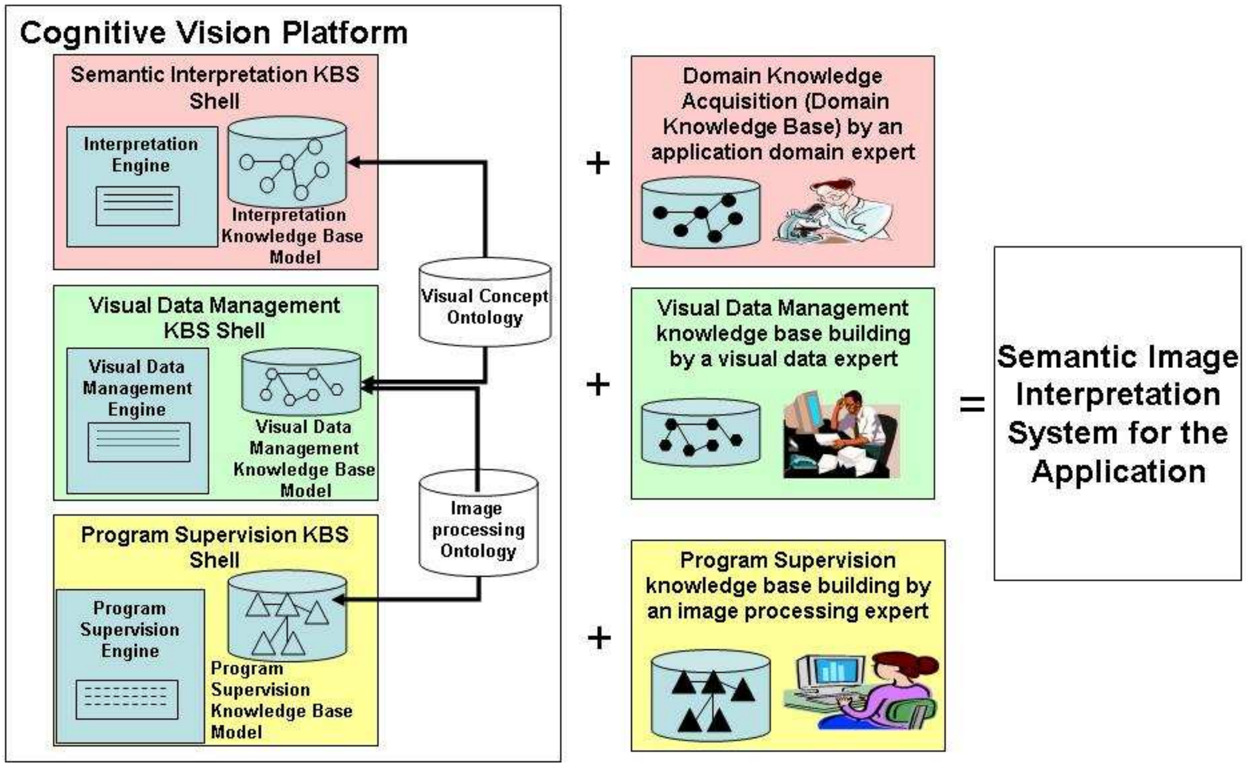


Figure 3.6: Design of semantic interpretation system for a specific application using the cognitive vision platform

3.3 Conclusion

This section has given a brief overview of what we mean by cognitive vision platform. More than yet another image interpretation system, the cognitive vision platform can be defined as a unified framework for the design of image interpretation systems. This unified framework consists of a generic engine and of a generic conceptual knowledge base model for the three sub-problems of semantic image interpretation. This framework enables **application independence** and **re-usability**.

The distributed and **modular** architecture of the proposed platform is another good point for the application independence of the platform.

We argue that the choice of knowledge based techniques are a convenient way to be close to and to emulate the human expertise. Moreover they are **convenient** to use and easy to **maintain**.

Moreover, another important point is the use of ontologies for knowledge and information sharing among the different platform modules and to reduce the bottleneck of knowledge acquisition. These ontologies and their benefits and role in the platform are described in the next chapter (chapter 4).

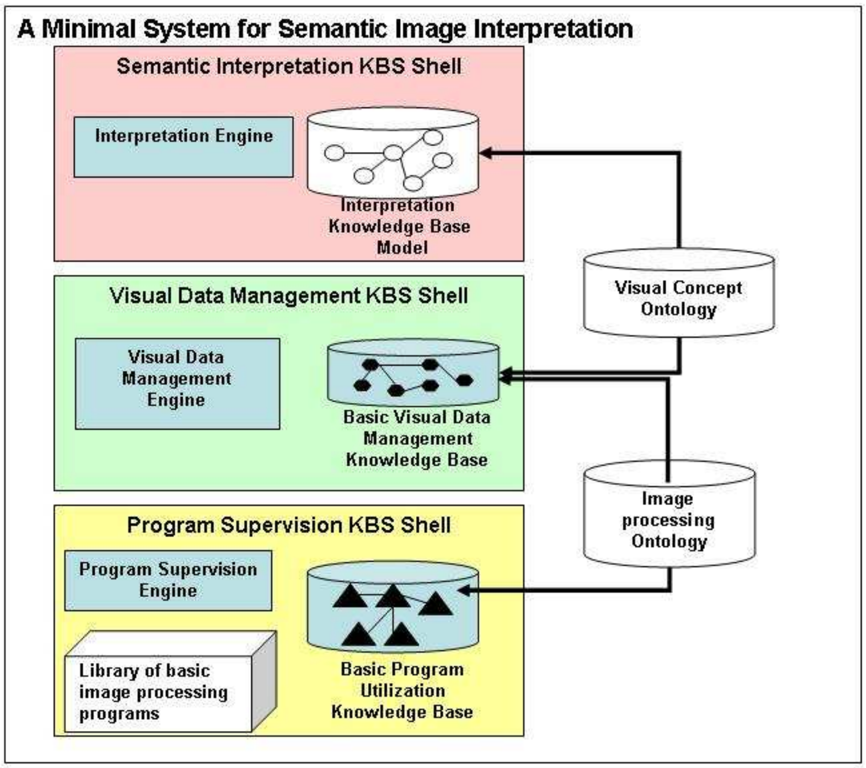


Figure 3.7: A minimal system for semantic image interpretation

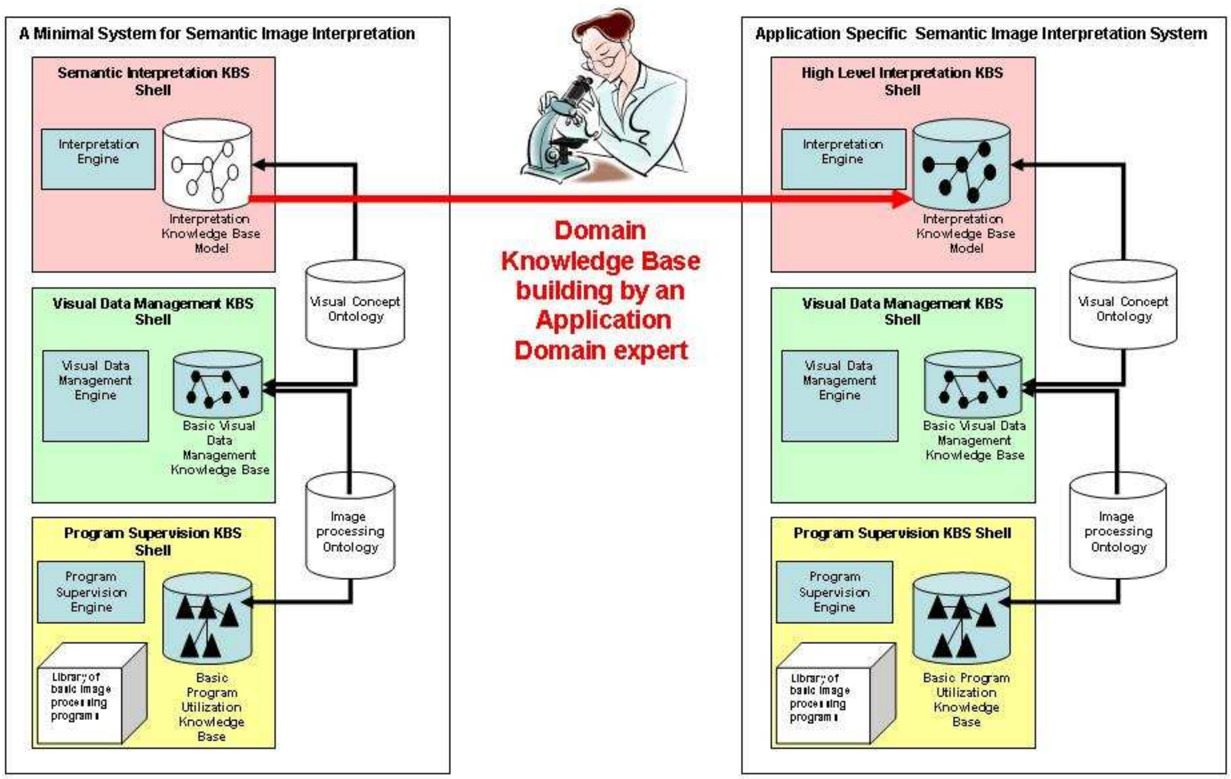


Figure 3.8: Principle of use of the minimal system

Chapter 4

Towards Ontologies for Semantic Image Interpretation

The concept of Ontology emerged in the beginning of the 1990's in the artificial intelligence and the knowledge engineering communities. The term *Ontology* was borrowed from philosophy where it means a theory about the nature of being, the study of the kinds of things that exist. Ontological engineering became an active research topic in artificial intelligence and covers a wide range of applications [Devedzić, 2002]. In this chapter, we first briefly define the notion of ontology and explain the main benefits of ontological engineering. Then, a short overview of the use of ontologies for semantic image interpretation is given. A last, we describe two ontologies, i.e. **a visual concept ontology** and **an image processing ontology** and their roles for the cognitive vision platform.

4.1 A Brief Definition of Ontology

The most referenced definition of the notion of ontology is given by Tom Gruber in [Gruber, 1993] as an *"explicit specification of a conceptualization"*. This definition is well admitted but often considered as too broad. It was refined in [Borst, 1997] as: *"Ontologies are defined as a formal specification of a shared conceptualization"*. According to [Studer et al., 1998]:

- *conceptualization* refers to an abstract model of some phenomenon in the world by identifying the relevant concepts of that phenomenon,
- *explicit* means that the identified concepts and the constraints on their use are explicitly defined,
- *formal* refers to the fact that the ontology should be machine readable,
- *shared* reflects the notion that an ontology captures consensual knowledge not private to some individuals but accepted by a group.

An ontology entails some sort of the world view with respect to a given domain. As a world view is often conceived as a set of concepts, their definitions and their relationships, an ontology is composed of a set of terms (concepts of the domain) and some specifications about their meanings (their definitions: properties, relationships).

A domain ontology is reusable in a given domain. It provides vocabulary about the concepts within the domain and their relationships, about the activities that take place in that domain, and about the theories and elementary principles governing that domain.

Building an Ontology

Methodologies for building ontologies are various. The design of an ontology is an iterative process. According to [Bachimont, 2000], the ontology development process has to be done in four distinct phases:

1. A phase of specification to state why the ontology is built and who are the end users. The specification gives the scope and the granularity of the ontology.
2. A phase of conceptualization that leads to a structured domain knowledge.
3. A phase of formalization that transforms the conceptual model into a formal model.
4. A phase of implementation that transforms the formal model into a computational model.

4.2 Motivations for Using Ontologies

Fundamentally, ontological engineering is a response to the need of communication between people, between people and systems and between systems. Ontologies promise a shared and common knowledge and understanding of some domains that can be communicated across people and computers. As well explained in [Gandon, 2002], the lack of this shared understanding leads to poor communication, to difficulties in identifying requirements (for system specifications), to limited interoperability and to limited potential of re-usability and sharing. According to [Gandon, 2002], an ontology *"provides a coherent base to build on, and a shared reference to align with, in the form of a consensual conceptual vocabulary on which one can build descriptions and communication acts "*.

In [Uschold and Grüninger, 1996], purposes and benefits of using ontologies are divided into three categories:

1. **Assistance for communication.**
Ontological engineering deals with the need of a unified framework for the communication between people, or between systems, with different needs, with different view points and with different background contexts. Any communication task is simplified by a shared lexicon. Ontologies enable to reduce conceptual and terminological confusions and misunderstandings due to different jargons and points of view by providing a shared understanding and unambiguous definition of a domain.
2. **Achievement of interoperability among computer system modules.**
Interoperability reflects the need to exchange data when different users and different software tools are involved. The aim is to build an integrated environment for different software tools. In achieving interoperability, ontologies are used as interchange formats. They are used to support translations between different modeling methods, paradigms, languages and software tools. In this case, ontological engineering aims at providing a common access to information for different systems. Indeed, information can be required by several modules but as these modules reason in their proper domain, this information is expressed in an inaccessible format. The ontology helps in rendering this information intelligible by providing a shared understanding of the terms. Benefits of this approach include interoperability and more effective use and reuse of knowledge sources.
3. **Improvements in software engineering: specification, reliability and reusability.**

- **Specification**: ontologies can assist the process of identifying requirements and defining a specification for information systems (e.g. knowledge based systems). Indeed, a shared understanding of the problem and the task can assist the specification of software systems.
- **Reliability**: a formal representation makes possible the automation of consistency checking, resulting in more reliable softwares.
- **Re-usability**: ontological engineering clarifies the structure of the knowledge of a domain. The domain ontology forms the heart of knowledge representation system for that domain. Ontology formally encodes the domain important concepts, their properties and their inter-relationships. Therefore, it could be used as a reusable or shared component in a software system. Related to the re-usability, the notion of **genericity** represents the extent to which an ontology is intended to be reused in a range of different applications.
- **Knowledge acquisition**: ontologies reduce the knowledge acquisition bottleneck. Indeed, an existing ontology can be used as the starting point and the basis for guiding knowledge acquisition for the building of knowledge based systems. As well explained in [Gandon, 2002], ontology is a powerful conceptual tool for knowledge modeling, it facilitates the construction of domain models.

4.3 Ontological Engineering and Semantic Image Interpretation

This section reviews some interesting works that use ontological engineering in different ways for semantic image interpretation.

4.3.1 Ontology-Based Content Image Retrieval

Ontologies are widely used in the research field of content based image retrieval. We distinguish two different approaches :

- The use of a domain ontology at the high level. These works refer to ontology based content annotation of images, query-based image retrieval [Town and Sinclair, 2004], [Von-Wun et al., 2002].

- The use of a visual or object ontology at the intermediate level : between low level features and domain concepts.
  A parallel Ph.D work in the Orion team is based on this approach. In [Maillot et al., 2003b], a visual concept ontology independent of the application domain is proposed. It guides the domain knowledge acquisition process by providing a set of generic visual terms close to natural language and closer to images features. The benefits of this visual ontology are twofold: (1) the reduction of the domain knowledge acquisition bottleneck and (2) the reduction of the semantic gap between domain concepts and low level features. In [Maillot et al., 2004b], a learning based process based on this visual concept ontology is proposed for the automatic recognition of isolated objects. This method is used with image indexing and retrieval purpose.

  A similar approach is proposed in [Mezaris et al., 2004]. The authors propose an **Object Ontology** which is a set of qualitative intermediate-level descriptors. This object ontology is used to enable the qualitative description of the semantic concepts the user queries for. Low level arithmetic descriptors extracted from images are automatically associated with these intermediate qualitative descriptors. The content image retrieval process is based on the comparison of the intermediate descriptor values associated with both the semantic concept and the image regions. Irrelevant regions are rejected and the remaining regions are ranked according a relevance feedback mechanism based on support vector machines.

In [Mao and Bell, 1998], a visual ontology independent of the application domain is proposed. Their aim is to propose a shared knowledge representation of image contents at a higher level than low level image features and not dependent of a application domain. Contrary to [Maillot et al., 2003b], their approach is bottom up: the visual ontology is not used to describe domain concepts but objects of interest and regions on images.

4.3.2 Towards Ontologies for the Different Abstraction Levels of Image Interpretation

An interesting approach is described in [Camara et al., 2001] from a Geographic Information Systems (GISs) perspective. In this paper, it was argued that images have an ontological description of their own, distinct and independent from the domain ontology a domain scientist uses to extract information from them. In particular, they propose an ontology divided into three interrelated components:

1. A **physical ontology** which describes the physical process of image creation including concepts like *spectral response*, *Lambertain target*, ...

2. A **structural ontology** which describes the geometric, functional and descriptive structures that can be extracted or measured in images. (*line*, *region*, *optical flow*, ...)

3. A **method ontology** consisting of a set of algorithms and data structures describing image processing techniques to transform the image from the physical level to the structural level.

The interpretation process consists in linking a domain ontology with this image ontology.

4.4 Ontologies for the Proposed Cognitive Vision Platform

As explained before, the cognitive vision platform has a highly modular structure. The distribution of the platform in three highly specialized modules, corresponding to the three main sub-problems of semantic image interpretation, enables modules to reason in their specific domain:

- for the semantic interpretation module, it corresponds to physical objects and to physical situations that can be observed by the sensor. For instance, for the biological domain, some concepts are *leaf, disease, insect, body, infection, healthy, ...*

- The visual data management module reasons in term of generic visual and spatial concepts : *color (e.g. blue, red), shape (e.g. line, rectangular, ...), spatial relations (e.g. near of, in, left of), size, ...*

- The image processing module reasons in terms of image primitives and descriptors, e.g. *edges, regions, histogram* and in term of image processing functionalities, e.g. *image segmentation, feature extraction, ...*

An example of these differences between the different domains is depicted in figure 3.2. Nevertheless, despite these differences, we have to manage information exchanges between the three modules. By information, we mean two different notions: (1) the information produced during the analysis and (2) the descriptive information a priori provided to the system. The cooperation between the different modules involves not only data sharing but also knowledge sharing.

- The first information results from an action in one of the modules: i.e. the facts or the data. As the data created by one of the modules can be set as input of another module, a shared understanding of these data by the two different modules seems essential. In our platform, the results of the program supervision module are taken as input and processed by the visual data management module. The resulting symbolic description is the data to interpret by the semantic interpretation module.

- The a priori knowledge of each module is used to guide the lower level module. This guiding consists in building a request for a specific action on the lower level module. It implies knowledge sharing. For instance, the visual data management module has knowledge of several image processing functionalities and uses this knowledge to build a request for the program supervision module.

To achieve the interoperability between the three modules of the cognitive vision platform, we propose two ontologies:

- a **visual concept ontology** for the interoperability between the semantic interpretation module and the visual data management module;

- an **image processing ontology** for the interoperability between the visual data management module and the program supervision module.

4.5 A Visual Concept Ontology

The proposed visual concept ontology is based on the works of the Orion team previously mentioned [Maillot et al., 2003b], [Maillot et al., 2004a].

4.5.1 Definition

Experts of different semantic interpretation application domains often use and share a generic visual vocabulary to describe concepts and objects of their domain. This vocabulary is generic in the sense it is not dependent of the application domain. Domain experts usually describe the appearance of objects of their domain by information about their shape, their size, their color and their textural descriptions. In the case of a scene, objects or spatial structures can be described using spatial relationships.

As in [Maillot et al., 2003b] and [Mao and Bell, 1998], the aim of the visual concept ontology is to encode generic and intuitive visual concepts used by humans to visually describe real world objects and abstract real world concepts on images. In this section, we introduce a visual concept ontology used as a guideline to describe the specific knowledge of an application domain. The proposed **visual concept ontology** is a hierarchical set of visual concepts which can be used to visually describe real world concepts. These visual concepts enable to represent qualitative properties of real world objects. According to the definition of Gruber [Gruber, 1993], the visual concept ontology is a conceptualization of the visual description of objects and scenes by humans. It encodes common sense visual description terms and therefore is close to the natural language used by an expert.

Our contribution concerning the visual concept ontology does not lie in the complete building of this ontology. We propose to adapt a previous work, which was made in our team by Nicolas Maillot, and presented in [Maillot et al., 2004a]. Firstly, we briefly present the taxonomy of visual concepts proposed in [Maillot et al., 2004a] and we focus on the changes and additions with respect to the initial ontology. Then we describe the role of the visual concept ontology in the cognitive vision platform

4.5.2 Visual Concept Ontology Overview

The visual concept ontology designed in our team is divided in three parts [Maillot et al., 2004a] [Maillot et al., 2003b]:

- **Spatio-temporal concepts**
  They provide concepts for describing objects from a spatio-temporal point of view. These concepts include geometric concepts, size concepts and spatio-temporal relations.

  As we are only interested in the semantic interpretation of static images, the part of the ontology concerning temporal concepts are not in our concerns. In the following, we will not refer to this part of the ontology.

  Moreover, spatial relations are included in the spatio-temporal concepts and they are limited to topological relations. Contrary to [Maillot et al., 2003b], we want to make a distinction between spatial concepts used to describe by qualitative properties (like the shape, the size) the appearance of real worlds objects and spatial relations used to describe their relationships, i.e. visual scenes. We propose to distinguish spatial relations from spatial concepts by a dedicated **spatial relation ontology** which is presented in the section 4.5.6.

- **Color concepts**
  This part of the ontology is based on experiments performed by the cognitive science community on the visual perception of color by humans.

- **Texture concepts**
  This part of the ontology is also based on experiments performed by the cognitive science community.

4.5.3 Spatial Concepts

This part of the ontology provides concepts to visually describe objects from a spatial point of view. It provides concepts to describe notions as the shape, the size and the location of real world objects.

In the real world, the majority of objects can be described in terms of their shape. Shape is an important property and carries with it a great deal of information which is essential when we want to recognize objects, distinguish between objects, describing objects and manipulate them. The shape of an object can be defined as the description of the properties of its boundary (boundary-based approaches) or of its interior (region based approaches). There is an intensive work about the quantitative description and representation of shapes by low level features. A good review can be found in [Zhang and Lu, 2004].

In [Meathrel, 2001], the authors discuss the importance of the shape in general and for artificial intelligence tasks in particular. They emphasize the complexity involved in the general representation of shape and propose a theory for the qualitative representation of two-dimensional shapes. In the best case, a shape can be described by its geometry and by simple nouns as illustrated in the left image of figure 4.2. The current version of the ontology proposed by [Maillot et al., 2003b] provides a set of geometrical concepts to describe object shape. The hierarchy of geometric concepts is depicted in 4.1. This part of the ontology was inspired by a work in the field of projective geometry [Furst et al., 2003]. Nevertheless, objects can be complex without a predefined geometric model as the center image and the right image of figure 4.2.

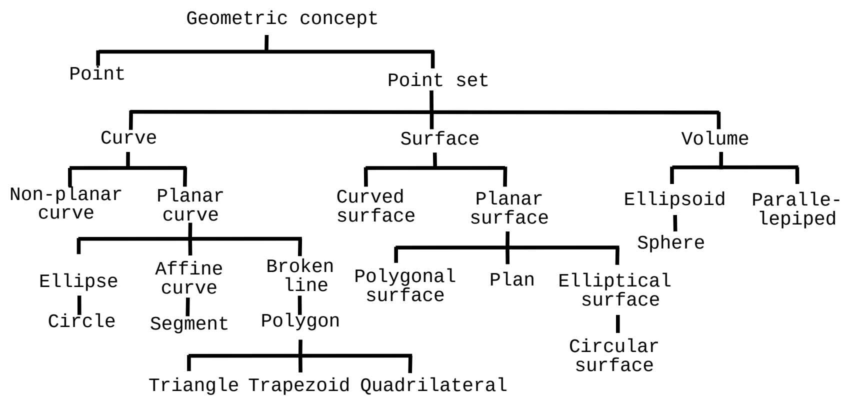


Figure 4.1: The taxonomy of geometric concepts from [Maillot et al., 2004a]

A human perception theory proposed by Biederman in [Biederman, 1987] states that objects and scenes are represented as an arrangement of simple, viewpoint-invariant volumetric primitives, such as bricks, cylinders, wedges, and cones, termed geons. This theory proposes the principle of **recognition by components** and part decomposition for recognition: a complex shape can be described by the composition of basic geometric shapes. We agree with this theory in the sense that the decomposition of objects in sub-parts is useful. For example, in the case of the two left images of the figure 4.2, it is easier to describe the body of the white fly and to consider the antenna or the legs as sub-parts. Moreover, the description of the shape of the object in the right image of the figure 4.2 is described as the structured composition of simple geometric shapes (e.g. lines). A common way to describe this kind of objects is : *a star-like network of Hyphae.* To enable the description of such objects, we add **spatial structure concepts** in the visual concept ontology.

Nevertheless, a more natural way to describe complex shapes is to use approximation or non geometric qualitative terms as for example the term *elongated heart like shape* used to describe the shape of the object in the center image of figure 4.2 instead of a geometrical decomposition of the shape.

We propose to extend the previous spatial concept ontology by a set of qualitative and more approximative concepts. To complete this spatial concept ontology, we have studied a set of experiments conducted on humans in [Socher, 1997]. They were asked to visually describe a set of objects using linguistic terms. A set of terms commonly used to describe object shapes is : *round, rounded, angular, hexagonal, rectangular, symmetric, curved, star-like, smooth, irregular, convex,...* . *Long, big, small, short, large, wide, high, thick, narrow* are commonly used to describe object size. We can extract different visual notions

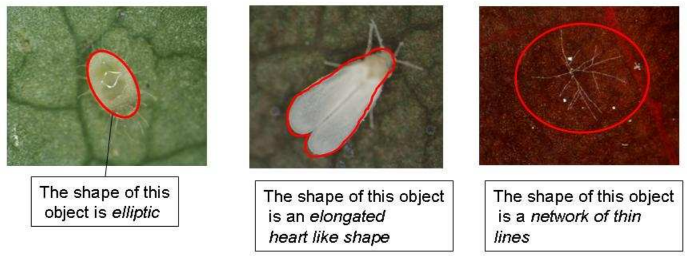


Figure 4.2: Examples of object shapes

from this set of terms. Concerning the shape, we add to the visual concept ontology:

- concepts of elongation
- concepts of convexity
- concepts of compactness
- concepts of curvature
- concepts of symmetry. Indeed, symmetry is one of the basic properties of object shapes. It is widely used to describe and discriminate objects.
- concepts of smoothness

The resulting hierarchy of shape concepts is described in figure 4.3

Concerning the size, different notions have been extracted. They are the length, the width and the height. They are common notions to describe object size. Size concepts are relative concepts. Indeed, to characterize the size of an object it is interesting to give information about its proper range of dimensions according to its self referential axis. The description of the relative size of an object is important to discriminate it from other objects. It has also an influence on the interpretation strategy of a scene: i.e. the size of objects was used in [Sandakly and Giraudon, 1995] to drive the interpretation strategy (*the biggest object first*).

Positional concepts are added to the visual concept ontology to describe position of an object in the space.

4.5.4 Color Concepts

Color is one of the main visual cues to describe the real world: objects appear to have color properties. Therefore, the representation of color information is primordial for semantic object description. Color is a perceptual phenomenon related to the spectral characteristics

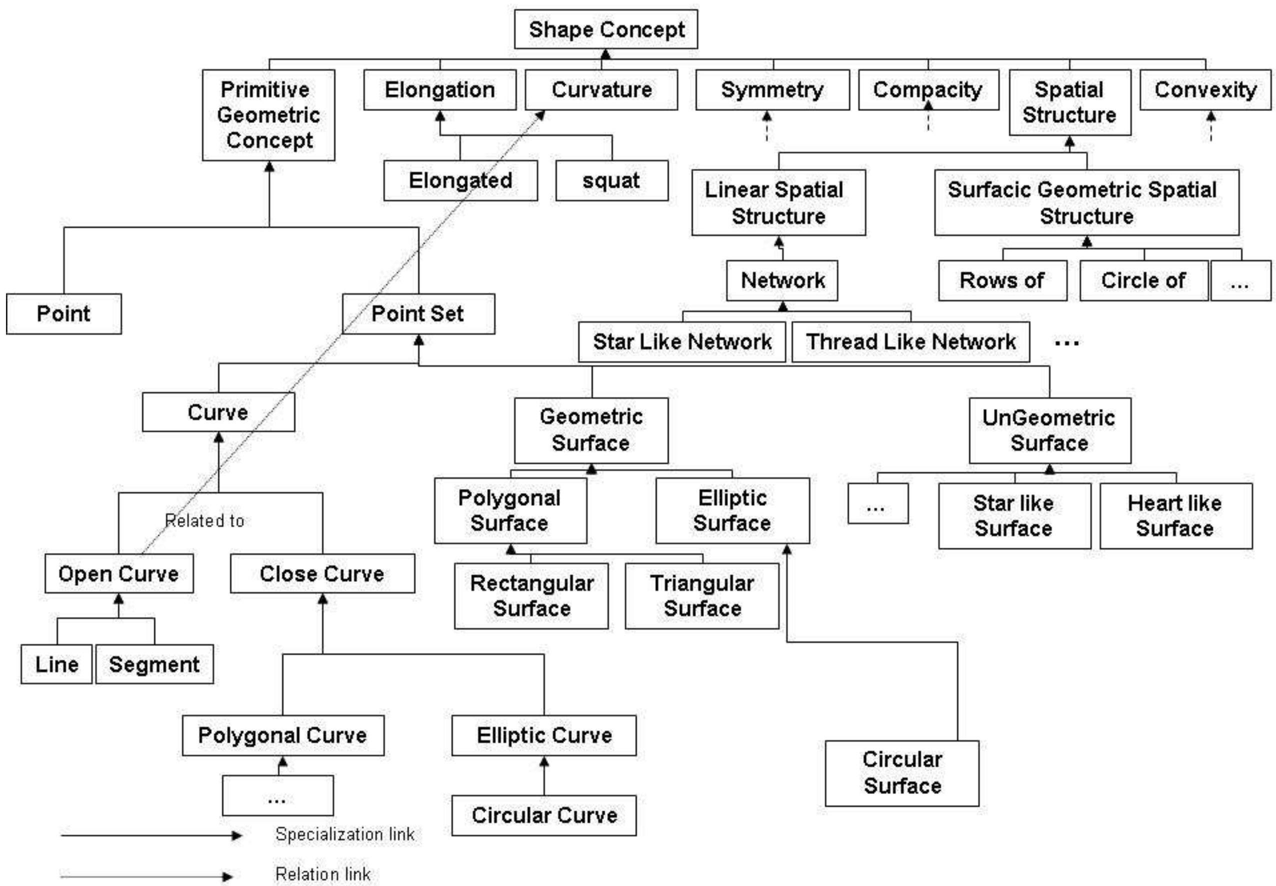


Figure 4.3: Part of the taxonomy of shape concepts in the visual concept ontology

of electromagnetic radiation that strikes the retina. A lot of experiments was performed by the cognitive science community about the color perception by humans, in particular concerning the **color naming and categorization**. **Color naming** is the process of attaching linguistic names to color patches. A good introduction to color naming can be found in [Lammens, 1994]. The Inter-Society Color Council (ISCC) and the United State Department of Commerce's National Bureau of Standards (NBS) have created a method, called the ISCC-NBS method for designating colors, as a flexible, yet comprehensive color naming system. The NBS/ISCC system is a standardized set of color terms. It defines a set of 267 color centroids. Color centroids are based on the Munsell system of color [Munsell, 1923]. The ISCC-NBS lexicon uses English terms to describe colors along the dimensions of **hue** (28 terms constructed form a basic set shown in table 4.4), **lightness** (5 terms which are *very dark, dark, medium, light, very light*), **saturation** (4 terms: *grayish, moderate, strong, vivid*) and **lightness/saturation** (3 terms: *brilliant, pale, deep*). The part of the visual concept ontology [Maillot et al., 2003b] concerning colors is based on this lexicon. It enables the description of objects from the points of view of lightness, of hue and of saturation. The association of lightness and saturation concepts into significant lightness/saturation concepts (for example the concept *Brilliant* which is the association of the concepts *Light* and *Strong*) is expressed by axioms in the ontology [Maillot et al., 2003a].

Red	Purple
Reddish Orange	Reddish Purple
Orange	Purplish Red
Orange Yellow	Purplish Pink
Yellow	Pink
Greenish Yellow	Yellowish Pink
Yellow Green	Brownish Pink
Yellowish Green	Brownish Orange
Green	Reddish Brown
Bluish Green	Brown
Greenish Blue	Yellowish Brown
Blue	Olive Brown
Purplish Blue	Olive
Violet	Olive Green

Figure 4.4: Hue concepts in the ISCC-NBS color system

In [Mojsilovic et al., 2002], a perceptually based and computational naming method for the description of color composition in images is presented. As we do, the author is interested in assigning semantics to images. She criticizes the ISCC-NBS lexicon for its lack of systematic syntax and the Munsell system for its lack of exact transform from other color spaces. A color naming vocabulary and syntax is proposed. As we are also involved in the creation of the link between image data (like the HSV values of a pixel) and high level concepts (like the name of colors), we were inspired by this work to modify the color concept ontology. The color visual concept ontology with the proposed changes is described in figure 4.5. The changes enable to describe objects without considering the chromatic information (for example when only gray level images are available).

4.5.5 Texture Concepts

This part of the visual concept ontology is inspired from results from two experiments led by the cognitive science community [Bhushan et al., 1997]. The first experiment deals with the categorization of texture words. The second one measures the strength of association between words and texture images. The resulting concept taxonomy is described in figure 4.6. We have not used this part of the ontology in our application of validation.

4.5.6 Spatial Relation Ontology

Space is an important feature of our environment and spatial perception and spatial knowledge is involved in a lot of human problem solving. In particular, people considerably use spatial relations between objects and their environment to design, detect and recognize them. There are different methods to describe the relationships between objects. In particular, we can describe them according to their topology, their orientation and their distance. Spatial relations are widely studied in different fields. In particular, interesting theories of spatial relations can be found in:

- Linguistics and cognitive science [Freksa et al., 2004], [Knauff et al., 1997], [Asher and Vieu, 1995]...

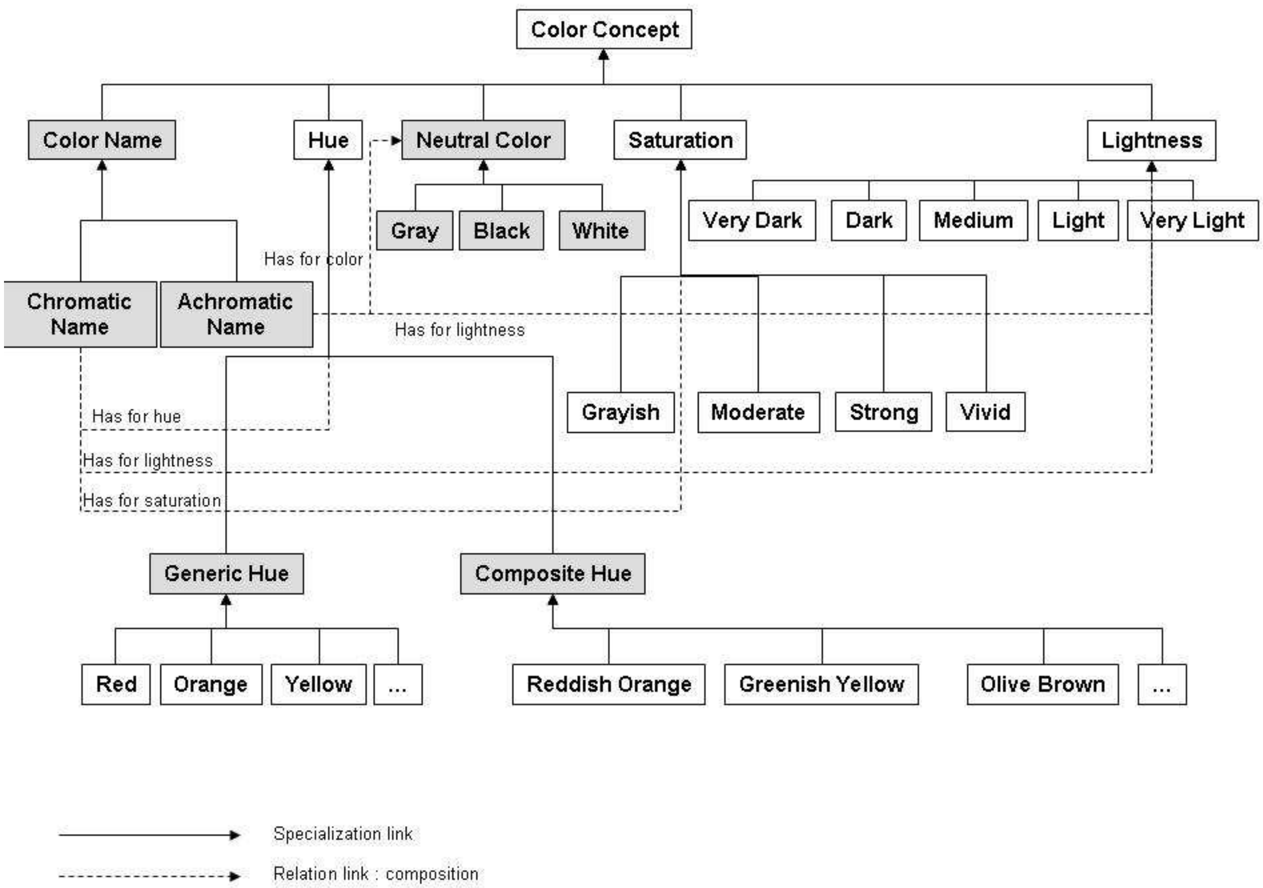


Figure 4.5: The ontology of color concepts. The additional concepts are in gray.

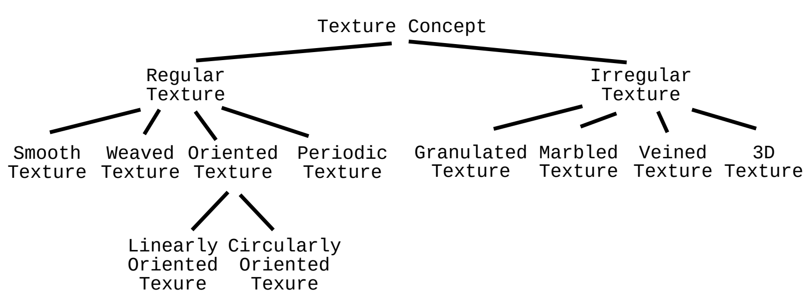


Figure 4.6: The taxonomy of texture concepts (from [Maillot et al., 2004a])

- Artificial intelligence with the works of the Leed Qualitative Spatial Reasoning group[1] [Cohn and Hazarika, 2001], [Cohn et al., 1994], with the works of Eliseo Clementini and its colleagues [Clementini et al., 1997], [E. Clementini and Hernandez, 1997], [Clementini and Felice, 1998] and with the works of Nebel [Renz and Nebel, 1999]...

[1] http://www.comp.leeds.ac.uk/qsr/

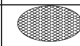
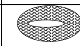
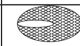
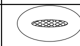
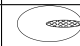
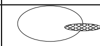
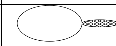
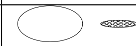
Name	**meaning**	**Icons**
$EQ(x, y)$	x is identical to y	
$NTTP(x, y)$	x is a non tangential proper part of y	
$TTP(x, y)$	x is tangential proper part of y	
$NTTP^{-1}(x, y)$	x non tangentially contains as a proper part y	
$TTP^{-1}(x, y)$	x tangentially contains as a proper part y	
$PO(x, y)$	x partially overlaps y	
$EC(x, y)$	x is externally connected with y	
$DC(x, y)$	x is disconnected from y	

Figure 4.7: The eight base relations of the RCC-8 theory

- Geographic information science with the works of Egenhofer and his colleagues [Egenhofer, 1993], [Papadias and Egenhofer, 1997], [Shariff et al., 1998]...
- Computer vision and image interpretation [Bloch and Ralescu, 2003], [Le Ber and Napoli, 2002], [Matsakis et al., 2001]...

Due to the importance of the space, it seems necessary to define a spatial relation ontology. The works mentioned above were inspirations for the definition of this ontology. The spatial relation ontology is defined as the set of concepts used to describe relations between spatial entities. As proposed in [Kuipers, 1996], the spatial relation ontology is divided into topological relations, distance relations and orientation relations. We have restricted it to 2D binary relations.

1. **Topological relations**.
   They enclose the notion of *mereotopology*, i.e. the notion of connectedness and inclusion. *Mereology* is the theory of part-hood relations: *relations of part to whole and relation to part to part within a whole* [J. T. J. Srzednicki and Czelakowski, 1984]. *Topology* refers to the notion of connectedness of objects. Mereotopology is an extension of Mereology based on the notion of connection [Clarke, 1981].
   Topological relations are the most studied in the scientific fields defined above. Indeed, cognitive empirical studies have shown that humans considerably use topological relations [Knauff et al., 1997], [Renz et al., 2000]. Topological relations are binary relations and there exist good formalizations of them in logical frameworks. In particular, the **RCC-8** (**R**egion **C**onnection **C**alculus) theory, based on the connection relation (i.e. *two objects are connected if they share at least a point*) defines eight basic topological relations [Cohn A.G, 2002]. It is the most used theory for topological relations. The name, the semantic meaning and the iconic representation of the eight basic topological relations are depicted on figure 4.7.

   To take into account specialization links between topological relations, we also take into account spatial relations of the RCC-5 theory [Clarke, 1981]. In particular, the following topological relations are part of the ontology:

   - $DR(x, y)$ means *x is discrete from y.* This relation can be specialized in **Disconnected** and **Externally Connected**.

- $PP(x, y)$ means *x is a proper part of y*. This relation can be specialized in **Tangential Proper Part** and **Non Tangential Proper Part**.
- $PP^{-1}(x, y)$ means *x contains as part y*. This relation can be specialized in **Tangentially Contains** and **Non Tangentially Contains**

The hierarchy of topological relations is shown in figure 4.9.

2. **Orientation relations**.
Orientation relations or directional relative relations describe where an object is located relatively to another one. They enable to represent an order in the space. Orientation relations are **fully metric spatial relations**. Orientation relations are not binary relations. They are established in terms of three basic concepts : **the primary object**, **the reference object** and **the frame of reference**. The **frame of reference** is the mean to represent relative locations of entities in the space. Indeed, to specify the relation of a **primary object** with respect to a **reference object**, we need to have a **frame of reference**. It exists three kinds of frames of reference : **allocentric**, **egocentric** or **intrinsic**.

   - **Allocentric or extrinsic** frame of reference refers to a fixed coordinate system imposed by external factors. This representation is independent of the position of the perceiver.
   - **Egocentric or deictic** frame of reference specifies the location and the direction of objects according to the location and the perspective of the perceiver. The orientation is given by the point of view from which the reference object is seen.
   - **Intrinsic** frame of reference lies on inherent properties of an object. These properties are used to give the orientation and to determinate the coordinate system.

   The difference between these different representations are shown in figure 4.8. These different notions are very interesting in the field of autonomous robotics or in cognitive science. In our case, the aim is to describe orientation relations on images. We make the assumptions that for orientation relations between an object of reference and its (physical) sub-parts, the frame of reference is an intrinsic one. It is defined with the axis of orientation of the object of reference. For all the other relations between an object of reference with primary objects, the frame of reference is based on the image coordinate system. We illustrate these two notions in the figure 4.8.

   Orientation relations can be described through cardinal direction (*North of*, *South of*, ...) in the context of geographic space where a reference point such as the North Pole exists. As for topological relations, there are good formalizations of them in a logical framework [Clementini et al., 1997]. Directional relations as *Left of*, *Right of* are more commonly used. Interesting works for the linguistic description of relative position on images using this directional relations can be found in [Matsakis et al., 2001], [Bloch and Ralescu, 2003]. We choose these directional relations for our spatial relation ontology. It is composed of the four primitive directional relations proposed by Freeman in [Freeman, 1975]. These four primitive relations are : 1.*Left Of*, 2.*Right of*, 3.*Above* and 4.*Below*. The directional relation *In front of* and *Behind* are also commonly used. In the case of descriptions on images, we make the assumption that they are not necessary and can be supplied by the description in terms of topological relations. Intermediate orientation relations can be described by the

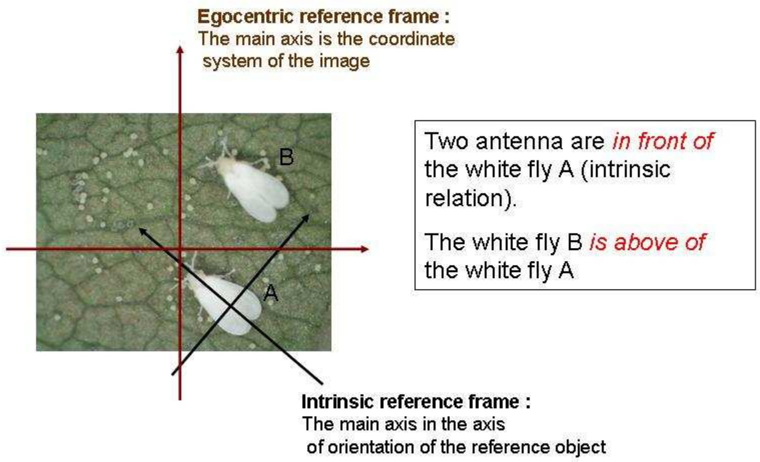


Figure 4.8: The two kinds of frames of reference to describe orientation relations on images

composition of basic orientation relations as proposed in [Bloch and Ralescu, 2003] or in [Matsakis et al., 2001].

3. **Distance relations**
   Distance relations involve distance concepts between objects. Analogously to orientation, three basic elements are needed to establish a distance relation: **the primary object**, **the reference object** and **the frame of reference**. Distance relations are highly dependent on scale factors. Indeed the relation A *Is near* B depends not only on the absolute positions of A and B but also of their relative size, of their shapes and of the reference frame. Identically to orientation relations, the frame of reference for distance relations can be:

   - **Intrinsic**: the distance is determined by an inherent characteristic of the reference object. In most of the time, the size of the reference object is used.
   - **Extrinsic**: the distance is determined by external factors like for example the spatial arrangements of spatial objects.
   - **Deictic**: the distance is determined by the point of view of the observer.

   To describe the distance between objects, two distance relations are commonly used, i.e. *Close* and *Far*. Further level of granularity can be introduced to specify distance relations. We adopt four level granularity for the proposed spatial relation ontology: *Very Close*, *Close*, *Far*, *Very Far.*

The semantic hierarchy of spatial relations is depicted in figure 4.9. We consider that this spatial relation ontology represents a basic set of spatial relations. We think this set is sufficient and ensures the full covering of all possible spatial object arrangements. All these relations are binary relations. There is also a set of n-ary spatial relations as for instance the relation *A between B and C*. The current spatial relation ontology is not complete in the sense that these n-ary relations are not taken into account. For simple relation as *A between B and C*, we argue that this relation can be represented by the combination of the two spatial relation assertions: *B is left of A and C is right of A*.

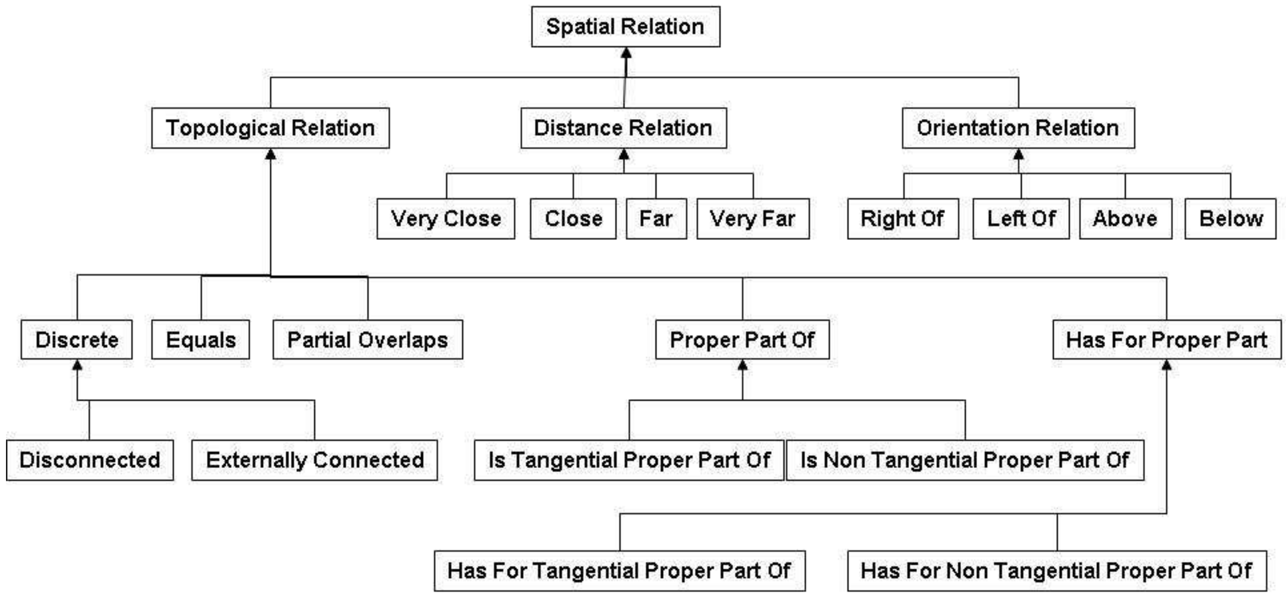


Figure 4.9: The taxonomy of binary 2D spatial relation concepts

4.5.7 Contributions of the Visual Concept Ontology for the Cognitive Vision Platform

In this section, we describe the role and the benefits of the visual concept ontology for the proposed cognitive vision platform. The main contributions of the visual concept ontology are:

1. **To make easier the application domain knowledge acquisition.**
   In the introduction, we have underlined the fact that semantics is not inside the image: domain knowledge acquisition is useful to perform the global task of semantic image interpretation. For example, in the case of the semantic interpretation of the image of the figure 4.11, knowledge of rose pests is useful. The knowledge acquisition process is a hard and highly time-consuming process. This issue is often called the

knowledge acquisition bottleneck in the knowledge engineering community. It refers to the difficulty of capturing knowledge in use in the system. For semantic image interpretation, the domain knowledge consists in the description of domain concepts. The visual concept ontology reduces the knowledge acquisition bottleneck by guiding the acquisition process. It provides to domain experts a set of predefined terms to describe their domain. The visual concept ontology (including the spatial relation ontology) provides potential terms to describe application domain knowledge. The ontology driven acquisition process is depicting in figure 4.10. This ontology driven method enables interaction with domain experts: they can build themselves the domain knowledge base. In a first step, the domain expert defines an organized and structured set of domain concepts (often a taxonomy of domain concepts). A domain concept can be composed of several domain concepts representing its sub-parts. Then the domain expert uses the visual concept ontology to describe the visual appearance of domain concepts, including their sub-parts and their spatial relationships with other domain concepts. The result of the acquisition process is a semantic knowledge base of the domain. A user friendly tool with a graphical interface was built in our team [Maillot et al., 2003a].

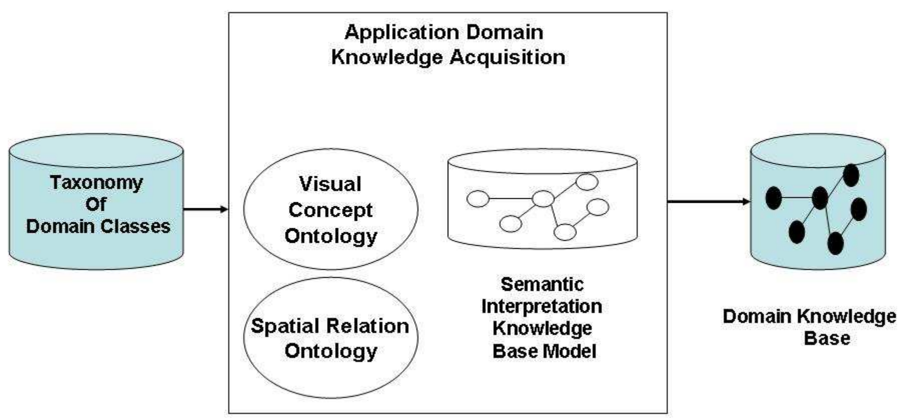


Figure 4.10: Domain knowledge acquisition with the visual concept ontology and with the spatial relation ontology

2. **To make easier the visual data management knowledge acquisition.**
   The visual concept ontology and the spatial relation ontology are skeletons to build the symbolic part of the visual data management knowledge. Indeed, they represent the types of symbolic data and the types of spatial relationships that will be processed by the visual data management system.

3. **To reduce the semantic gap between semantic concepts and low level**

features.
The use of the visual concept ontology offers the advantages of reducing the semantic gap between high level concepts and low level concepts. Indeed, concepts of the visual ontology are close to our natural language and can be well understood by end users. The visual concept ontology encodes the verbal descriptors we commonly use to describe scenes and objects. Moreover, visual concepts are also close to low level concepts. Indeed, it is more obvious to link the concept of blue with its HSV value than to link the sky with low level descriptors.

4. **To enable interoperability between the semantic interpretation system and the visual data management system.**
In our modular cognitive vision platform, each system is highly specialized for one of the sub-problems of the global semantic image interpretation problem. This modular architecture enables to separate the different types of knowledge and reasoning. Each module has its own knowledge domain and reasons in its specific domain. For example, if we take the image of the figure 4.11 :

 - The semantic interpretation module reasons in terms of semantic domain concepts: *There is a white fly on a rose leaf*;
 - The visual data management module reasons in terms of generic visual and spatial concepts: *There is a visual object composed of: (1) a symmetric elongated heart-like shaped surface; (2) two thin white lines symmetrically connected to the head of the heart-like shaped surface; (3) two thin white lines symmetrically connected to the sides of the geometric surface. The neighborhood of the object is a green textured surface.*

A communication level is needed to enable the cooperation between these two modules. The two modules have to exchange and share information to achieve the global problem of semantic image interpretation. We propose a communication process based on the visual concept ontology and on the spatial relation ontology. Indeed, they represent a corpus of common terms that are comprehensible to both modules and that provide a common access to information. The visual concept ontology provides all the necessary terms to describe domain concepts used by the semantic interpretation module and represents the symbols that will be processed by the visual data management module. Visual concepts are the symbols to ground with sensor data. The communication process consists in :

 - The building of a visual data management request from the semantic interpretation module to the visual data management module using the visual concept ontology. The visual data management request consists in a visual hypothesis represented by a structured set of hypothetical visual concepts. More details will be given in the next chapter.
 - Visual object instances created by the visual data management system are expressed according to the shared visual concept ontology. They are facts to interpret. They are processed by the semantic interpretation knowledge base system.

This ontology based communication has two main advantages. First, it enables the interoperability of two systems having their own knowledge domain. Moreover, this ontology based communication is generic. It is independent of any applications and can be reused for different applications.

Figure 4.11: Example of image to interpret. It represents a white fly on a rose leaf

4.6 An Image Processing Ontology

4.6.1 Definition

Image processing is the process of manipulating and analyzing images with a computer according to a given objective. As can be seen from the existence of reusable image processing libraries, image processing experts use and share a common vocabulary to describe their domain: i.e. the image processing terminology. First, it exists a set of generic terms to describe images or image processing results from the point of view of image processing experts. For instance, we refer to terms as region, edge, ridge, compactness, area, hue, luminosity, red value, ... They are terms currently used in the domain of image processing. Moreover, there is a set of basic image processing functionalities: e.g. image segmentation, object extraction, image feature measurements are examples of such image processing objectives. The aim of this image processing ontology is to formally encode the important concepts of image processing, their properties and their relationships.

A distinction has to be made between the program supervision knowledge model and the image processing ontology. The program supervision knowledge model represents the knowledge on how to solve a given image processing problem using a given set of programs. The image processing ontology can be seen as a set of common predefined terms used to describe an image processing problem and its results. Nevertheless, they are interrelated.

Contrary to the visual concept ontology, the image processing ontology takes part in our contributions.

4.6.2 Design of an Image Processing Ontology

4.6.2.1 Phase of Specification

Independently of the role of the image processing ontology for the cognitive vision platform, the aim of this image processing ontology is to formally encode the important concepts of image processing, their properties and their relationships. Indeed, there are a wide range of image processing applications (medical imaging, image retrieval) and a wide range of image processing program libraries. These libraries can be application dependent or application independent. The terminological analysis of these applications and of these libraries shows that a set of common concepts exists to communicate about image processing and to build image processing applications. To build the image processing ontology, the study of some works about image processing application design using existing tools [Clouard et al., 1999], [Nouvel and Dalle, 2001] were interesting. Particularly, in [Nouvel and Dalle, 2001], an interactive approach based on an image ontology is proposed to build image processing applications. The aim of the image processing ontology is to structure generic knowledge of image processing.

The impact of the image processing ontology is not limited to the cognitive vision platform but it could have a strong importance in several related domains as for example the design of an image processing library or the explicitation of an image processing problem. The image processing ontology structures the image processing knowledge.

4.6.2.2 Phase of Conceptualization

The terminological study of the image processing domain enables us to collect a set of image processing common terms representing the linguistic expression of the image processing knowledge. This set of terms is defined as an image processing lexicon. The conceptualization phase consists in organizing and structuring the different notions of the lexicon. This conceptualization phase results in a taxonomy of concepts. We differentiate two families of concepts:

- Data concepts which refer to the different types of data managed in image processing;
- Image functionality concepts which refer to the purpose of an image processing application.

The phases of **formalization** and **implementation** are not described here. They take part in the implementation of the cognitive vision platform.

4.6.3 Image processing Ontology Overview

The image processing ontology contains concepts organized in a taxonomy. An overview of the taxonomy can be seen in figure 4.12. The image processing ontology is divided into:

1. **Image Data Concepts**
   They are concepts for describing the image processing domain from the point of view of data. They are composed of:
   - A set of **Image Entity concepts** representing the different kinds of data structures that can be extracted from images. From a physical point of view, an image entity concept represents a set of image pixels. As in [Nouvel and Dalle, 2001], image entity concepts can be divided into three families:
     - *pixel* for image entities composed only by one pixel as for example the concepts of *image point*, *junction point* and *corner point*,
     - {*pixels*} for image entities composed by a set of pixels,
     - {{*pixels*}} for image entities composed by a set of set of pixels. They are structured set of pixels as for example *region graph*.

     Some image entity concepts are described in the figure 4.13.
   - A set of **image descriptor concepts** representing the different kinds of features that can be measured on images. Image descriptors are used to characterize image entities. The figure 4.14 describes some size descriptors and the figure 4.15 describes some shape descriptors
   - The relationships between **image entities** and **image descriptors**. These relations are useful because some image descriptors have no sense for particular image entities.

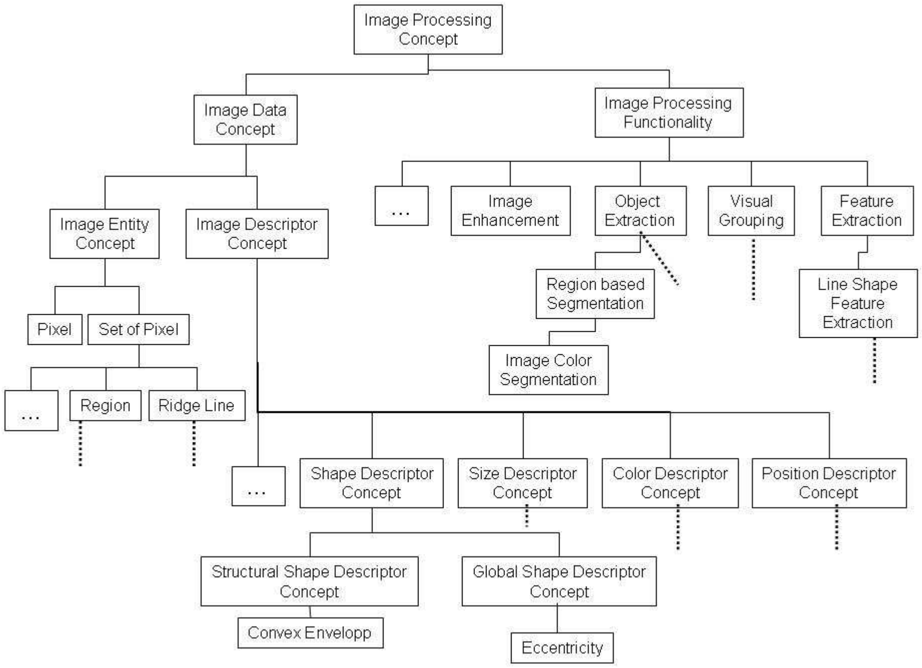


Figure 4.12: Overview of the image processing ontology

2. **Image Processing Functionality Concepts**
The considerable amount of works about the semantic integration of image processing programs [Clement and Thonnat, 1993a] has proved the goal oriented nature of the image processing problem. Actually, the use of programs, whatever is the domain, is a goal oriented problem. We argue that whatever are the motivations under the use of image processing programs, there are a set of generic image processing functionalities. These functionalities are generic in sense that they are totally independent of any applications. In [Clouard et al., 1999], three kinds of image processing tasks have been identified: intentional tasks (*what to do ?*), functional tasks (*how to do?*) and operational tasks (*by means of what?*). To formulate an image processing problem, end users have to express their request in conformity with a predefined grammar and a predefined set of terms. The aim of the image processing functionality concepts is to encode the generic high level functionalities of image processing (intentional or functional tasks in [Clouard et al., 1999]). The concepts of the ontology express the intention which is under the use of image processing programs. These functionalities refer to generic image processing functionalities as *image enhancement* or *image segmentation*. By high level functionality, we means that these functionalities are conformed to the end user point of view who is aware but not a specialist on image processing. For example, the image processing functionality *compute the Convex Hull of the main region* is too specific. For an end user non-specialist in the field of image processing, the high level functionality should have been: *compute shape descriptors*. The functionality *compute the convex hull of a region* can be seen as a specialization of the latter functionality.

This part of the ontology was defined by studying and by gathering the set of image processing functionalities hidden under standard image processing programs. Cur-

Pixel	
Image Entity	**Definition**
Valley Point	pixel for which the intensity assumes a local minimum of curvature
Ridge Point	pixel for which the intensity assumes a local maximum of curvature
{Pixel}	
Image Entity	**Definition**
Class of Pixels	Set of pixels which have common properties
Region	Set of connected pixels which have common properties
Edge	Set of connected points representing a transition on image
Curvilinear Structure	Set of connected points which assume a local extremum in the main principal curvature.
Ridge Line	Set of connected ridge points (maximum in the main principal curvature
Valley Line	Set of connected ridge points (minimum in the main principal curvature
{Pixel}	
Image Entity	**Definition**
Region Graph	Set of regions and their relations
Relative Neighborhood Graph	Set of regions and their relations
...	...

Figure 4.13: Definition of some image entity concepts of the image processing ontology

Image size Descriptor	**Definition**
Area	Number of pixels
Perimeter	Number of boundary pixels
Length (Feret dimension)	Longest straight line distance between two points within the entity
Equivalent Diameter	Size of a circle having the same area as the entity
...	...

Figure 4.14: Definition of some size descriptors

Image Shape Descriptor	Definition
Global Shape Descriptor	
Eccentricity	Ratio of the length of the maximum chord A to the maximum chord B
Compactness	How close a circle the shape is
Elongation	Ratio of the length and width of the region bounding rectangle of minimal area
...	...
Structural Shape Descriptor	
Convex Hull	Minimal convex region that entirely encompasses an image region
Medial Axis	Locus of the center of all the maximal inscribed circle of the object
...	...

Figure 4.15: Definition of some shape descriptors

rently, this part of the image processing ontology contains 5 generic image processing functionalities that can be specialized.

4.6.4 Contributions of the Image Processing Ontology for a Cognitive Vision Platform

In this section, we describe the role and the benefits of the image processing ontology for the proposed cognitive vision platform. The main contributions of the image processing ontology are:

1. **To make easier the program supervision knowledge acquisition.**
The image processing program supervision knowledge consists in the description of the use of image processing programs. A program supervision knowledge base encapsulates knowledge about the best use of programs, which may be complex for unexperienced users. The program supervision knowledge acquisition bottleneck is already reduced by an knowledge conceptual model defined in [Moisan et al., 2001] and summarized in the section 5.3.2. The process of the program supervision knowledge acquisition using this knowledge conceptual model is described in figure 4.16. The image processing ontology goes further to reduce the knowledge acquisition bottleneck by proposing a set of basic concepts to represent the input and output data (corresponding to operator arguments) and a set of generic image processing functionalities that can be used a a skeleton to build a program supervision knowledge base. The image processing ontology is used to guide the program supervision knowledge acquisition process as illustrated in figure 4.17.

2. **To make easier the visual data management knowledge acquisition.**
The image processing ontology is a skeleton to build the perceived part of the visual data management knowledge. Indeed, the image entity concepts represent the types of image data that will be processed by the visual data management system with the aim to build their symbolic descriptions. Moreover, the image functionality concepts of the ontology represent the basic sets of concepts that the visual data management should know with the plan to build program supervision requests.

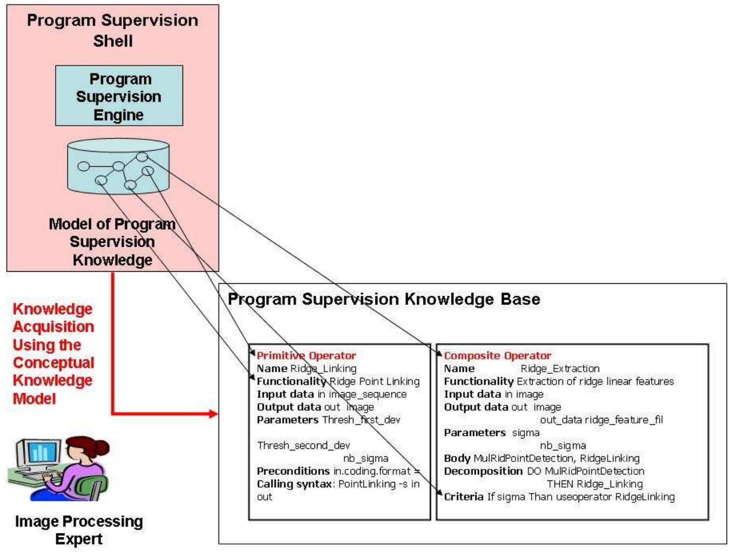


Figure 4.16: Program supervision knowledge acquisition process

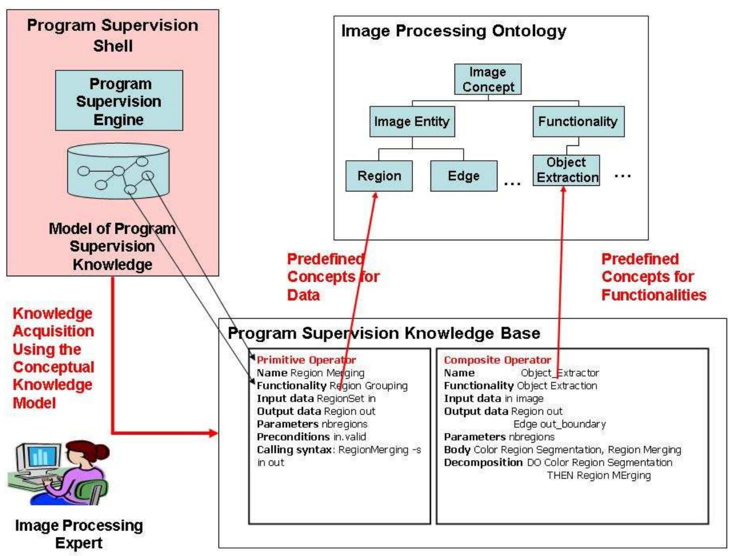


Figure 4.17: Program supervision knowledge acquisition process guided by the image processing ontology

3. **To enable interoperability between the visual data management system and the program supervision system.**
A communication level is needed between the visual data management system and the program supervision system. Indeed, on one hand the visual data management system has to ask for and has to guide the numerical data extraction by the program supervision system. On the other hand the data extracted by the program supervision system has to be easily understood by the visual data management to build their symbolic description. Due to the goal oriented nature of the program

supervision system, the visual management system has to build a request according to the predefined set of image processing functionalities provided by the image processing ontology. It needs to know the name of the functionality and its description. The image processing ontology enables this communication. Moreover the program supervision system has to produce data that will be processed by the visual data management module. Therefore, a common corpus of low level data (image entity ontology) has to be available between the two systems. To conclude, the image processing ontology enables the interoperability between the visual data management and the program supervision module in the following way:

- The building of an image processing request from the visual data management module to the program supervision module using the image processing ontology.
- The data resulting from the program supervision system are expressed according to the shared image ontology.

Chapter 5

Detailed Description of the Proposed Cognitive Vision Platform

The cognitive vision platform is composed of three independent modules. Each module deals with one of the sub-problems of the global semantic image interpretation problem. Each module has its proper modeling. As described in the chapter 3, each module of the cognitive vision platform is composed of:

- A conceptual knowledge base model;
- A dedicated engine.

In this section, we first present the detailed description of each module of the cognitive vision platform. For each sub-problem, after the analysis of the sub-problem, we propose a dedicated conceptual knowledge model and we give the algorithm of the dedicated engine. Finally, we briefly give some information on the implementation of the cognitive vision platform.

5.1 The Semantic Interpretation Framework

5.1.1 Analysis of the Semantic Interpretation Problem

The role of the semantic interpretation system is to assign a meaning to the perceived description of the scene, i.e. the data extracted from images. This meaning refers to application domain expertise and terminology. For example, for a biologist, the semantic interpretation of the image in figure 5.1 is *"Important infection of white flies in phase of insemination"*.

The semantic image interpretation problem is often limited to a classification problem: i.e. to find the class of the structured data extracted from images using predefined models. This point of view suggests that the interpretation process is a purely bottom up process. Nevertheless, although a big part of the interpretation process refers to a classification process, the semantic interpretation problem could also involve hypothesis management and a kind of planning (the strategy of interpretation depends on the high level goal).

Typically, we can illustrate the semantic interpretation problem with an example of semantic interpretation: the interpretation process of the image in figure 5.1. Let us suppose that you know that you have to interpret a biological image. According to :

Figure 5.1: A rose leaf microscopic image

- the current context of the image: i.e. a microscopic image of a greenhouse rose leaf acquired during a favorable period for pest infection,
- your knowledge about rose leaf diseases,
- your high level goal: i.e. to make a diagnosis of the health state of the leaf,

you will first make hypotheses (your expectations) about the content of the image before to analyze it. Then you will validate or reject your hypotheses by matching what you have perceived of the scene (what you see) with your expectations (what you know). At last, you refine your interpretation by a repetition of the hypothesis and test cycle to reach your high level goal. A semantic interpretation of the image in the figure 5.1 could be: *"the leaf is non healthy. There is an important infection of white flies in phase of insemination"*.

We retain the following key points which are of main importance to tackle the semantic interpretation problem:

1. **An intensive knowledge based process**
   The semantic interpretation problem is highly based on conceptual knowledge and experience about the specific world to interpret. Indeed, as long as we do not know anything about what we expect to be in images, the interpretation could be done in many ways. As a consequence, it is necessary to build models about the expected contents of images in order to be able to understand them. The use of explicit domain expertise and knowledge based techniques seems to be well adapted for this problem.

2. **Importance of the context**
   We have already mentioned that in the real world, there is an important relation between objects and their environment. Some studies in the perception psychology research field [Biederman, 1987] have shown that the human visual system considerably uses spatial relations between objects and their environment to detect and recognize them. These studies suggest that the recognition of the context should be done before object identification and recognition. They emphasize:

   - The importance of the contextual information for the interpretation process and the necessity of making explicit the context.
   - The importance of the representation of spatial relations between objects.
   - The recognition of objects by their subparts and their spatial configuration.

3. **A problem beyond single object recognition or categorization**

   Contrary to previous work in our team on image understanding [Thonnat, 2002], we make no assumptions about the content of the scene. The scene can contain an isolated object in a constrained and simple environment but also multiple objects in natural and complex environment. In [Neumann and Weiss, 2003], the high level interpretation problem is defined as the problem of *understanding a visual scene beyond single object recognition or categorization.* We agree with this statement.

   Indeed, the high level interpretation goes further than the task of finding the class of objects belonging to the scene based on a priori models. In particular, it can involve the management of multiple objects and of spatial structures of the scene. Spatial relations between objects are useful.

   High level concepts to recognize can be:

   - real physical objects of the domain (as a *white fly* for example),
   - sub-parts of physical objects (*white fly antenna*),
   - more abstract notions as for example *insemination phase of white fly.* These abstract notions are called **domain situations**. We define domain situations as known and fixed spatial configurations of domain physical objects or sub-parts of objects. They represent a set of domain physical objects constrained by spatial relations. These domain situations are related to a high level meaning. For example, in the case of our biological application, the domain situation described by the presence of circles of white fly eggs near white fly adults means *"White flies in phase of insemination"*.

   We could also mention **events** but they usually refer to the analysis of dynamic scenes and they usually include the temporal dimension. We have restricted our work to the case of 2D static visual scenes.

4. **Management of the uncertainty**
   The semantic interpretation problem has to manage two kinds of uncertainty:

   - On one hand, the perceived description of the scene extracted from images may be partial or missing and introduces uncertainty and imprecision.
   - On the other hand, the knowledge base about the expected content of the scene contains abstract descriptions which are generally qualitative and vague. It is another source of uncertainty and imprecision.

5. **A taxonomy based approach**
   From our point of view, application domain experts are the best persons to recognize objects of their domain. We propose to mimic the strategy of application domain experts by knowledge based system techniques. The aim of the semantic interpretation module is to provide tools to perform the interpretation in the same way experts do, using their usual terminology and knowledge organization system. The use of domain expertise terminology has some advantages:

   - The semantic interpretation results are expressed in terms close to natural language or to the application domain specific vocabulary, mainly qualitative.
   - The semantic interpretation results can interface with other decision processes and can be easily understood by end users.

We make the assumption that targeted applications are applications with an existing and well defined knowledge.The scope of our framework covers applications where domain experts are able to produce, possibly helped by dedicated tools, an organized conceptual knowledge base of their domains. Therefore, the domain knowledge acquisition is made by domain experts. To cope with a wide range of application domains, we choose a taxonomy based approach as knowledge organization system. Indeed, taxonomy is defined as *the science of classification according to a predetermined system, with the resulting catalog used to provide a conceptual framework for discussion, analysis, or information retrieval.* It refers to a hierarchical classification of things. We choose this knowledge organization structure because almost anything (animate objects, inanimate objects, events, scenes) can be classified according to some taxonomic scheme. It appears that human mind naturally organizes its knowledge of the world into taxonomic systems. As a consequence, taxonomy based approach seems to be the most natural and a generic way to mimic the strategies of domain experts.

5.1.2 Overview of the Proposed Semantic Interpretation Framework

With our framework, a knowledge based system performing semantic interpretation is composed of :

1. A **semantic interpretation knowledge base (SI knowledge base)**
2. A **semantic interpretation engine (SI engine)**
3. A **semantic interpretation fact base (SI fact base)**

5.1.2.1 The Semantic Interpretation Knowledge Base (SI knowledge base)

The SI knowledge base contains the domain conceptual knowledge. It is written by experts of the application domain (rose pathologists for the example of the figure 5.1). The domain knowledge acquisition process is guided by the **visual concept ontology** and **the spatial relation ontology** according to the domain knowledge acquisition process described in the chapter 4. The content of the SI knowledge base is application dependent but the way to represent and organize this knowledge is generic.
The main generic knowledge concepts to model the SI knowledge base are **domain classes**, **properties**, **domain taxonomy**, **domain context**, **acquisition context**, **context criteria** and **domain requests**.

- **Domain classes** represent the explicit description of the different objects or situations of the application domain. They are defined by a list of descriptive **properties**. **Domain classes** are implemented by frames [Minsky, 1974] and **properties** are attributes of domain classes with predefined slots. The values of the **properties** are instances of either visual concepts of the **visual concept ontology**, domain classes or spatial relations.
- Domain classes are organized in a **domain taxonomy**. It enables to better organize the knowledge and it reflects the specialization hierarchy of domain classes.
- **Domain context** represents the explicit description of the application domain context and **acquisition context** represents the explicit description of the image acquisition context. They are also implemented by frames.

- Moreover, **context criteria** are used to describe decisions during the problem solving. They represent the expertise on how to take decisions to make easier the semantic interpretation process according to the domain context, the acquisition context and the end user goal. **Context criteria** are implemented by rules.

- **Domain requests** express queries of the end user, i.e. the high level goal to achieve, the particular image to interpret, the particular domain context and the particular acquisition context.

The detailed description of this general semantic interpretation knowledge model is given in the section 5.1.3. A semantic knowledge description language called SIKL++ enables the domain knowledge description as close as the natural language as possible. This language is inspired by previous work of the Orion team [Thonnat, 2002] and its syntax is given in the annex A. It is important to note that during the reasoning process, the knowledge base is not modified.

5.1.2.2 The Semantic Interpretation Engine (SI engine)

The SI engine is application independent. It uses the domain taxonomy to build the semantic interpretation of the perceived data in the semantic interpretation fact base. The aim is to interpret the perceived data in terms of domain classes. To emulate the strategy of an expert on semantic interpretation, the semantic interpretation reasoning is modeled as an hypothesis and test cycle based on :

- The domain knowledge: i.e. the hierarchical description of possible domain classes (**domain taxonomy**).

- The current partial visual evidence which consists of the description of the perceived scene. This description is done by the lower level modules by the extraction of the information from images and by its symbolic description. In the following, the term *perceived* will refer to the visual information actually present in the images and which results from a processing in the lower level modules.

- The current domain and acquisition context.

- The high level goal of the end user.

During the hypothesis phase of the cycle, the engine has to build and propagate semantic hypotheses of what is visually expected according to the four previous points. These hypotheses provide top down guidance for the lower level problem achievements.

During the test phase of the cycle, the engine has to verify the hypotheses by the matching of the current description of the perceived scene (resulting from the lower level modules) with the domain classes. The algorithm of the SI engine is given in section 5.1.5.

5.1.2.3 The Semantic Interpretation Fact Base (SI fact base)

Once the semantic image interpretation system has been generated from the expert knowledge and from the general engine according to a methodology described in figure 3.4, the end-user (for example an horticulturist who wants to control the sanitary status of his plants) has to provide to the semantic interpretation system information about the abstract **high level goal** to achieve.

The SI fact base also contains the facts: i.e. the data to interpret. These facts represent the perceived descriptive information on the data to interpret. The SI fact base contains the perceived data resulting from the visual data management process. These perceived data are organized in two data structures : **visual objects** and **perceived scene description**. To be easily understood and to be handled by the semantic interpretation system, these perceived data are described according to the visual concept ontology and to the spatial relation ontology. The description of the perceived data is given in section 5.1.4.

5.1.2.4 Problem Formalization

We can formally define the semantic interpretation process as follows :
Given as input :

- $r = \langle go, dco, aco, im \rangle$ the **domain request** built by the end user
  - go is the **high level goal** of the end user,
  - dco is the current **domain context** explicitly represented,
  - aco is the current **acquisition context** explicitly represented,
  - im is the current **image** to be interpreted;
- τ the **domain taxonomy** and $\mathcal{C}$ the set of **context criteria**;
- $\mathcal{PSD}$ the current **perceived scene description** in the SI fact base;

it produces some visual hypotheses represented by a visual data management request ($\mathcal{VDM}$ request) and an interpretation $\mathcal{I}$ by the repetition of an hypothesize and test cycle:

- **Hypothesize**: it consists in building or completing **visual object** descriptions according to the current analyzed **domain class**
- **Test** : this step of verification consists in a **matching** procedure between the **perceived visual description** $\mathcal{PVD}$ and the current **domain class**.

The figure 5.2 presents an overview of the semantic interpretation framework.

5.1.3 Proposed Knowledge Model For Semantic Interpretation

The summary of the main knowledge concepts of a semantic interpretation knowledge base and their interrelations is depicted in figure 5.3. This section sketches the proposed semantic interpretation knowledge model. It details and gives a formalization of the generic concepts previously identified: i.e. **domain class**, **property**, **domain taxonomy**, **domain context**, **acquisition context**, **context criteria** and **domain request**.

5.1.3.1 Domain Class

Domain Class is the main knowledge entity of the semantic interpretation knowledge base. Domain classes are explicit descriptions of physical domain objects or domain situations. A **Domain Class** is defined by a list of **properties** shared by all the instances of the domain class. The representation of a domain class includes:

- **A name**: it corresponds to an application domain term.
- **A specialization link**: it represents the hierarchy of domain classes. An empty specialization link corresponds to a root domain class.

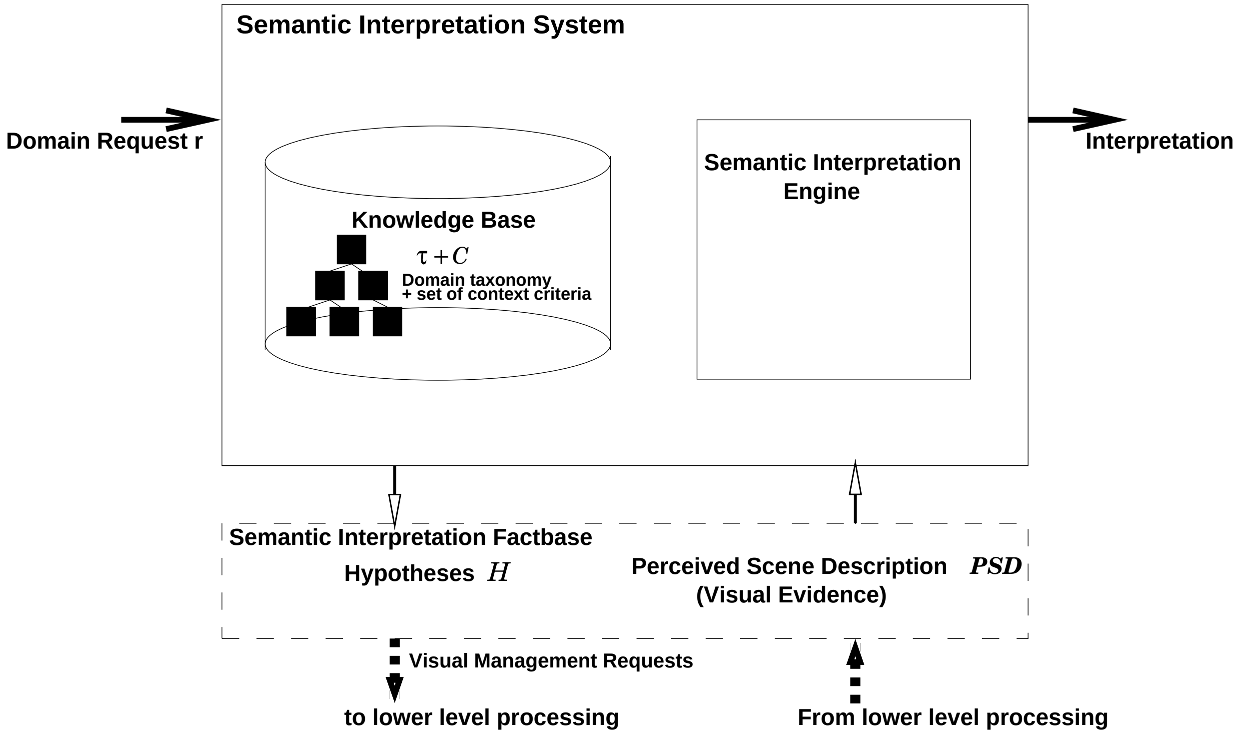


Figure 5.2: Semantic interpretation knowledge based system overview

- **A sub-part description** : it is the optional set of properties which represents the sub-parts or components of the domain class. It is represented by a list of **sub-part properties**.

- **A visual description** : it is the set of optional properties which enables to visually describe the domain class from a spatial, color and texture point of view. According to the different points of view, the visual description is divided into **spatial description**, **color description** and **texture description**. It is represented by a list of **visual properties**. This visual description is made using the **visual concept ontology**.

- **A spatial relation description**: it describes the set of optional spatial relations with other **domain classes**. It is represented by a list of **relational properties**. This description is made using the **spatial relation ontology**.

- **An importance order**: it represents the importance of the domain class in the taxonomy. The importance order is represented as a number between 0 (unimportant domain class) and 1 (highest importance domain class) in parenthesis after the name of the domain class. This importance order is used to sort out the list of domain class to process. The default value is 1.

The general syntax of a **domain class** is represented in the figure 5.4.

5.1.3.2 Properties

Properties represent descriptive attributes of **Domain Class**. They are represented by slots in the frame based formalism. The representation of a **property** includes:

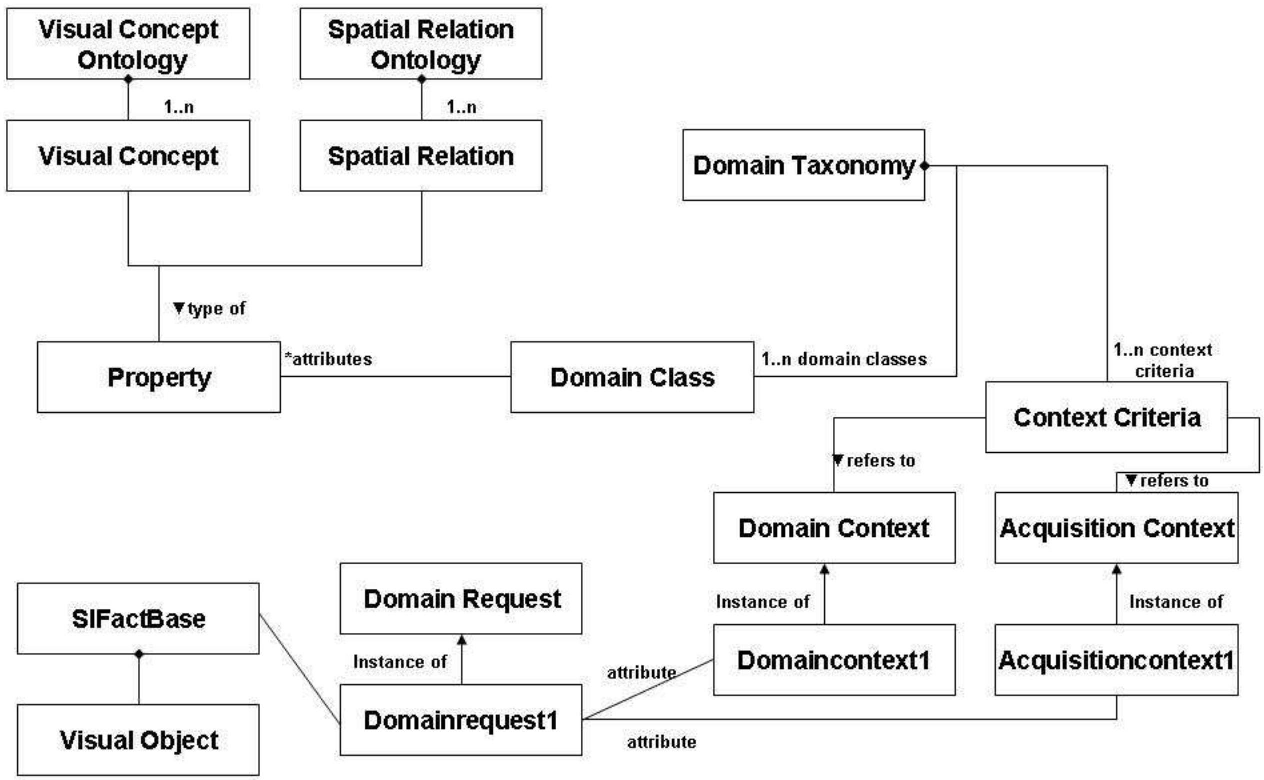


Figure 5.3: Relations between generic knowledge concepts of the semantic knowledge base

- **A type**: the value type of the property.
- **A name**: the name of the property.
- **A range**: the complete set of values that the property can assume.
- **A comment**: an informal comment on the property.
- **Facets**: a set of constraints. We introduce two kinds of facets: **at-least** and **at-most**. They are useful to represent uncertain notion as *a mycelium is composed of at least an hyphae* or spatial structures.
- **A weight**: the importance of the property represented as a number between 0 and 1. The default value of the weight is set to 1.

There are three kinds of properties:

- **sub-part properties**: they represent sub-parts of the domain class. They are instances of **domain classes**.
- **visual properties**: they represent the visual description of a domain class. According to the ontology guided domain knowledge acquisition, they are instances of **visual concepts** provided by the **visual concept ontology**.

DomainClass {
  name *a term of the application domain* **(1)**
  comment *"I note here with my own words what is important for this domain class"*
  SuperClass *the name of the parent domain class*
  SubPart Description
    An optional list of ***sub-part properties***
  Visual Description
    Spatial Description
      An optional list of ***visual properties***
    Color Description
      An optional list of ***visual properties***
    Texture Description
      An optional list of ***visual properties***
  Spatial Relation Description
    An optional list of ***relational properties*** }

Figure 5.4: The general syntax of a domain class. The syntax imposed by the model is represented in bold face.

- **spatial relationship properties** : they represent the property of having a spatial relation with another domain class. Their representations are a particular case. As explained in [Le Ber and Napoli, 2002], the spatial relation properties are instances of spatial relations and their range are domain classes. An example is given in the figure 5.5 with the property *hyphae_properpart_relation*.

Properties are always defined in a **domain class**. Their general syntax is :

Type name property_name
  comment *"an informal comment on the property"*
  range [*a set of values*]

The figure 5.5 represents an example of a **domain class** in the rose pathological domain and its associated properties.

5.1.3.3 Context Criteria

Various **context criteria**, implemented by rules, play a role in the semantic interpretation solving problem. They represent inferential knowledge.

1. **Initialization interpretation criteria** contain information on how to initialize the semantic interpretation solving problem. According to the domain context and the acquisition context, initialization interpretation criteria enable either the initialization of some characteristics of domain classes (e.g. figure 5.6) or, in particular cases, the initialization of data in the semantic fact base (e.g. figure 5.7). They are mainly used to set up the value of the importance order of **domain classes**.

DomainClass {
name HYPHAE
comment *"A thread-like, tubular filamentous fungal structure"*
SuperClass FUNGISYMPTOM
Visual Description
Spatial Description
Geometry **name** hyphae_geometry
range [*Curve Line Segment*]
Thickness **name** hyphae_thickness
range [*Very_Thin Thin*]
Straightness **name** hyphae_straightness
range [*Almost_Straight*]
Color Description
Neutral_Color **name** hyphae_color
range [*White Gray*]
Lightness **name** hyphae_lightness
range [*Very_Light Light*]
Spatial Relation Description
ProperPartOf **name** hyphae_properpart_relation
range [LEAF]
}

Figure 5.5: Representation of the domain class HYPHAE

Initialization Interpretation Criteria
Rule {
name *the name of the rule*
comment *"I note here with my own words the meaning of the criteria"*
LinkedDomainClass *the name of the domain class linked to the criteria*
Let *context a* ***Domain context*** *(or* ***Acquisition context****)*
If *context attribute a has value v*
Then *set domain class importance order to value v1*
}

Figure 5.6: The general syntax of an initialization interpretation criteria that makes decisions about the importance order of domain classes.

2. **Post interpretation criteria** contain information to refine results of the interpretation according to the domain context. They are applied after the semantic interpretation process. They generate an interpretation report (e.g. figure 5.8) .

Initialization Interpretation Criteria
Rule {
  name *the name of the rule*
  comment *"I note here with my own words the meaning of the criteria"*
  LinkedDomainClass *the name of the domain class linked to the criteria*
  Let *context a* ***Domain context*** *(or* ***Acquisition context****) and object a visual object*
  If *context attribute a has value v*
  Then *set object attribute a1 to value v1*
}

Figure 5.7: The general syntax of an initialization interpretation criteria

All the context criteria are linked to domain classes. The external form of all kinds of context criteria is :

Let *declarations*
If *premise*
Then *action*

- *declarations* declare typed free variables used in premise or in the action. They refer to objects in the fact base. Their types are either domain context or acquisition context or a visual object in the fact base.
- *premise* represents a condition to be fulfilled for given actions to take place. It checks some properties of free variables or global variables in declarations. It corresponds to a typical situation with respect to the domain expertise.
- *actions* are decisions guiding the interpretation process in response to the stated premises.

5.1.3.4 Domain Taxonomy

Domain taxonomy represents the tree reflecting the specialization hierarchy of the set of **domain classes**. The representation of the domain taxonomy includes :

- A name;
- The root of the taxonomy;
- The list of domain classes that composes the taxonomy;
- The list of context criteria linked to the domain classes of the taxonomy.

The general syntax of a domain taxonomy is given in figure 5.9

Post Interpretation Criteria
    Rule {
        name *the name of the rule*
        comment *"I note here with my own words the meaning of the criteria"*
        LinkedDomainClass *the name of the domain class linked to the criteria*
        Let *context a* ***Domain context*** *(or* ***Acquisition context****)*
        If true and *context attribute a has value v*
        Then send *"interpretation report"*
    }

Figure 5.8: The general syntax of a post interpretation criteria

DomainTaxonomy {
    name *a string*
    Root *a* ***domain class***
    Domain_Class_List *a list of* ***domain class***
    Context_Criteria_List *a list of* ***context criteria***
    }

Figure 5.9: The general syntax of a domain taxonomy

5.1.3.5 Domain Context

Domain context corresponds to the explicit description of the context of the application domain. The domain context represents all the additional, non visual, declarative knowledge which influences the semantic interpretation problem solving. Domain context is implemented by frames. The domain context description in the semantic knowledge base is made by domain experts with respect to the syntax given in figure 5.10. Instances of domain context are written by the end user and stored in the semantic fact base. The syntax of domain context instances is also given in figure 5.10.

5.1.3.6 Acquisition Context

Acquisition context corresponds to the explicit description of the knowledge of the image acquisition. This knowledge can influence the semantic interpretation process. The importance of the explicit representation of this kind of knowledge was emphasized in [Sandakly and Giraudon, 1995]. The acquisition context contains:

- information on the sensor : its type, its use mode, its magnification, its pass band.

- information on the image acquired with the sensor : the image resolution.

DomainContex {
  name *DomainContext_name*
  Attributes
    A list of attributes
  }

DomainContextInstance {
  DomainContext_name **name** *the name of the instance*
  Attributes
    A list of attributes with their assigned values
  }

Figure 5.10: The general syntax of a domain context

In the same way than domain context, instances of acquisition context are written by the end user and stored in the fact base. Information on the syntax is depicted in figure 5.11.

5.1.3.7 Domain Request

Domain Requests are queries of semantic interpretation on particular data. Their representation is composed of:

- **An input image;**
- **A domain context;**
- **An acquisition context;**
- **A targeted domain class**: the end user can focus the semantic interpretation process to a specific part of the domain taxonomy by giving the name of the **Domain class** to start the interpretation;
- **A high level goal.**
  The high level goal refers to the objective of the end user. We identify three kinds of high level goals :
  - The detection of the presence of a precise object in the scene and its identification (**Single Detection**);
  - The detection of all the occurrences of a precise object in the scene and their identification (**Multiple Detection**);
  - The detection and identification of all the objects which are present in the scene (**Scene Analysis**)

Domain requests are the mean for the end user to describe the initial problem of semantic image interpretation. The aim of the global semantic interpretation system is to respond to this request. The typical representation of a domain request and of one of its instances are represented on figure 5.12.

```
AcquisitionContex {
    name GeneralAcquisitionContext
    Attributes
        Symbol name Sensor_type
                default
                range [a list of sensors]
        Float name Sensor_magnification
                default
                range [a list of magnification values]
        Symbol name Sensor_use_mode
                default passive
                range [passive active]
        Float name Sensor_pass_band
                default
                range [the interval of light ray]
        Float name Image_resolution
                range [a list of resolution values]
    }

AcquisitionContextInstance {
    GeneralAcquisitionContext name context1
    Attributes
        Sensor_type:= ...
        Sensor_magnification:= ...
        Sensor_use_mode:= ...
        Sensor_pass_band:= ...
        Image_resolution:= ...
    }
```

Figure 5.11: The general syntax of an acquisition context

5.1.3.8 Visual Data Management Request

Visual data management requests are hypotheses of visual objects. They are built by the semantic interpretation system and sent to the visual data management system. A visual data management request contains:

- The description of the hypothesized visual object to process. It is built by the semantic interpretation system using a domain class description;
- The mode of the process. It refers to the high level goal. This mode is represented by the number of visual objects to find. It is either a fixed number or unknown.

Imprecise and uncertain knowledge is represented using results from possibility theory and fuzzy set theory. All visual concepts are imprecise by nature. Their imprecision is managed by the visual data management module. An imprecise fact is characterized

Domain Request {
name *the name of the request*
comment *"an informal comment on the domain request"*
Attributes
Image name input_image
DomainContext name domain_context
AcquisitionContext name acquisition_context
Symbol name high_level_goal
range [SingleDetection MultipleDetection CompleteSceneUnderstanding]
DomainClass name targeted_domain_class
}

Domain Request {
name *Powdery_Mildew_Request*
comment *"Is there an infection of powdery mildew and in what stage of development ?"*
Attributes
input_image:= image1
domain_context:= dcontext1
acquisition_context:= acontext1
high_level_goal:= SingleDetection
targeted_domain_class:= FUNGI
}

Figure 5.12: The general syntax of domain request

by a possibility distribution. A uncertain fact is characterized by a confidence factor (a possibility measure) and a doubt factor.

5.1.3.9 Formal definitions

According to the knowledge concepts, we give the following definitions. They are useful to describe the algorithm of the semantic interpretation engine.

- **Definition 1** Let $\theta = \{C_i/i \in 1..n\}$ a set of **visual concepts**.
$\preceq_\theta$ is a partial order between visual concepts. $\forall (C_i, C_j) \in \theta^2, C_i \preceq_\theta C_j$ means that C_i is a sub-concept of C_j
$\langle \theta, \preceq_\theta \rangle$ represents the **Visual Concept Ontology** as a hierarchical structured set of terms to describe real world concepts on images. More precisely, it represents the **visual concept taxonomy**.

- **Definition 2** Let $\mathcal{R}el = \{R_j/j \in 1..p\}$ a set of **spatial relations**.
$\preceq_\mathcal{R}$ is a partial order between spatial relations. $\langle \mathcal{R}, \preceq_\mathcal{R} \rangle$ represents the **Spatial Relation Ontology** as a hierarchical structured set of spatial relations to describe the spatial configurations of physical objects in the scene.

- **Definition 3** Let $\phi = \{\alpha_k / k \in 1..m\}$ a set of **domain classes**.
  $\preceq_\phi$ is a partial order between domain classes
  $\tau = \langle \phi, \preceq_\phi \rangle$ the **Domain Class Taxonomy**.
  $\mathcal{A} \in \theta$ is the set of instances of visual concepts used to describe visual properties.
  $\mathcal{S} \in \phi$ is the set of instances of domain class used to describe sub-part properties.
  $\mathcal{R} \in \mathcal{R}el$ is the set of instances of spatial relation used to describe relational properties.
  For a **Domain Class** $\alpha \in \phi, \alpha = (\mathcal{A}_\alpha, \mathcal{S}_\alpha, \mathcal{R}_\alpha)$ we call :
  - $\mathcal{A}_\alpha \subseteq \mathcal{A}$ the visual description of α
  - $\mathcal{S}_\alpha \subseteq \phi$ the sub-part description of α
  - $\mathcal{R}_\alpha \subseteq \mathcal{R}$ the spatial relational description of α
- **Definition 4** Let $a \in \mathcal{A}_\alpha$, $s \in \mathcal{S}_\alpha$ and $r \in \mathcal{R}_\alpha$ be respectively a visual property, a sub-part property and a relational property of $\alpha \in \phi$.
  We define $Dom : \mathcal{A}_\alpha \rightarrow \theta$ so that $Dom(a)$ is the range of a, i.e. the set of possible values of a.
  We define $Dom : \mathcal{S}_\alpha \rightarrow \phi$ so that $Dom(s)$ is the range of s, i.e. the set of possible values of s.
  We define $Dom : \mathcal{R}_\alpha \rightarrow \phi$ so that $Dom(r)$ is the range of r, i.e. the set of possible values of r.

- **Definition 5** $\mathcal{C} = \{cr_k\}$ is a set of context criteria.

5.1.4 The Semantic Interpretation Fact Base

The semantic interpretation fact base contains interpretation facts. These interpretation facts are either the initial semantic interpretation problem description or the data to interpret. The initial semantic interpretation problem is described by a particular domain request with the corresponding instances of input image, domain context and acquisition context. Moreover, the semantic interpretation fact base contains the data to interpret. These data are structured in **visual objects** and **perceived scene description**.

5.1.4.1 Visual Objects

According to the state of the interpretation, **visual objects** are either hypotheses of a domain class (which have to be processed by the visual data management system) or symbolic description of perceived data on images (which have been processed by the visual data management system). As visual objects are processed by the visual data management system, they are shared with the data management system. THere are three kinds of visual objects: **(1) primitive visual object**, **(2) composite visual object** and **(3) visual scene object**.

1. **Primitive visual object**
   A primitive visual object is composed of:
   - A state: it defines the state of the object in the global semantic interpretation process. The state can be:
     - *hypothesized* (hyp): the visual object is a visual hypothesis of a semantic domain class. It has not yet been processed by the lower level modules. The link with corresponding data on images does not exist.

  - *missing* (miss): the hypothesized visual object has been processed by the lower level modules but no data in images correspond to the hypothesis made by the semantic interpretation module.
  - *perceived* (peir): the hypothesized visual object has been processed by the lower level modules. It is associated with data on images. In robotics, we will say that is is *anchored* with image data. Perceived visual objects are inputs of the semantic interpretation module.
  - *partially recognized* (partial): the perceived visual object has been processed by the semantic interpretation module. A partial interpretation of this object has been made. It is partial in the sense that it is not perfectly recognized. Some attributes can be missing due to occlusions or bad object extraction for example.
  - *recognized* (rec): a semantic interpretation is associated with the visual object. The visual object has been recognized as a possible instance of a domain class.
- A set of visual attributes which corresponds to the symbolic description of the visual object. A visual attribute has a state which is either *hypothesized* or *perceived*. A visual attribute is defined by :
  - A name;
  - A type (Visual Concept);
  - A state;
  - Optional perceived values (completed by the visual data management process);
  - An expected range of values (completed by the semantic interpretation hypothesis phase);
- A link with the associated (or anchored) image data in image;
- A list of associated domain classes which are possible semantic interpretations for the visual object and their compatibility and incompatibility values.

Visual objects are automatically built by the semantic image interpretation system and processed and completed by the visual data management system. They are not pieces of knowledge written by experts but facts. Nevertheless, they could also be hand coded (to test the semantic interpretation engine for example) according to a syntax described in figure 5.13. This figure enables to summarize and clarify the notion of visual object.

2. **Composite Visual Object**

A composite visual object is the type of visual object generated by spatial structures. A composite visual object is a visual object composed of a structured and spatially constrained set of identical primitive visual objects. It corresponds to the spatial repartition of a primitive visual object and a spatial relation. Examples of composite visual objects are network of connected lines (mycelium), circle of neighborhood circular surfaces (white fly eggs), rows of neighborhood rectangular surfaces (rows of building,...).

A **Composite Visual Object**(figure 5.14) is represented by:

- A state;
- The description of the structural primitive visual object;

PrimitiveVisualObject {
  name *a string*
  state [*hyp miss peirc partial rec*]
  Visual Attributes
    VisualConcept name *attribute1*
      expectedvalues *List of VisualConcept Instances*
      perceivedvalues *empty or names of the recognized visual concepts with confidence degree*
    ...
  ImageData *empty or instance of image data*
  AssociatedDomainClasses *empty or weighted list of domain class names*
  }

Figure 5.13: General description of a primitive visual object

- The spatial relation which links the set of primitive visual objects;
- The visual description of the complete spatial structure (e.g. list of attributes which describe the complete structure);
- A link to the corresponding perceived image data;
- The list of associated domain classes which are possible semantic interpretation for the visual object and their compatibility and incompatibility values.

CompositeVisualObject {
  name *a string*
  state [*hyp miss peirc partial rec*]
  structuralobjectdescription *A primitive visual object*
  spatialrelation *A spatial relation*
  Visual Attributes
    VisualConcept name attribute1
      expectedvalues *List of Visual Concept instances*
      perceivedvalues *empty or names of the recognized visual concepts with confidence degree*
    ...
  ImageData *empty or instance of image data*
  AssociatedDomainClasses *empty or weighted list of domain class names*
  }

Figure 5.14: General description of a composite visual object

3. **Visual Scene Object**
   A visual scene object represents a set of visual objects which are linked by spatial relationships.

A **visual scene object** (figure 5.15) is composed of:

- A state;
- The description of the main primitive or composite visual object;
- The set of related visual objects;
- The set of spatial relations between the main object and the related visual objects;
- A link to the corresponding perceived image data;
- The list of associated domain classes which are possible semantic interpretations for the visual object and their compatibility and incompatibility values.

VisualSceneObject {
    name *a string*
    state [*hyp miss peirc partial rec*]
    mainvisualobject *A primitive visual object*
    relatedobjects *A list of related objects*
    spatialrelations *A set of spatial relations*
    ImageData *empty or instance of image data*
    AssociatedDomainClasses *empty or weighted list of domain class names*
    }

Figure 5.15: General description of a visual scene object

5.1.4.2 Perceived Scene Description

The perceived scene description is a list of visual objects. It contains all the visual objects created and analyzed during a session of global semantic image interpretation process. It is the memory of the semantic interpretation system and of the visual data management system. Some of the objects contained in the perceived scene description can be *"active"* and the others. *"passive"*. Active objects correspond to objects in processing or waiting for processing. Passive objects correspond to objects that have been processed but that can be used during the processing of other objects (typically for the management of spatial relations).

5.1.4.3 Formal Definitions

- **Definition 6** A **Primitive Visual Object** $O = (s_O, M_O, I_O, D_O)$ is defined by a state s_O, a set of attributes with state $M_O = \{m_i / i \in 1..q\}$, a link with the associated data in image I_O (if they exist) and a semantic interpretation D_O (if it exists). The state s_O could be *hypothesized*, *missing*, *perceived*, *partially recognized* and *recognized*. It defines the current state of the visual object in the global semantic interpretation process. An attribute m_i is a 4-tuple $\langle m_i, t_i, h_i, e_i \rangle$ where:
    - m_i is the name of the attribute

- t_i is the type of the attribute, $t_i \in \theta$
- e_i is the set of perceived values of the attribute, resulting from measurements on images and completed by the lower level modules
- h_i is the expected range of values the attribute, resulting from hypotheses of domain classes. $m_{q,i}$ is the notation of the q-th value of the range of the attribute m_i

- **Definition 7** A **Composite Visual Object** $O = (s_O, M_O, I_O, D_O, Struct_O, Rel_O)$ is also represented by the description of its structural visual object $Struct_O$ and the spatial relation that links the structural objects Rel_O.
- **Definition 8** Let $\mathcal{PSD} = \{\mathcal{O}p \cup \mathcal{O}h \cup \mathcal{O}\}$ the **Perceived Visual Description**. It is an input of the interpretation module.
  $\mathcal{O}p = \{Op_i/i \in 1..p\}$ is a set of perceived **Visual Objects**
  $\mathcal{O}h = \{Oh_i/i \in 1..r\}$ is a set of hypothesized **Visual Objects**
  $\mathcal{O}$ the other visual objects.

5.1.5 Semantic Interpretation Reasoning

All the introduced knowledge and fact concepts are managed by a semantic interpretation problem solving mechanism. This mechanism is implemented in a semantic interpretation engine. The aim is to interpret the perceived scene description from a semantic point of view, using the application domain terminology. The semantic interpretation engine is based on a depth-first traversal of the tree of domain classes (**domain taxonomy**). For each domain class, the semantic interpretation engine performs an hypothesis and test cycle:

- During an hypothesis phase, the semantic engine asks for low level information. It builds a visual data management request by the generation of hypotheses on the visual and relational attributes of expected visual objects. It uses the description of **domain classes**.
- During the test phase, the perceived visual object to interpret is compared to each node of the domain class tree from the current domain class to the leaf classes of the domain taxonomy. The aim is to find the class the perceived visual object belongs to. If low level information is needed, a recursive call to the hypothesis phase is made.

The complete semantic interpretation algorithm corresponds to the algorithm 1. The semantic interpretation process starts with an end user domain request. At the beginning of the global process, the perceived scene description is empty. The general model of the semantic interpretation problem solving can roughly be decomposed in several phases, as shown in the figure 5.16:

1. Initialization phase (see algo 2)
2. Hypothesis building (see algo 3)
3. Semantic Matching (see algo 4)
4. Interpretation Refinement (see algo 1 from lines 13 to 23)

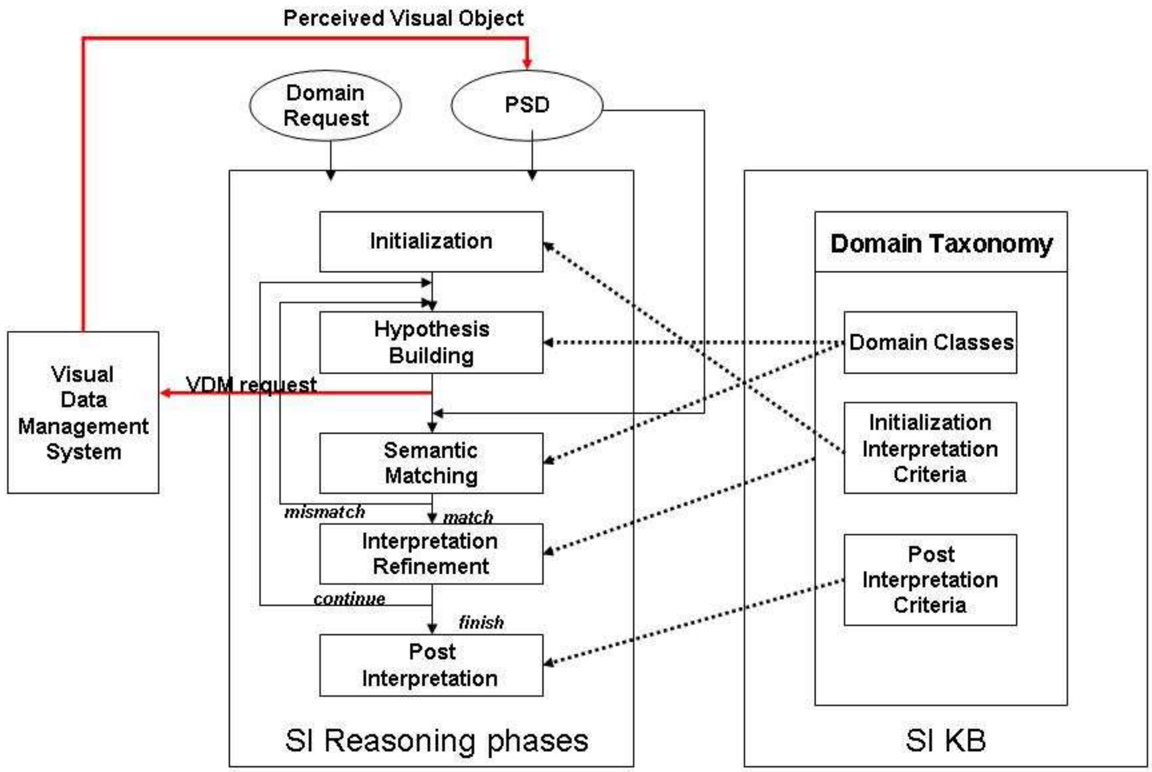


Figure 5.16: Relations between generic knowledge concepts and reasoning phases. Dotted arrows show which types of knowledge base concepts are used for which reasoning phase

5. Post Interpretation phase (see algo 1 line 26)

For each domain class, two alternatives are possible :

1. The perceived scene description is complete with respect to the domain class: i.e. the necessary information has been extracted from images and managed by the visual data management. Thus, the semantic interpretation engine has to find the class the current visual object (or visual spatial configuration of objects) belongs to by matching the current visual object description with the current domain class

2. The perceived scene description is incomplete with respect to the domain class : i.e low level processes are necessary to perform the semantic interpretation. The aim of the semantic interpretation engine is to guide these low level processes by the building of hypotheses. The building of hypotheses is described in algorithm 3.

5.1.5.1 Initialization Phase

The phase of initialization has two main objectives. First, the initialization phase consists in the building of a visual object corresponding to the root of the domain taxonomy.

Algorithm 1 Semantic Interpretation Engine(Domain Request r, Perceived Scene Description PSD)

Given a Semantic Interpretation Knowledge Base IKB and its domain taxonomy taxo

1: **Initialization**(r,PSD,taxo)
2: **while** The list of domain classes (**DCList**) is not empty **do**
3:    **Current_Domain_Class** α:= first in the **DCList**
4:    **Current_Visual_Object** CVO := **FindVisualObject**(PSD,α)
5:    **Hypothesis Building**(CVO, α)
6:    **VisualDataManagementRequestBuilding**(CVO,**Nb_objects**)
7:    Waiting for visual data management results
8:    Updating of PSD
9:    **while** The list of *active* visual objects in PSD is not empty (multiple detection) **do**
10:       CVO:= first in the list
11:       **Semantic Matching**(CVO, α)
12:       Test global compatibility and incompatibility coefficients using compatibility and incompatibility thresholds
13:       **if** α is accepted **then**
14:          α is a possible interpretation of **CVO**: add α in CVO.AssociatedDomainClasses
15:          CVO.state:= *recognized* or CVO.state:= *partially recognized*
16:          **if** α is a leaf of the Domain Taxonomy taxo **then**
17:             Go to 9
18:          **else**
19:             Sort out and add sub-classes of α in **DCList** (using **domain taxonomy** taxo and importance order of sub-classes of α)
20:          **end if**
21:       **else**
22:          Drop Current Domain Class and its sub-classes
23:       **end if**
24:    **end while**
25: **end while**
26: Return the interpretation results : list of domain classes of the recognized visual objects and diagnosis (using **postclassification criteria**)

This visual object is added in the perceived scene description. This visual object is not inevitably processed by the visual data management process. It is used as background information. In most of the cases, the visual concept corresponds to the complete image (as the root domain class corresponds to the abstraction of the complete scene). To set up the values of the visual attributes of this visual object, initialization interpretation criteria are activated. They also initialize the domain knowledge base by setting up the importance degree of the domain classes of the domain taxonomy. Then, this phase consists in the selection of the first domain class to process according to the domain request.

Algorithm 2 Initialization(Domain Request r, Perceived Scene Description PSD, Domain Taxonomy taxo)

```
if r.high_level_goal == Single Detection then
    Add r.DomainClass in the list of domain class to process (DCList)
    Nb_objects := 1
    Hypothesis Building(new Visual Object VO_back , taxo.root)
else if r.high_level_goal == Multiple Detection then
    Add r.DomainClass in the list of domain class to process (DCList)
    Nb_objects := Unknown
    Hypothesis Building(new Visual Object VO_back , taxo.root)
else if r.high_level_goal == Scene Analysis then
    Add taxo.root in the list of domain class to process (DCList)
    Nb_objects := 1
end if
Activation of Initialization Interpretation Criteria
```

5.1.5.2 Hypothesis Building Phase

The role of the hypothesis building phase is to propagate domain knowledge to guide the lower level processes. The hypothesis building is made using the description of the **domain classes**.

5.1.5.3 Semantic Matching Phase

The semantic matching phase consists in the comparison of the current perceived visual object with a domain class.

Algorithm 3 Hypothesis Building(Visual Object VO, Domain Class to process α)

1: **for** Each attribute a of α ($\forall a \in (\mathcal{A}_\alpha, \mathcal{S}_\alpha, \mathcal{R}_\alpha)$) **do**
2:   **if** a is a sub-part attribute ($\forall a \in \mathcal{S}_\alpha$) **then**
3:     Recursive Call to 1 on sub-part taxonomy
4:   **else if** a is a relational attribute ($\forall a \in \mathcal{R}_\alpha$ a represents $R(\alpha, \alpha_{rel})$) **then**
5:     VO is a *Visual Scene Object*
6:     Create a hypothetical relation in VO
7:     Search for a partially recognized or recognized visual object VO_{rel} corresponding to the domain class in relation in the fact base
8:     **if** VO_{rel} exists **then**
9:       Add VO_{rel} in the list of visual objects in relation with VO
10:     **else**
11:       **Hypothesis Building**(New Visual Object NVO_{rel}, α_{rel})
12:       Add NVO_{rel} in the list of visual objects in relation with VO
13:     **end if**
14:   **else if** a is a visual attribute ($\forall a \in \mathcal{A}_\alpha$) **then**
15:     **if** a corresponds to a *spatial structure*: spatialstructure(R,α_i) **then**
16:       VO is a Composite Visual Object
17:       **Hypothesis Building**(Component of VO,α_i)
18:       The relation between the component of VO is R
19:     **else**
20:       Create an hypothetical attribute m corresponding to a in VO
21:     **end if**
22:   **end if**
23: **end for**

Algorithm 4 Semantic Matching (Visual Object **CVO**, Domain Class α)

1: **for** Each attribute a of α ($\forall a \in (\mathcal{A}_\alpha, \mathcal{S}_\alpha, \mathcal{R}_\alpha)$) **do**
2:   **if** a is a sub-part attribute ($\forall a \in \mathcal{S}_\alpha$) **then**
3:     Recursive Call on sub-part taxonomy
4:   **else if** a is a relational attribute ($\forall a \in \mathcal{R}_\alpha$) **then**
5:     Search in the perceived scene description $\mathcal{PSD}$ for the corresponding visual object : **RVO**
6:     Recursive call on the domain class in relation α_{rel} **Semantic Matching**(RVO, α_{rel})
7:   **else if** a is a visual attribute ($\forall a \in \mathcal{A}_\alpha$) **then**
8:     Compute coefficients of compatibility and incompatibility of the values of the corresponding attribute of a in **CVO** with the range of possible values of α
9:   **end if**
10: **end for**
11: Compute global coefficients of compatibility and incompatibility for α and **CVO**

5.2 The Visual Data Management Framework

5.2.1 Analysis of the Visual Data Management Problem

5.2.1.1 Introduction

A visual application is an application that inherently manipulates visual data, involving its management. These visual data should correspond:

- either to symbolic data: i.e. the abstract representation of the scene, its symbolic description essentially qualitative by nature. In our case, the symbols are visual concepts predefined in the visual concept ontology.

- or to a low level information describing the image in terms of image data primitives or numerical descriptors. This information is essentially a quantitative one.

The visual data management problem consists in making the link between these two different kinds of data. Indeed, these data represent the same physical scene but from different points of view in different representation spaces: the perceived one and the symbolic one. It has a role of interface between a perception system (in our case, an image processing system) and a high level symbolic system (semantic interpretation system).

It emphasizes one of the major sub-problems of the image semantic interpretation: the correspondence between symbols and sensor data that refer to the same physical objects. We refer to this problem by the term **semantic gap** in the image retrieval community and by the term **symbol grounding** in artificial intelligence. It represents the lack of coincidence between meaningful descriptions expected by end users and low level features that systems actually compute. In the domain of semantic image interpretation this problem was rarely considered as a problem as such. This problem was often included and limited to a comparison process between the domain dependent classes and the observations. This comparison is often based on complex algorithms or on solutions which are highly dependent on the application domain (e.g. the data abstraction rules in [Ossola et al., 1996]). Our aim is to make this problem explicit. For this, we have made good use of works done in the Robotics community. They refer to this problem as the *Anchoring problem* [Coradeschi, 1999] defined as *the problem of establishing and maintaining the correspondence between the abstract representation and the perceptual data that refer to the same physical objects*. Good introduction on the Anchoring problem can be found in [Coradeschi et al., 2001], [Coradeschi, 1999]. In [Bloch and Saffiotti, 2004], an interesting parallel has been made between anchoring and pattern recognition.

Moreover, the visual information is by nature a spatial information and it appears as an evidence that the management of visual data implies the introduction of spatial reasoning processes. Spatial reasoning consists in the study of the representation, the use and the reasoning about the various spatial relations between objects in the space. There is an intensive and productive research on spatial reasoning in artificial intelligence and we have ever mentioned that spatial structures and spatial reasoning are essential to perception and recognition. However, concerning the spatial reasoning for image interpretation, the statement is about the same that the symbol grounding one : there are only few image interpretation systems [Cohn A.G, 2002] which have integrated a generic spatial reasoning service.

5.2.1.2 Overview of the Visual Data Management Problem

The main goal of the visual data management problem is to automatically make the link between the semantic interpretation module and the program supervision module. It plays a role of interface between high level (symbolic or even semantic) and low level (image) vision.

1. **An usual symbol grounding problem**
   The main sub-problem of the visual data management problem is to ground symbols of the description of the expected scene (the hypothesis made by the high level interpretation system) with the image data structures resulting from the image processing. Indeed, given :

   - A set of image data with associated features extracted from the current image (in our case this set of data is described according to the image processing ontology),
   - a high level symbolic description of the expected scene (in our case a set of visual objects described by a set of visual concepts and spatial relations),

   the role of the visual data management system is to make the correspondence in a fully automatic manner between the two previous sets. An overview of this process is described in figure 5.17. The aim is to fill the gap between visual concepts and image concepts. This process refers to the *Find* functionality of the Anchoring problem [Coradeschi, 1999] and can be achieved by a selection and by a structural matching process between the two different representations.

   The main difficulty of symbol grounding lies in the different natures of the two set of data. Indeed, the representation space of the two kinds of data are different and correspondence links between both types of representations have to be built explicitly or learned [Maillot et al., 2003a].

   Moreover, this process seems to be highly application specific. Indeed, the grounding of the size visual concept *Important Size* with the numerical value of the *area* image region descriptor highly depends on the application. As a consequence, the extraction and the explicitation of a generic knowledge for the symbol grounding task seems difficult. Nevertheless, there is a common sense knowledge on symbol grounding which is application independent. For example, whatever the application is, the correspondence link (or **grounding relation**) between the image region descriptors eccentricity, circularity, rectangularity and the shape visual concept *Geometric Surface* is obvious.

   We propose to model and to make explicit this kind of knowledge in a Visual Data Management Knowledge Base.

   We present our approach with one of the typical example of figure 5.17. This example is deliberately chosen simple because the aim is both to illustrate knowledge concepts and the specific visual data management engine behavior.

   Given the simplified description of an hypothetical Visual Object **VO1** :

   - Hypothesis : White Fly
   - Description in terms of **Visual Concepts** (in italic)
     - *Heart-Like Surface*
     - *White* OR *Gray*
     - *Medium Size*

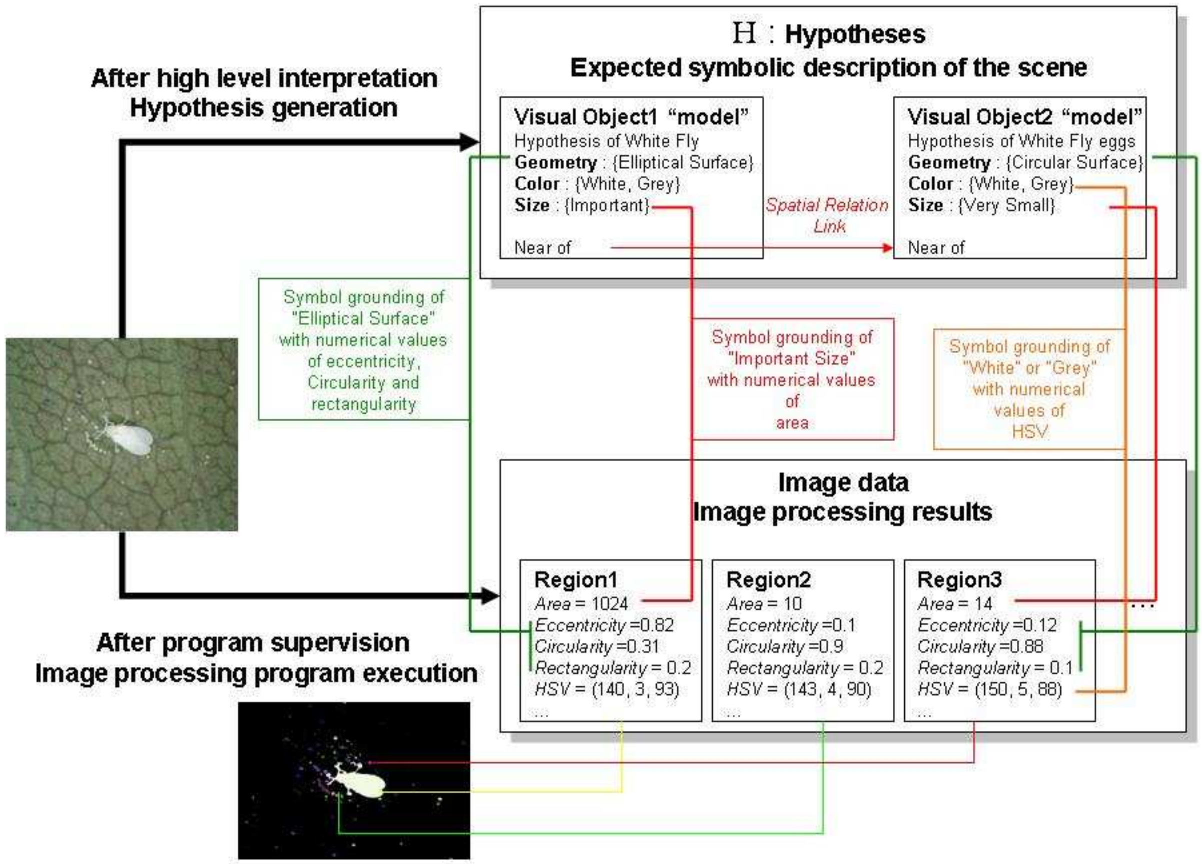


Figure 5.17: Graphical illustration of our symbol grounding problem

It exists a common sense link between some visual concepts and well known features that can be extracted from images. For example, it seems obvious to lie the visual Concept *White* with the following set of image color descriptors:$\{H, S, V\}$. In the same manner the following set of image shape descriptors {*Eccentricity, Circularity, Rectangularity, Convexity, Compactness, ...*} is quite obviously linked to the visual concept *Geometric Surface* and to all its sub-concepts.

As the link between visual concepts and image data, each descriptor is modeled as a fuzzy linguistic variable with a set of linguistic values and their associated fuzzy sets. Fuzzy set theory enables the representation of the imprecision. It is close to the way humans would approach this problem of correspondence. Indeed a lot of visual notions used by humans to describe objects are fuzzy variables. For example, due to the approximation we use when we describe shapes, a circular shape does not only correspond to region in image which circularity(form factor) is 1. A visual description is by nature imprecise.

The link between **visual concepts** and **image data descriptors** constitutes **the symbol grounding link** $\mathcal{G}$ between **Visual Concept** and **Image Data descriptors**. As in the Anchoring framework presented in [Coradeschi, 1999], the symbol grounding link encodes the correspondence between visual concepts and admissible numerical values of image descriptors.

In the previous example, the symbol grounding link could be:

- $\mathcal{G}$(Heart-Like Surface) = { ellipticity == medium & compactness == high & convexity==low }
- $\mathcal{G}$(White)= {hue(H) in [104..180] & value(V) in [127..255]}
- $\mathcal{G}$(Gray)= {hue(H) in [104..180] & value(V) in [127..255]}
- $\mathcal{G}$(MediumSize) = {length in [0.8 ..1.2] unit :milimeter}

Building the grounding link consists in defining the set of linguistic values with their corresponding fuzzy sets for the image data descriptors associated with instances of visual concepts. Concerning the acquisition process of this knowledge, it is actually hand coded. The approach is similar in [Mezaris et al., 2004]. Nevertheless some interested works in our team [Maillot et al., 2003a] propose an automatic acquisition of this knowledge by learning techniques.

Concerning the behavior of the visual data management engine, it uses the symbol grounding link $\mathcal{G}$ on one hand to constrain the image processing problem and in the other hand to build a symbolic description in terms of visual concepts of the image data. As a consequence, the symbol grounding task can be modeled by a **fuzzy matching process** between visual objects (structured set of visual concepts) and between a structured set of image data entities and associated descriptors.

2. **A process involving grouping**
   It seems very limited to reduce the process of symbol grounding to a matching process. Indeed, the extraction of image data by image processing programs is often imprecise. It could be missing, incomplete, erroneous and uncertain. Although the fuzzy matching process enables us to manage this imprecision, it is not sufficient: i.e. the matching process relies too much on the quality of image processing results. It is not possible to establish the correspondence with high level representation of physical objects by a direct matching process. Indeed, most of the time, due to imperfect image processing results, it is necessary to add a step of grouping of image data (to cope with problem of under-segmentation or over-segmentation).

   Moreover, visual grouping is a natural process for the human natural perception. It was the subject of an intensive research in the Gestalt school of psychology. Gestalt theory argues that human vision performs domain independent perceptual grouping to group together parts of image that most likely represent a single object in the scene. This perceptual grouping is done according to several pointed out factors: proximity, similarity, closure, continuity and symmetry. As a consequence, it seems essential to add a grouping process in the visual data management task.

   Grouping is defined as the process that organizes image data entities into higher level structures. Interesting thoughts about grouping can be found in [Engbers and Smeulders, 2003]. In most cases, the design of visual grouping process is application specific. Nevertheless, it exists some works on generic visual grouping. A general statement about these works is that they are all data-driven, i.e. they do not use high level knowledge about expected object spatial configurations to manage the visual grouping process. In [Zlatoff, 2004], it is argued that to make a system aware of what it treats (though not fully dependent), the visual grouping has to be controlled by high level knowledge. We agree with this statement. Visual grouping is a data-driven (perceptual grouping) and model-driven (knowledge based grouping) process. Spatial relations play an important role in the visual grouping process and it emphasizes another time the role of spatial reasoning in the visual data management problem.

According to [Engbers and Smeulders, 2003], for any generic grouping framework, the following concepts are defined:

- The data to be grouped: it defines the smallest entities into which a high level structure can be decomposed.
- The targeted structure: it describes the high level properties of the high level structure.
- A grouping measure.
- A process of grouping.

The study of a lot of works concerning the grouping in vision enables us to draw up the following points:

- A lot of grouping processes begin with a **graph building** of low level structures. The way this graph is built highly depends on the type of the data to be grouped and on the expected high level structure. Nevertheless, in most of the cases, based on the Gestalt law of *proximity*, recognized as the most important one in a grouping process, the graph is a proximity graph (e.g. adjacency graph, relative neighboring graph ...)
- Then, the grouping process consists in computing a **grouping measure** between all the linked edges of the graph. This grouping measure is also highly dependent on the type of the data to be grouped and on the expected high level structure.
- The grouping process is an iterative process consisting in grouping all the edges of the graph whom grouping measure is under a predefined threshold.
- The grouping process is stopped when the expected structure is reached or when the grouping measure between all the edges is above a given threshold.

In our approach we have decided to delegate the computational part of the grouping process to the image processing module. Indeed, the grouping process can be modeled as a generic functionality of image processing and it exists a lot of image processing programs that can be used by the grouping process. It gives birth to the notion of *grouping operator* which can be decomposed into:

- a *graph building operator*;
- a *grouping measurement operator*;
- a *grouping decision process* based on the results of the *grouping measurement operator*.

Nevertheless, the grouping process control is a task of the visual data management system. Indeed, in particular, it :

- takes decisions to activate the grouping process according to the image processing results and the visual object hypothesis. These decisions are represented by **evaluation criteria**.

- propagates the necessary constraints to manage the grouping process. The description of the expected visual object: e.g. appearance, expected number of objects, spatial relationships, are useful to conduct the grouping process.

3. **Top down guiding of lower level processes** The reliability of image processing results is a point of strong importance. Unreliable image processing results can lead to wrong interpretations. Nevertheless, we have already mentioned that a perfect image segmentation does not exist and we have seen that uncertain data management processing and visual grouping can help to deal with the imprecision of image processing results. Another way to make the image processing results more reliable is to provide a top down guidance. Although this fact has been known for a long time, this problem were rarely studied in a generic way. This fact reveals another problem to deal with: given the expected visual scene description, how to guide the extraction process of image data in a generic way? In our case, this guidance implies to make the interaction with the program supervision module. A program supervision system receives as input an image processing request: the image processing functionality to achieve, the data on the particular case to work on and particular constraints. To guide the image processing process, the visual data management system has to build image processing requests by:

   - The choice of the appropriate image processing functionality among a set of predefined basic functionalities provided by the image processing ontology and according to the current state of the visual data management process.
   - The building of appropriate constraints on the image content.

   For example, the use of known spatial relations between visual objects can enable to define the area of interest on images. The knowledge of the discrimination between two adjacent regions is another example of constraints. This latter constraint can be used by the program supervision system to choose a simple color segmentation algorithm to select the object of interest.

   We can sum up that by defining a sub-task of the general visual data management task which consists in inferring image processing constraints using the expected scene description by visual concepts and by spatial relations. Our approach is based on inference criteria. For each generic image processing functionality provided by the image processing ontology, a visual content context is defined. For instance, the simplified visual content context associated with the *object extraction* functionality of image processing contains, as proposed in [Clouard et al., 1999], the following attributes:

   - knowledge about the appearance of the object to extract:
     - image data type (chosen according to the predefined image processing ontology);
     - discriminative object color;
     - discriminative object texture;
     - discriminative object luminosity;
     - relative object size;
     - area of interest.
   - knowledge about the background:
     - cluttered background.

- knowledge about the scene:
  - number of objects,
  - object repartition in images.

The visual data management system gives values to the previous attributes by the activation of generic criteria contained in the visual data management knowledge base. By the following we refer to these criteria by **visual object extraction criteria**. They represent this kind of common sense knowledge:

- *If the expected visual object is described as a thin line then the probability for it to be represented by ridges on images is high*
- *If the expected visual object is described as a compact surface, its representation on images is region*

These criteria model an experienced knowledge about visual concepts and their link to image concepts. It is different from the hypothesis coming from the high level domain knowledge. The points of view are different as illustrated in figure 5.18.

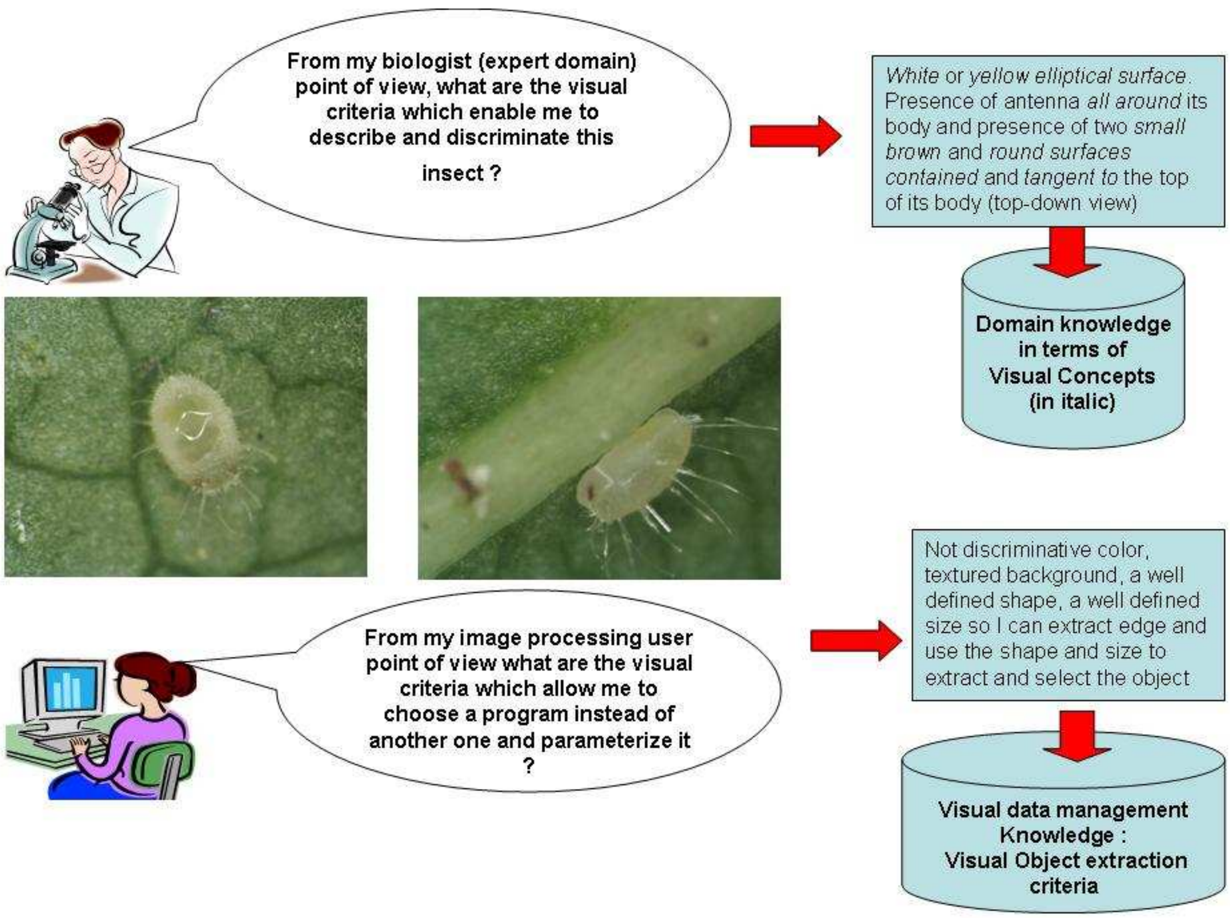


Figure 5.18: Illustration of the different points of views concerning the top-down guiding of lower level processes by high level knowledge and the guiding of the program supervision process by visual object extraction criteria

4. **A highly spatial reasoning process**
We have seen that the use of spatial relations is widely used in all the sub-problems of visual data management. In several cases, the visual appearance of visual objects is not sufficient to make high or low level decision and an information about their spatial configuration is useful. Spatial reasoning is a big part of the visual data management reasoning. Indeed, spatial relations can be used to:

    - infer spatial relation from another ones,
    - verify the consistency of the detected spatial configuration,
    - guide the lower level processes by the propagation of spatial constraints,
    - guide the object recognition process,
    - perform visual grouping of image data into higher level structures.

    The main difficulty lies in the management of spatial relations of different natures simultaneously. The involved spatial relations can be topological relations, distance relations or orientation relations. The visual data management knowledge base contains knowledge of spatial relations and knowledge of how to use them to achieve the sub-tasks of the visual data management.

5. **Evaluation of the image processing results**
We have already mentioned the great importance of the image processing phase. In particular, the phase of object extraction, currently called object segmentation is of great importance for the next steps of the semantic image interpretation. A good segmentation refers to the fact that the corresponding segmented image reflects the expected visual object correctly. According to this definition we argue that the step of evaluation of such image processing results is beyond the scope of the image processing system but takes part in the visual data management system. The evaluation of segmentation results currently leads to three different assessment results:

    - correct segmentation,
    - over-segmentation,
    - under-segmentation.

5.2.2 Overview of the Visual Data Management Framework

We propose to emulate the different strategies of visual data management by a visual data management knowledge based system. It is typically composed of:

1. **A visual data management knowledge base (VDM knowledge base)**
2. **A visual data management engine (VDM engine)**
3. **A visual data management fact base (VDM fact base)**

5.2.2.1 The Visual Data Management Knowledge Base

The visual data management knowledge base is written by a visual data expert. It depends on the visual concept ontology and on the spatial relation ontology used to describe the application domain and on the image processing ontology used to describe the image (see chapter 4). The achievement of visual data management requires the clear description of the different types (symbolic or perceived) of handled data. Knowledge of the grounding link between symbolic and sensor data is also needed to solve the visual data management problem. The visual data management encapsulates this expertise in a declarative manner. Moreover, the knowledge base also contains expertise on how to perform automatically visual data management action, such as spatial reasoning, visual constraint building or visual grouping. This expertise is an inferential knowledge. For the cognitive vision platform, we propose a general model of knowledge for visual data management: i.e. the set of application independent concepts used to build and structure a visual data management knowledge base. This model is composed of:

- **Visual data** are explicit descriptions of the different types of handled data. They are implemented by frames. It exists two kinds of visual data in our framework:
  - Symbolic visual data are symbols. They correspond to the description of the data coming from or intended to a symbolic reasoning module. In our framework, the symbolic reasoning module is the semantic interpretation module and **symbolic visual data** are **visual concepts**. They are provided by the visual concept ontology.
  - Perceived visual data are sensor data. They are explicit descriptions of data coming from or intended to a perception module. In our framework, the perception module is the program supervision module and **perceived visual data** are **image data**. They refer to the entity concepts of the image processing ontology.
- **Fuzzy Descriptors** are associated with **visual concepts** and with **image data**. They play an important role because they enable to make the link between visual concepts and image data. Fuzzy descriptors refer to the descriptor concepts of the image processing ontology.
- **Descriptor Sets** are associated to image Data. They are structured sets of descriptors which characterize image data. They also refer to the descriptor concepts of the image processing ontology.
- **Spatial relations** are explicit descriptions of the different kinds of spatial relations used to describe spatial organisations.
- The inferential knowledge is represented by various **visual data management criteria**, implemented by rules. They are used to describe decisions during the visual data management problem solving. These criteria are:
  - **Object extraction criteria** either linked to **visual concepts** or to **spatial relations**. They are used to constrain program supervision request.
  - **Spatial deduction criteria** are linked to **spatial relations**. They are used to infer spatial relations from another ones during the visual data management process.
  - **Visual evaluation criteria** are used to diagnose the results of the program supervision module from a visual data management point of view.
- **Image processing functionalities** and their associated **visual content context**.
- **Visual data management requests** express queries of the semantic interpretation module. We have already described them in the section 5.1.3.

- **Program supervision requests** express image processing queries for the program supervision module.

A detailed description of these visual data management knowledge concepts is presented in section 5.2.3.

5.2.2.2 The visual data management engine

The visual data management engine is application independent. Its role is to use the knowledge stored in the visual data management knowledge base to create the *"anchor"* between instances of image data and visual objects (a structure set of instances of visual concepts and spatial relations) in the fact base. To achieve this, the visual data management engine performs top-down and bottom-up strategies. They include constraint propagation, fuzzy matching, visual object instantiation and spatial reasoning.

5.2.2.3 The visual data management fact base

The visual data management fact base depends on the visual data management request sent by the semantic interpretation module. This request describes the current visual data management problem to solve. The fact base contains the instances of symbolic data which describes the current problem. Identically to the semantic interpretation fact base, the visual data management fact base is structured in **visual objects** and **perceived scene description**. Moreover, the visual data management fact base also contains the instances of image data resulting from the program supervision module. During the visual data management reasoning, some data in the fact base can be modified, added or deleted.

5.2.3 Proposed Knowledge Model for Visual Data Management

This section details the important concepts involved in the visual data management process. These concepts have been modeled from the point of view of software reuse as well as from the point of view of the *cognitive* process of visual data management.

5.2.3.1 Visual Concepts

Visual concepts are descriptions from a data management point of view of the concepts of the **visual concept ontology**. They are qualitative terms used to visually describe the real world scene. They are the symbolic data of our framework. According to the visual concept ontology, they are organized in a **visual concept taxonomy**. The representation of a **visual concept** is composed of:

- A name;
- A specialization link to situate it in the visual concept taxonomy;
- A grounding link which is the list of fuzzy descriptors associated to the visual concept and their description;
- Object extraction criteria (optional).

The general syntax used to represent visual concepts is described in figure 5.19.

VisualConcept {
  name *a term of the visual concept ontology*
  SuperConcept *name of the parent in the taxonomy*
  Grounding Link
    list of fuzzy descriptors
  Object Extraction Criteria
    list of object extraction criteria }

Figure 5.19: General syntax of visual concepts in the visual data management knowledge base

5.2.3.2 Fuzzy Descriptors

Fuzzy descriptors are associated to visual concepts to define the grounding link with image data features. A fuzzy descriptor models an image data feature provided by the image processing ontology as a linguistic variable with a fuzzy formalism as in [Coradeschi et al., 2001]. Its representation is composed of:

- A type (e.g. float, integer, symbol,...);
- A name (it corresponds to a term of the image processing ontology);
- A comment (e.g. an informal comment on the descriptor);
- A set of linguistic values (e.g. the set of values that can be taken by the linguistic variable);
- A domain (e.g. the range of numerical values of the descriptor);
- A set of fuzzy sets. A fuzzy set is associated with each linguistic value. A fuzzy set is defined by its membership function on the domain. In our framework, fuzzy sets are represented by trapezoidal membership function.
- A unit (optional) if the feature represents a real physical measurable property (as the length for example).

In some cases, if an image data feature represents a real physical measurable property, the representation of the fuzzy descriptor has not associated linguistic values and fuzzy sets.

Fuzzy descriptors are represented as visual concept grounding link attributes. Their general syntax, inside a visual concept is :

Grounding Link
  Type **name** descriptor_name
    comment *"an informal comment on the descriptor"*
    linguistic_values [*a set of descriptor linguistic values*]
    domain [*numerical range of the descriptor*]
    fuzzy_sets
    list of fuzzy sets
    unit [*list of common units for the descriptor*]

To illustrate the concepts of **visual concept** and **fuzzy descriptors**, the representation of the visual concept *Elliptical Surface* is described in figure 5.20. According to the process of inheritance between visual concepts, a sub visual concept inherits of the grounding relation of its parents. The specialization corresponds either to additional fuzzy descriptors in the grounding link or to the restriction of existing fuzzy descriptors.

VisualConcept {
  name *Elliptical_Surface*
  SuperConcept *Geometric Surface*
  Grounding Link
    Symbol name eccentricity
      comment *ratio of the length of the longest chord to the longest chord perpendicular to it*
      linguistic-values: [very_low low medium high very_high]
      FuzzySet
      $F_{verylow} = \{0.0, 0.0, 0.19, 0.21\}$
      $F_{low} = \{0.19, 0.21, 0.38, 0.42\}$
      $F_{medium} = \{0.38, 0.42, 0.575, 0.625\}$
      $F_{high} = \{0.575, 0.625, 0.76, 0.84\}$
      $F_{veryhigh} = \{0.76, 0.84, 1, 1\}$
      Domain: [0 1]
      unit: none
    Symbol name compactness
      comment *Measure of how the shape is closely-packed*
      ...
    Symbol name ellipticity
      comment *Euclidian ellipticity : distance between fitting ellipse and region boundary*
      ...
  }

Figure 5.20: The representation of the visual concept *Elliptical_Surface* in the visual data management knowledge base

5.2.3.3 Image Data

Image data are explicit descriptions of the type of image data used to describe program supervision results. They are the different types of data structures that can be extracted from images. They are provided by the image processing ontology. In concrete terms, image data are pixels or set of pixels as described in chapter 4. Their common representation is composed of:

- A name;
- A specialization link to situate it in the image processing taxonomy;
- An information on the **interior** and on the **boundary** of the image entity; **Interior** and **boundary** are instances of **Image Data**;

- A list of **descriptor sets**. They are the numerical descriptors which characterize image data.

Interior, **boundary** and **descriptor sets** are represented as attributes of **image data**. Their values are computed and completed by the program supervision system.

Image data are used to described program supervision results or to instantiate program supervision requests. In consequence, they represent either **data** or operator **arguments** in the program supervision knowledge model (see section 5.3.2). Their syntax is identical to the syntax used to describe data in the program supervision knowledge model. The general syntax of image data is represented in figure 5.21.

ImageData {
  name *image data name (entity concept in the IP ontology)*
  comment *"An informal comment on the data"*
  Subtype Of *the name of the parent class in the image data taxonomy*
Attributes
  ImageData name interior
  ImageData name boundary
  List of descriptor sets }

Figure 5.21: General syntax of image data in the visual data management knowledge base

5.2.3.4 Descriptor Set

Descriptor Sets are structured sets of descriptors. This knowledge structure enables to organize and to gather descriptors of the same family (size descriptor, shape descriptor,...). Moreover, with this representation, the association of a set of descriptors to an image data is more flexible. Their general syntax is given in figure 5.22.

DescriporSet {
  name *a name*
  comment *"An informal comment on the descriptor"*
  Subtype Of *the name of the parent descriptor set*
Attributes
  List of descriptors (i.e. descriptor concepts in the IP ontology }

Figure 5.22: General syntax of descriptor sets in the visual data management knowledge base

5.2.3.5 Spatial Relation

Spatial Relation is the main knowledge concept used to represent knowledge of space: i.e. the spatial relationships between objects. Spatial relation is the explicit representation of a spatial relation provided by the spatial relation ontology, i.e. the explicit description of spatial relation properties. This explicit representation of spatial relationships enables to process them independently and to perform a spatial reasoning only based on spatial relations. According to the set theory of the space, spatial relations can have a predefined set of properties. Be $\mathcal{O}$ a set of objects. Properties of spatial relations are:

- *Symmetry*: a spatial relation $\mathbf{R}$ is symmetric if:
  $\forall(O_i, O_j) \in \mathcal{O}^2 : \qquad O_i \,\mathbf{R}\, O_j \quad \Leftrightarrow \quad O_j \,\mathbf{R}\, O_i$

- *Inverse Relation* : the inverse spatial relation $\mathbf{R}^{-1}$ of a spatial relation $\mathbf{R}$ is the spatial relation such that:
  $\forall(O_i, O_j) \in \mathcal{O}^2 : \qquad O_i \,\mathbf{R}\, O_j \quad \Leftrightarrow \quad O_j \,\mathbf{R}^{-1}\, O_i$

- *Transitivity* : a spatial relation $\mathbf{R}$ is transitive if:
  $\forall(O_i, O_j, O_k) \in \mathcal{O}^3 : \qquad O_i \,\mathbf{R}\, O_j \quad , O_j \,\mathbf{R}\, O_k \quad \Leftrightarrow \quad O_i \,\mathbf{R}\, O_k$

- *Composition* : given two spatial relations $\mathbf{R}_1$ and $\mathbf{R}_2$ and 3 objects, $O_1, O_2, O_3 \in \mathcal{O}$ such that $O_1 \,\mathbf{R}_1\, O_2$ and $O_2 \,\mathbf{R}_2\, O_3$, the composition of the two spatial relations $\mathbf{R}_1$ and $\mathbf{R}_2$ is the spatial relation $\mathbf{R} = \mathbf{R}_1 \circ \mathbf{R}_2$ such that $O_1 \,\mathbf{R}\, O_3$
  The property of *composition* is a powerful property because it enables the inference of relations.

- *Complement* : the complement of a spatial relation $\mathbf{R}$ is the spatial relation $\mathbf{R^C}$ such that:

  - $\mathbf{R}$ and $\mathbf{R^C}$ are disjoint , i.e. $\forall(O_i, O_j) \in \mathcal{O}^2 : \qquad O_i \,\mathbf{R}\, O_j \quad \Leftrightarrow \quad (\text{not}) O_i \,\mathbf{R^C}\, O_j$
  - $\forall(O_i, O_j) \in \mathcal{O}^2 : \qquad O_i \,\mathbf{R}\, O_j \quad \text{or} \quad O_i \,\mathbf{R^C}\, O_j \quad \text{holds}$

In reference to the spatial relation ontology, it exists two kinds of spatial relations:

1. **Topological Spatial Relations** represent topological relations

2. **Metric Spatial Relations** represent fully metric spatial relations, i.e. distance and orientation relations.
   Metric Spatial Relations can be established in terms of three basic concepts [Cohn A.G, 2002]: the primary object (PO), the reference object (RO) and the frame of reference. To be able to deal with **Metric Spatial Relation** as binary relations, the frame of reference is explicitly represented. If the primary object is a part (**dependent part**) of the reference object, the reference frame is the reference object coordinate system. If the primary object is an independent object of the scene the reference frame is the image coordinate system.

At last, another important aspect of spatial relations is their meaning. For example, the spatial relation O_1 **Is Near Of** O_2 means that the distance between O_1 and O_2 is greater than a predefined threshold. This meaning is explicitly represented by a **condition**. Each spatial relation has an associated **condition**. The verification of the condition implies the verification of the **spatial relation**. This **condition** is used to make the **grounding** of spatial relations in images and spatial relations in the real world as it is done in [Egenhofer and Sharma, 1993]. The computation of relations on images is the same that

Name of the relation	**Conditions**
$EQ(x, y)$	$\{x^\circ - y^\circ = \emptyset, y^\circ - x^\circ = \emptyset, x^\circ \cap y^\circ \neq \emptyset, \partial x \cap \partial y \neq \emptyset\}$
$NTTP(x, y)$	$\{x^\circ - y^\circ = \emptyset, y^\circ - x^\circ \neq \emptyset, x^\circ \cap y^\circ \neq \emptyset, \partial x \cap \partial y = \emptyset\}$
$TTP(x, y)$	$\{x^\circ - y^\circ = \emptyset, y^\circ - x^\circ \neq \emptyset, x^\circ \cap y^\circ \neq \emptyset, \partial x \cap \partial y \neq \emptyset\}$
$NTTP^{-1}(x, y)$	$\{x^\circ - y^\circ \neq \emptyset, y^\circ - x^\circ = \emptyset, x^\circ \cap y^\circ \neq \emptyset, \partial x \cap \partial y = \emptyset\}$
$TTP^{-1}(x, y)$	$\{x^\circ - y^\circ = \emptyset, y^\circ - x^\circ = \emptyset, x^\circ \cap y^\circ \neq \emptyset, \partial x \cap \partial y \neq \emptyset\}$
$PO(x, y)$	$\{x^\circ - y^\circ \neq \emptyset, y^\circ - x^\circ \neq \emptyset, x^\circ \cap y^\circ \neq \emptyset, \partial x \cap \partial y \neq \emptyset\}$
$EC(x, y)$	$\{x^\circ - y^\circ \neq \emptyset, y^\circ - x^\circ \neq \emptyset, x^\circ \cap y^\circ = \emptyset, \partial x \cap \partial y \neq \emptyset\}$
$DC(x, y)$	$\{x^\circ - y^\circ \neq \emptyset, y^\circ - x^\circ \neq \emptyset, x^\circ \cap y^\circ = \emptyset, \partial x \cap \partial y = \emptyset\}$
$DR(x, y)$	$\{x^\circ \cap y^\circ = \emptyset\}$
$PP(x, y)$	$\{x^\circ - y^\circ = \emptyset, y^\circ - x^\circ \neq \emptyset, x^\circ \cap y^\circ \neq \emptyset\}$
$PP^{-1}(x, y)$	$\{x^\circ - y^\circ \neq \emptyset, y^\circ - x^\circ = \emptyset, x^\circ \cap y^\circ \neq \emptyset\}$

Figure 5.23: The conditions for the set of topological relations

verifying a set of conditions on image data.
For the topological relations, the conditions are based on operations on sets [Egenhofer, 1993] which are based on the intersection and the difference of the notions of interior and boundary of the objects in relation. In [Le Ber and Napoli, 2002], [Mangelinck, 1998], the authors, interested in satellite image understanding, lay stress on the problem of the boundary definition on images. They propose four set operations to define the conditions associated to the topological relations based on the intersection and the difference of the boundary and interior of two objects. If x and y are the two objects in relations defined by their interiors (x°, y°) and their boundaries $(\partial x, \partial y)$, the four operations are: the intersection of the interior $x^\circ \cap y^\circ$, the intersection of the boundaries $\partial x \cap \partial y$, and the differences of the interiors $x^\circ - y^\circ$ and $y^\circ - x^\circ$. The conditions for topological relations given by the **spatial relation ontology** are described on figure 5.23.

To summarize, the common representation of **spatial relations** is composed of:

- A name;
- A comment;
- A specialization link (to represent the hierarchy of spatial relations);
- The complement of the spatial relation (if it does not exist, the complement is set to **None**);
- The inverse of the spatial relation (if it does not exist, the inverse is set to **None**);
- Information on the property of symmetry of the spatial relation, i.e. if it is symmetric or not;
- The set of conditions to verify the relation;
- Information on objects in relations (e.g. their name, their types). In our framework, objects in relation are instances of visual objects.

The general syntax of spatial relations is given in figure 5.24. In addition to this common information, a **metric spatial relation** has information on the reference frame. In our framework, the reference frame is either the image coordinate system either the related object coordinate system.

SpatialRelation {
  name *the name (according to the spatial relation ontology)*
  SuperRelation *name of the parent spatial relation*
  Inverse *a spatial relation*
  Complement *a spatial relation*
  Symmetry [*True False*]
  Conditions
    set of conditions that verify the spatial relation
  Objects_In_Relation
    VisualObject **name** object1
    VisualObject **name** object2 }

Figure 5.24: General syntax of a spatial relation

Spatial relation representation			
Name	Externally Connected (EC)	Left Of	Near Of
Parent	Discrete	Orientation Relation	Distance Relation
Inverse	Externally Connected	Right Of	Near Of
Complement	None	None	Far From
Symmetry	True	False	True
Condition	$EC(x, y) = \{x^\circ - y^\circ \neq \emptyset, y^\circ - x^\circ \neq \emptyset, x^\circ \cap y^\circ = \emptyset, \partial x \cap \partial y \neq \emptyset\}$	Angle$(x, y) \in [-\frac{\pi}{4}, \frac{\pi}{4}]$	Distance$(x, y) \in [10, 20]$

Figure 5.25: Abstract view of the representation of different spatial relations in our framework

An abstract view of three spatial relations is shown in figure 5.25. This representation enables to manage the verification process of spatial relations according to pre-exiting ones and to verify the consistency of detected spatial configurations using the properties of specialization, complement and inverse of spatial relations. For example, if the system is asked to compute a specific spatial relation $\mathcal{R}$ between two objects and if the complement spatial relation $\mathcal{R}^c$ is already present between this two objects, then the relation $\mathcal{R}$ is false.

5.2.3.6 Visual Data Management Criteria

The visual data management knowledge base is composed of different criteria. They represent dynamic knowledge about decisions (e.g. how to constrain a program supervision request, how to infer spatial relations from existing ones, how to evaluate program supervision results). In our framework, criteria are implemented by rules which are attached either to visual concepts or to spatial relations. This locality of criteria enables each piece of knowledge to carry its own inferential knowledge. The external form of each visual data management criteria is the same than **context criteria** in the interpretation knowledge model :

Let *declarations*
If *premise*
Then *action*

1. **Object extraction criteria** are used to decide how to constrain the building of a program supervision request according to instances of visual concepts and spatial relations. In particular, a **spatial constraint criteria** is attached to each spatial relation. These criteria defines the acceptance area for the object in relation with respect to the reference object. For example, if we want to extract an object B which is an the right of a perceived object A, we will focus our research in the image area defined by the center G of the object A and two lines going through G and of angles $\frac{\pi}{4}$ and $-\frac{\pi}{4}$ in the reference coordinate system.

2. **Spatial deduction criteria** are used to deduce spatial relations from another ones. They are only associated to spatial relations. These criteria enable to represent the known properties of transitivity and composition of spatial relations.

   - **Transitivity criteria** represent the property of transitivity of spatial relations. They are defined for the spatial relations *Left Of, Right Of, Above, Behind, Equals,* $NTTP$,$NTTP^{-1}$.
   - **Composition criteria** are defined to represent the property of composition between the different spatial relations. We have begun our work with existing composition table for topological relations [Le Ber and Napoli, 2002] and we have completed it with orientation and distance relation. Most of the time, the topological, distance and orientation relations are processed independently. Only few works were interested in the composition of spatial relations of different types [Sistla and Yu, 2000]. We make the hypothesis that the **metric relations** are in the same reference system.

3. **Visual evaluation criteria**
   Evaluation criteria state information on how to evaluate results of the program supervision module according to predefined constraints defined by the visual data management module. We will see that the program supervision knowledge base also includes some **evaluation criteria**. The latter states information to access the quality of the operator results after its execution. We propose to add **evaluation criteria** at the visual data management level to make a distinction between two kinds of evaluation:

   - The evaluation intrinsic to each low level operator. The scope of these evaluation criteria is restricted to the program supervision process.
   - The evaluation of the image processing results according to the constraints given by an higher module. Their consequences are actions at the level of the visual data management module. These evaluation criteria are based on the constraints given by the expected visual object description.

5.2.3.7 Image Processing Functionality

The visual data management knowledge base contains a set of **image processing functionalities** (provided by the image processing ontology) used to build requests for the program supervision system. As they are a mean to communicate with the program supervision system, their representation in the visual data management framework is really close than their representation in the program supervision framework (see section 5.3.1.3). The general syntax of image processing functionalities is given in figure 5.26.

IPFunctionality {
  name *the name of a goal concept of the image processing ontology*
  comment *"an informal comment on the functionality"*
  Input Data
    list of input data with their types and names
  Output Data
    list of output data with their types and names }

Figure 5.26: General syntax of an image processing functionality

To each image processing functionality corresponds a **visual content context**. **Visual content contexts**, implemented by frames, represent the set of information that enables to constrain a particular image processing problem (e.g. a particular image processing functionality). They corresponds to **domain objects** in the program supervision knowledge model. Their general syntax is describe in figure 5.27

VisualContentContex {
  name *contextname*
  IPFunctionality *name of the concerned functionality*
  Attributes
    A list of attributes }

Figure 5.27: The general syntax of a visual content context

5.2.3.8 Program Supervision Requests

Program supervision requests express image processing queries from the visual data management system to the program supervision system. These program supervision requests have to be comprehensible by the program supervision module. As the program supervision knowledge base model is well experienced, the representation is adapted from the representation of **request** as described in [Moisan et al., 2001]. Program supervision requests are queries for an abstract image processing functionality (provided by the functionality concepts of the image processing ontology) on particular data, under particular constraints. They describe the initial program supervision program. The general syntax of a program supervision request is depicted in figure 5.28. They are instances of IPfunctionalities.

Examples of **image processing functionalities** and their associated visual content context are shown in table 5.29.

Program Supervision Request {
IPFunctionality **name** *a name*
comment *"an informal comment on the IP request"*
Attributes
list of attributes (functionality input data assignment)
}

Figure 5.28: The general syntax of a program supervision request

Image Processing Request		
Image Processing Functionality to achieve	Object Extraction	Grouping
Information on Input Data	Input Image	Set of Image Entities
Information on Visual Content Context	Image Entity type, Discriminative Color, Discriminative Texture, Discriminative Intensity, Area of Interest, Number of Objects, Object repartition, Background	Image Entity Type, Spatial Relation, Size, Color

Figure 5.29: Abstract view of the representation of image processing requests

5.2.4 Visual Data Management Fact Base

Identically to the semantic interpretation fact base, the visual data management fact base is structured in **visual objects** and **perceived scene description**. Moreover, the visual data management fact base also contains instances of image data resulting from the program supervision module. During the visual data management reasoning, some data in the fact base can be modified, added or deleted.

5.2.5 Formalizations

- **Definition 1** $\theta = \{C_i/i \in 1..n\}$ is the set of **visual concepts** defined in the previous section

- **Definition 2** Let $\mathcal{P}e = \{pe_i/i \in 1..n\}$ a set of perceived data
  $\mathcal{P}e$ is the current set of *instances* of $\mathcal{CP}e = \{Cpe_i/i \in 1..m\}$ defining and describing the possible type of perceived data (**Image Data**)
  $\preceq_{\mathcal{CP}e}$ is a partial order between **Image Data**.
  $\mathcal{CP}e = (\mathcal{E}, \mathcal{D})$ where $\mathcal{E}$ is the sub-set of $\mathcal{CP}e$ corresponding to perceived entities and $\mathcal{D}$ is the sub-set of perceived descriptors or low level descriptors

For a perceived entity $e \in \mathcal{E}$, we define $desc : \mathcal{E} \to \mathcal{D}$ so that $desc(e)$ is the set of possible descriptors of e.

$\langle \mathcal{CP}e, \preceq_{\mathcal{CP}e}, \mathcal{R}e \rangle$ composes the data concept part of a **Image Processing Ontology**: a hierarchical structured set of terms to describe image processing results or image processing requests.
$\mathcal{R}e$ represents the set of possible attributed relations between $\mathcal{E}$ and $\mathcal{D}$

- **Definition 3** For each $C_i \in \theta$, we define $\mathcal{LD} : \theta \to \mathcal{D}^n$ so that $\mathcal{LD}(C_i$ is the set of **low level descriptors** related to the visual concepts C_i

- **Definition 4** A low level descriptor $d \in \mathcal{D}$ is modeled as a linguistic variable. It means that each low level descriptor is defined as : $\langle d, L_d, Dom(d), Fuz_d, unit \rangle$
  - d is the name of the linguistic variable
  - $L_d = \{L_d^1, L_d^2, ...\}$ is the set of linguistic values that can be taken by the descriptor
  - $Dom(d)$ defines the domain of the low level descriptor, i.e. its range of possible numerical values
  - $Fuz_d = \{F_d^1, F_d^2, ...\}$ is the set of fuzzy set associated to each linguistic value. A fuzzy set is defined by its membership function
  - $unit$ represent the possible unit of the descriptor which may represent a measurement (may be empty)

- **Definition 4** We define the grounding relation $(G) : \theta \to \mathcal{D}^n$ between the **visual concept** C_i and its related **low level descriptors** $\mathcal{LD}(C_i)$. This grounding relation associate to each low level descriptor $d \in \mathcal{LD}(C_i)$ a numerical range or a linguistic variable.

- **Definition 5** We define $\mathcal{C}_{\mathcal{VC}} = \{Crvc_k\}$ a set of visual data management criteria on visual concepts

- **Definition 6** Let $\mathcal{R} = \{R_j / j \in 1..p\}$ the set of **spatial relations** previously defined.

- **Definition 7** A **Primitive Visual Object** $O = (s_O, M_O, I_O, D_O)$ is defined by a state s_O, a set of attributes with state $M_O = \{m_i / i \in 1..q\}$, a link with the associated data in image I_O (if they exist) and a semantic interpretation D_O (if it exists). The state s_O could be *hypothesized*, *missing*, *perceived*, *partially recognized* and *recognized*. It defines the current state of the visual object in the global semantic interpretation process. An attribute m_i is a 4-tuple $\langle m_i, t_i, h_i, e_i \rangle$ where:
  - m_i is the name of the attribute
  - t_i is the type of the attribute, $t_i \in \theta$
  - e_i is the set of perceived values of the attribute, resulting from measurements on images and completed by the lower level modules
  - h_i is the expected range of values the attribute, resulting from hypotheses of domain classes. $m_{q,i}$ is the notation of the q-th value of the range of the attribute m_i

- **Definition 8** A **Composite Visual Object** $\mathcal{C}(VO_p, R)$ represents a composite visual object, i.e. a constrained configuration of identical (from the semantical point of view) primitive visual objects VO_p. It is composed of :
  - n the number of primitive visual objects (which may be unknown)
  - VO_p the primitive visual object description
  - $\mathcal{C}_g$ a set of constraints on the spatial configuration of the primitive visual objects
    * R is the spatial relation which lies the primitive visual objects
    * $M_{C(VO_p,R)} = \{m_{c_i}/i \in 1..q\}$ the set of attributes: i.e. the description in terms of spatial visual concepts of the complete visual object

5.2.6 Visual Data Management Reasoning

The visual data management engine performs several tasks depending on the state of the interpretation process. It has to:

- Build an image processing request with the specified constraints according to the description of the hypothesized visual objects in terms of visual concepts.
- Select and manage image data to make the correspondence between numerical data coming from the program supervision KBS and the current visual object in analysis (data-driven reasoning).
- Perform spatial reasoning in the case of multiple objects. This reasoning is useful to put in evidence specific geometric arrangements as network, row, circle and to constrain the extraction of data from images.
- Build a symbolic scene description and send it to the semantic interpretation KBS.

The algorithm of the visual data management engine is described in the algorithm 5 described just below. The reasoning of the visual data management engine depends on the type of the request sent by the semantic interpretation system. We make the distinctions between three kinds of requests:

- Primitive Visual Object Request: see algo 5.
- Composite Visual Object Request: see algo 6.
- Visual Scene Request: see algo 7.

Algorithm 5 VisualDataManagementEngine_Primitive(VDM Request r, Image Data List IDList, Perceived Scene Description PSD)

Given a Visual Data Management Knowledge Base VDMKB

1: CurrentVisual Object CVO:=r.VisualObject
2: **if** CVO.state == *hypothetical* **then**
3: IPRequest:=**Program Supervision Request Building**(CVO,*Object extraction*)(using **Visual Object Extraction Criteria**)
4: Sending of IP Request to the Program Supervision KBS
5: Updating of the VDM Fact Base: IDList:= results of the Program Supervision KBS
6: **Evaluation** of the results (using **Visual Evaluation Criteria**)
7: **if** Evaluation is *correct* **then**
8: IPRequest:=**Program Supervision Request Building)** (CVO, *Feature extraction*) (using **visual concepts**)
9: current Image Data :=**ImageDataSelection**(CVO, IDList)
10: **Symbolic Description Generation**(CVO,current Image Data) by **fuzzy matching** (using **visual concepts** and their **grounding relations** to image descriptors)
11: CVO.state:= *perceived*
12: Sending of the symbolic description to the Semantic Interpretation KBS
13: **else**
14: **if** Evaluation is *noisy* **then**
15: IPRequest:=**Program Supervision Request Building**(CVO, *Image Enhancement*)
16: Go to 4
17: **else if** Evaluation is *over-segmentation* **then**
18: IPRequest:=**Program Supervision Request Building** (CVO, *Visual Grouping*)
19: Go to 4
20: **else if** Evaluation is *under-segmentation* **then**
21: IPRequest:=**Program Supervision Request Building**(CVO, *Visual Splitting*)
22: Go to 4
23: **else**
24: CVO.state:=*missing*
25: Sending of the symbolic description to the Semantic Interpretation KBS
26: **end if**
27: **end if**
28: **else if** CVO.state == *perceived* (CVO has an associated Image Data) **then**
29: IPRequest:=**Program Supervision Request Building**(CVO, *Feature extraction*)
30: Go to 10
31: **else**
32: Nothing to done
33: **end if**

Algorithm 6 VisualDataManagementEngine_Composite(VDM Request r, Image Data List IDList, Perceived Scene Description PSD)

Given a Visual Data Management Knowledge Base VDMKB

1: CurrentVisual Object CVO:=r.VisualObject
2: IPRequest:=**Program Supervision Request Building**(CVO.structuralobjectdescription,*Object extraction*)(using **Visual Object Extraction Criteria**)
3: Sending of IP Request to the Program Supervision KBS
4: Updating of the VDA Fact Base: IDList:= results of the Program Supervision KBS
5: **Evaluation** of the results (using **Visual Evaluation Criteria**)
6: **if** Evaluation is *correct* **then**
7: IPRequest:=**Program Supervision Request Building)** (CVO.structuralobjectdescription, *Feature extraction*) (using **visual concepts**)
8: current Image Data :=**ImageDataSelection**(CVO, IDList)
9: IPRequest:=**Program Supervision Request Building**(CVO, *Visual Grouping*) (using **spatial relations** and **spatial structure concepts**)
10: Updating of the VDM Fact Base: IDList:= results of the Program Supervision KBS
11: IPRequest:=**Program Supervision Request Building)** (CVO, *Feature extraction*) (using **visual concepts**)
12: **Symbolic Description Generation**(CVO,current Image Data) by **fuzzy matching** (using **visual concepts** and their **grounding relations** to image descriptors)
13: CVO.state:= *perceived*
14: Sending of the symbolic description to the Semantic Interpretation KBS
15: **else**
16: **if** Evaluation is *noisy* **then**
17: IPRequest:=**Program Supervision Request Building**(CVO, *Image Enhancement*)
18: Go to 4
19: **else if** Evaluation is *over-segmentation* **then**
20: IPRequest:=**Program Supervision Request Building** (CVO, *Visual Grouping*)
21: Go to 4
22: **else if** Evaluation is *under-segmentation* **then**
23: IPRequest:=**Program Supervision Request Building**(CVO, *Visual Splitting*)
24: Go to 4
25: **else**
26: CVO.state:=*missing*
27: Sending of the symbolic description to the Semantic Interpretation KBS
28: **end if**
29: **end if**

Algorithm 7 VisualDataManagementEngine_Scene(VDM Request r, Image Data List IDList, Perceived Scene Description PSD)

```
   CurrentVisual Object CVO:=r.VisualObject.mainvisualobject
1: if CVO.state == hypothetical then
2:    IPRequest:=Program Supervision Request Building( CVO,Object extraction)(using Visual Object Extraction Criteria)
3:    Sending of IP Request to the Program Supervision KBS
4:    Updating of the VDM Fact Base: IDList:= results of the Program Supervision KBS
5:    Evaluation of the results (using Visual Evaluation Criteria)
6:    if Evaluation is correct then
7:       IPRequest:=Program Supervision Request Building (CVO, Feature extraction) (using visual concepts)
8:       IDList :=ImageDataSelection(CVO, IDList)
9:       Symbolic Description Generation(CVO,current Image Data) by fuzzy matching (using visual concepts and their grounding relations to image descriptors)
10:      for Each Spatial Relation of CVO : R(CVO, O_R) do
11:         if O_R.state == Perceived then
12:            Spatial Relation Verification( R, CVO ,O_R) (using Spatial Relations and Spatial deduction criteria)
13:         else
14:            Visual Data Management of O_R (recursive call)
15:            Spatial Relation Verification( R, CVO ,O_R) (using Spatial Relations and Spatial deduction criteria)
16:         end if
17:      end for
18:      Sending of the symbolic description to the Semantic Interpretation KBS
19:   else
20:      if Evaluation is noisy then
21:         IPRequest:=Program Supervision Request Building(CVO, Image Enhancement)
22:         Go to 4
23:      else if Evaluation is over-segmentation then
24:         IPRequest:=Program Supervision Request Building (CVO, Visual Grouping)
25:         Go to 4
26:      else if Evaluation is under-segmentation then
27:         IPRequest:=Program Supervision Request Building(CVO, Visual Splitting)
28:         Go to 4
29:      else
30:         CVO.state:=missing
31:         Sending of the symbolic description to the Semantic Interpretation KBS
32:      end if
33:   end if
34: else if CVO.state == perceived (CVO has an associated Image Data) then
35:    IPRequest:=Program Supervision Request Building( CVO, Feature extraction)
36:    Go to 10
37: else
38:    Nothing to done
39: end if
```

5.2.6.1 Program Supervision Request Building

The phase of **Program Supervision Request Building** consists in the completion of predefined **Program Supervision Requests**. Their are instances of the **IPFunctionalities**. The choice of the image processing functionality depends on the state of the processing. **Object Extraction Criteria** and the description of the hypothesized **visual object** are used to constrain the program supervision request. The result of this phase phase is an instance of a program supervision request with completed values for:

- Functionality (chosen among the set of predefined image processing functionalities);
- Input Data;
- The Visual Content Context.

5.2.6.2 Evaluation of the Image Processing Results

The evaluation phase refers to the fact that images should correspond to real objects. There is no absolute definition of a segmentation being correct. A correct segmentation means that the resulting image reflects the real object correctly. This is beyond the scope of image processing knowledge and requires access to a high level assessment. Evaluation criteria represents knowledge on how to evaluate the program supervision results according to the visual object hypothesis. The phase of evaluation consists in the activation of these criteria.

5.2.6.3 Phase of Symbolic Description Generation

The phase of **Symbolic Description Generation** consists in associating perceived symbolic values to the attributes of a visual object using image data extracted from images and selected for being interpreted.

This phase consists in a fuzzy matching process between **Image Data** and the **Visual Object hypothesis**.

The global fuzzy matching process is shown below in algo 8.

Algorithm 8 Global Fuzzy Matching (Visual Object $\mathcal{O}$, Image Data $\mathcal{I}$)

for Each attribute m_i of $\mathcal{O}$ ($\forall m_i \in \mathcal{M}_O$) **do**
  if The state of m_i is hypothetical **then**
    for Each range value $m_{q,i}$ of the attribute m_i ($\forall m_{q,i} \in h_{m_i}, m_{q,i} \in \theta$) **do**
      Local Fuzzy Matching($m_{q,i}$,$\mathcal{I}$)
      The possibility $\text{poss}_{m_{q,i}}$ of $m_{q,i}$ is the confidence degree conf($\mathcal{I}$,$m_{q,i}$)
      Complete $\mathcal{O}$: a perceived value of m_i is $m_{q,i}$ with the possibility $\text{poss}_{m_{q,i}}$
      Change the state of m_i in *perceived*
    end for
  end if
end for

The overall confidence degree for a visual concept in the algorithm 9 is computed with a fuzzy logic approach: i.e. the minimum of the confidence degrees for all the image descriptors of the grounding link is taken. As a consequence, the overall confidence degree is very sensitive to a descriptor with a low confidence degree. We make the assumptions that the **grounding link** associated to a visual concept represents the necessary conditions for

Algorithm 9 Local Fuzzy Matching (Visual Concept VC, Image Data $\mathcal{I}$)

for Each image descriptor d_j grounding to the visual concept VC ($\forall d_j \in \mathcal{G}(VC)$) **do**
  if The Image Data $\mathcal{I}$ has a value v for the descriptor d_j **then**
    Compute *confidence degree* conf of v with respect to expected value of d_j : L_{d_j} (linguistic value) or R_{d_j} (numerical range of values)
    conf($\mathcal{I}$,d_j) $= \mu_{L_{d_j}}(v)$ or $\mu_{R_{d_j}}(v)$
  else
    conf($\mathcal{I}$,d_j) $= 1$
  end if
  conf($\mathcal{I}$,VC) = Minimum (confidence degree ($\mathcal{I}$,d_j), $\forall d_j \in \mathcal{G}(VC)$)
end for

the existence of the visual concept: i.e. all the descriptors have to exhibit a high confidence degree.

5.2.6.4 Spatial Relation Verification

The task of spatial relation verification is a primordial task in the case of a visual scene request. Indeed, the aim of this task is to use all the knowledge about spatial relations and to use all the pre-existing spatial relations between visual objects to verify hypothesized relations between two visual objects. The principle of this task is presented in the algorithm 10.

5.3 The Image Processing Program Supervision Framework

The definition of the program supervision module is not a contribution of this thesis. Indeed, the ORION team has a great experience in knowledge based systems for image processing program supervision [Thonnat et al., 1998b]. The image processing program supervision module is composed of a conceptual knowledge base model for program supervision, of an existing program supervision engine named PEGASE+ and of the YAKL language, a descriptive knowledge language which provides image processing experts with a user-friendly syntax. All these components are detailed in [Moisan and Thonnat, 2000]. We briefly describe them in the following sections to give a global view of the cognitive vision platform. However, the integration of the program supervision module in the platform and its interoperability with the visual data management module takes part in the contributions of this thesis.

5.3.1 Analysis of the Image Processing Problem

The aim of this module, as component of the cognitive vision platform, is to provide tools for the extraction and the numerical description of image primitives which correspond to the different objects of interest on images. To perform image processing, the most natural choice would be to used a specialized image processing program for each object. However, it is well known that image segmentation is an hard task which is not always reliable. Using a specialized program is not sufficient to cover a wide range of applications and to allow robustness, flexibility, and adaptability. Indeed, the image processing task has to be processed in an intelligent way, i.e. to be able to adapt itself to different image contexts. Moreover, the end user should not be aware of the technical details of image processing. Therefore, the image processing module must provide some level of autonomy. Based on

Algorithm 10 Spatial Relation Verification (Spatial Relation R, Visual Object $\mathcal{O}_1$, Visual Object $\mathcal{O}_2$)

1: Search in the fact base (Perceived Visual Description $\mathcal{PVD}$) for a spatial relation $R^{'}(\mathcal{O}_1, \mathcal{O}_2)$ of the same type of R
2: **if** $R^{'}(\mathcal{O}_1, \mathcal{O}_2)$ exists **then**
3:     **if** $R^{'}(\mathcal{O}_1, \mathcal{O}_2)$ is a sub-relation of $R(\mathcal{O}_1, \mathcal{O}_2)$ **Spatial Relation Specialization Link then**
4:         $R(\mathcal{O}_1, \mathcal{O}_2)$ is true
5:         Add a relational attribute in the description of the visual object
6:     **else if** $R^{'}(\mathcal{O}_1, \mathcal{O}_2)$ is incompatible with R **then**
7:         Add a relational attribute in the description of the visual object with confidence degree 0
8:     **end if**
9: **else**
10:     Search in the fact base for all the spatial relations of $\mathcal{O}_1$
11:     Search in the fact base for all the spatial relations of $\mathcal{O}_2$
12:     Inference of spatial relations between $\mathcal{O}_1$ and $\mathcal{O}_2$ (**using spatial deduction criteria : transitivity and composition criteria**)
13:     Complete the fact base $\mathcal{PVD}$ with the inferred spatial relations
14:     Run from 1 to 8
15:     Search for the set of conditions to verify $R(\mathcal{O}_1, \mathcal{O}_2)$
16:     Test conditions of $R(\mathcal{O}_1, \mathcal{O}_2)$ on image by the **Building of a Program Supervision Request**
17:     **if** The test of conditions is true **then**
18:         Instantiation of $R(\mathcal{O}_1, \mathcal{O}_2)$ and storing in the fact base $\mathcal{PVD}$
19:         Add a relational attribute in the description of the visual object with confidence degree
20:     **else**
21:         Instantiation of the inverse relation of R, $R^{-1}(\mathcal{O}_1, \mathcal{O}_2)$ and storing in the fact base $\mathcal{PVD}$
22:         Add a relational attribute in the description of the visual object with confidence degree 0
23:     **end if**
24: **end if**

good experience in our team [Thonnat et al., 1998b], we use program supervision techniques. As describe in [Moisan et al., 2001], they are good techniques for the semantical integration of image processing programs independently of any domains or library of image processing programs. Indeed, program supervision techniques favor the capitalization of knowledge of the use of complex programs and the operationalization of this utilization for users not specialist in the domain. It offers them help concerning the choice, the parameterization and the sequencing of image processing programs.

Program supervision aims at automating (partly or completely) the configuration of data processing programs, independently of any particular application domain. In our case, it makes the knowledge of how to apply, compose and to repair image processing programs explicit, and thereby help the non-expert on the field of image processing to do efficient image processing. The goal of program supervision is not to optimize the programs themselves but their uses.

5.3.1.1 Overview of the Image Processing Program Supervision System

A knowledge based image processing program supervision system emulates the strategy of an image processing expert in the use of image processing programs. As described in figure 5.30, it is composed of:

- a set of existing image processing programs (image processing program library),
- a knowledge base describing how to use the set of image processing programs,
- a program supervision engine,
- a fact base.

5.3.1.2 The Program Supervision Engine

The program supervision engine (PS engine) is application dependent. The role of the PS engine is to exploit knowledge about programs stored in the knowledge base to produce a plan of programs (that achieves the initial goal), to execute the programs of the plan and to control its execution. It emulates the strategy of an expert in the use of programs.

5.3.1.3 The Program Supervision Knowledge Base

The knowledge base is written by an expert. It depends on the application domain and on the set of programs that is modeled. The main concepts of a knowledge base in program supervision are:

- **Supervision operator** that performs actions and manipulates data: **primitive operator** corresponds to program and **composite operator** corresponds to known combination of operators that solve abstract processing steps.
- **Arguments** are attributes of supervision operators.
- Various **program supervision criteria**, attached to supervision operators, are used to describe decisions during problem solving.
- **data and domain objects** contain all necessary information on the problem of the end user

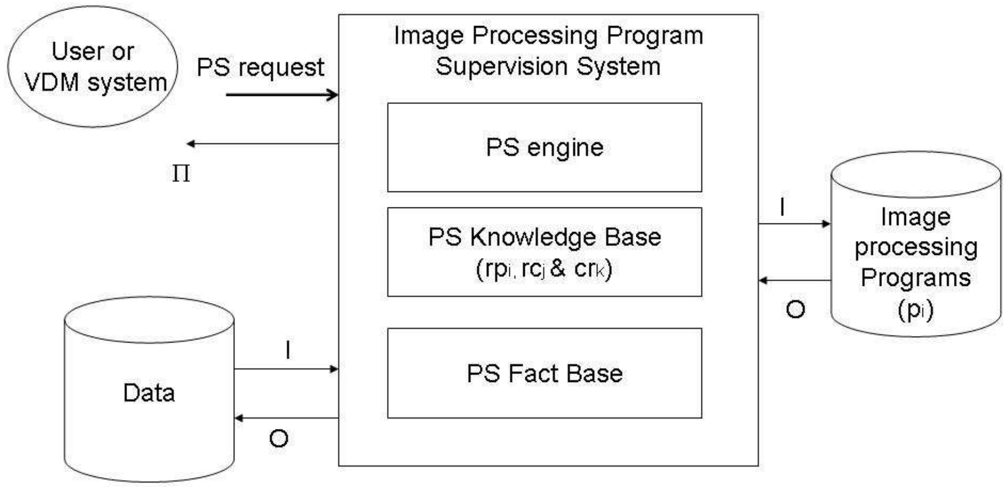


Figure 5.30: A knowledge based program supervision system helps a user to use a set of programs for solving a PS request on input data I to obtain output data O, as the result of the execution plan π. It is composed of a program supervision engine and a knowledge base. The knowledge base contains the rp_i and rc_j representations of programs p_i and combinations of programs c_j, as well as the representations of various decision criteria cr_k

- Abstract **functionality** expresses an objective to achieve.
- A **program supervision request** expresses a query, i.e. a functionality to achieve and the data of the particular case to work on.

5.3.1.4 The Program Supervision Fact Base

The program supervision fact base contains the data instances describing the current problem and all the necessary environmental information. It also contains the data created as result of the execution of operators.

5.3.2 Program Supervision Knowledge Base Model

This section details the main concepts of a knowledge base in program supervision. This approach provides experts with guidance for the program supervision knowledge representation.

5.3.2.1 Supervision Operators

They are two types of supervision operators: **primitive operators** and **composite operators**. They are used to define elements which perform actions and manipulate data.

They are implemented by frames.

1. **Primitive operators** represent particular programs. Their common representation includes:

    - An optional reference to an abstract functionality (e.g. information on what is the operator for).
    - Information on arguments, including their names, types, ranges or means to compute their value.
    - Semantical information: characteristics (known by the expert), constraints, pre and post conditions.
    - Information needed for the effective execution of the program (e.g. calling syntax, simulation method).
    - Various criteria to specify the reasoning which is made on operators: result **evaluation criteria**, argument **initialization criteria** and **adjustment criteria**.

    The general syntax of a primitive operator is given in figure 5.31

Primitive Operator { **name** *a name*
  Functionality *name of a functionality*
  Input Data
    list of input arguments
  Input Parameters
    list of parameters arguments
  Output Data
    list of output arguments
    I-O relations
  Preconditions
    list of conditions that must be verified by input data
  Postconditions
    list of conditions that must be verified by output data
  Initialization Criteria
    list of optional initialization criteria
  Assessment Criteria
    list of optional assessment criteria
  Adjustment Criteria
    list of optional adjustment criteria
  Call
    language shell
    syntax ... endsyntax }

Figure 5.31: General syntax of primitive operator in the program supervision knowledge base

2. **Composite operators** represent particular combinations of programs. They are skeletons of plans provided by the expert. They describe the network of known possible connections between operators (choice, sequence, entailment, repetition,...) in order to achieve a given goal. In addition to the common information of primitive operators, a composite operator is composed of:

   - Control information about the type of decomposition into sub-operators;
   - References to the sub-operators by their names;
   - Data flow information between father and sons and between sons in a sequential decomposition;
   - Additional criteria: **choice criteria** and **repair criteria**.

   The general syntax of a composite operator is given in figure 5.32

Composite Operator { **name** *a name*
  Functionality *name of a functionality*
  Input Data
    list of input arguments
  Input Parameters
    list of parameters arguments
  Output Data
    list of output arguments
    I-O relations
  Preconditions
    list of conditions that must be verified by input data
  Postconditions
    list of conditions that must be verified by output data
  Body
    decomposition in sub-operators
  Choice Criteria
    list of choice criteria
  Initialization Criteria
    list of optional initialization criteria
  Assessment Criteria
    list of optional assessment criteria
  Adjustment Criteria
    list of optional adjustment criteria
  Distribution
    information on data flows between father and sons
  Flow
    information on data flows between sons }

Figure 5.32: General syntax of composite operators in the program supervision knowledge base

5.3.2.2 Arguments

Arguments are associated with supervision operators and with functionalities. From the program supervision point of view, they play an important role because many decisions are based on the information that arguments provide. From the point of view of the interoperability with the visual data management module, they also play an important role. Indeed many of the arguments correspond to **image data** and **descriptor set** concepts of the visual data management module. The shared concepts correspond to data concepts of the image processing ontology.

5.3.2.3 Program Supervision Criteria

Various program supervision criteria, implemented by rules, play an important role during the reasoning. They are attached to operators.

- **Choice criteria**, attached to composite operators, enable to choose between different alternative operators having the same functionality.
- **Initialization criteria** contain information on how to initialize values of input arguments, before executing the current operator.
- **Evaluation/Assessment criteria** state information on how to assess the quality of the selected operator's actual results after its execution. Evaluation criteria enable to detect and diagnose a problem. It concerns the evaluation from the image processing point of view. In our framework, the evaluation of the image processing results according to the high level goal is performed by the visual data management system.
- **Adjustment criteria** propose a way to tune a parameter value to improve the quality of the results of an operator.
- **Repair criteria** define a repair strategy after a failure decided by evaluation rules.

5.3.2.4 Data and Domain Objects

Data correspond to descriptions of the data used by the programs. **Domain objects** correspond to specific application domain concepts that may influence the program supervision context. Data and domain objects are stored in the fact base. During the reasoning, they may be modified or added (results of the execution of operators).

Data and domain objects are two important concepts for the interoperability with the visual data management module. Data are the data that will be used to instantiate the corresponding **image data** of the visual data management module. Domain objects corresponds to concepts of the visual data management module. For example, the **visual content context** concept of the visual data management module corresponds to a domain object in the program supervision module.

5.3.2.5 Program Supervision Requests and Functionalities

A functionality enables to group together all the supervision operators achieving the same abstract functionality. To enable the interoperability with the visual data management module, several functionalities are provided by the image processing ontology. They correspond to the **image processing functionalities** of the visual data management module.

Program supervision requests are queries for an abstract functionality on particular data under particular constraints. They are built by the visual data management module as described in section 5.2.3.8.

5.3.3 Program Supervision Reasoning

We have reused the PEGASE+ engine, dedicated to program supervision knowledge based systems. It can automate the choice and execution of programs from a library to accomplish a processing objective.

The different phases of the program supervision engine are described on figure 5.33. The initial *planning* phase determines the best strategy to reach the end user goal (step 1). Then the *execution* phase launches the individual programs in the plan (step 2). An *evaluation* phase assesses the quality and contents of the resulting data (step 3). If the results are correct, planning can continue (step 4). If the results are incorrect (step 5), a *repair phase* can modify the plan (step 6) or re-execute the procedure with different parameters (step 7).

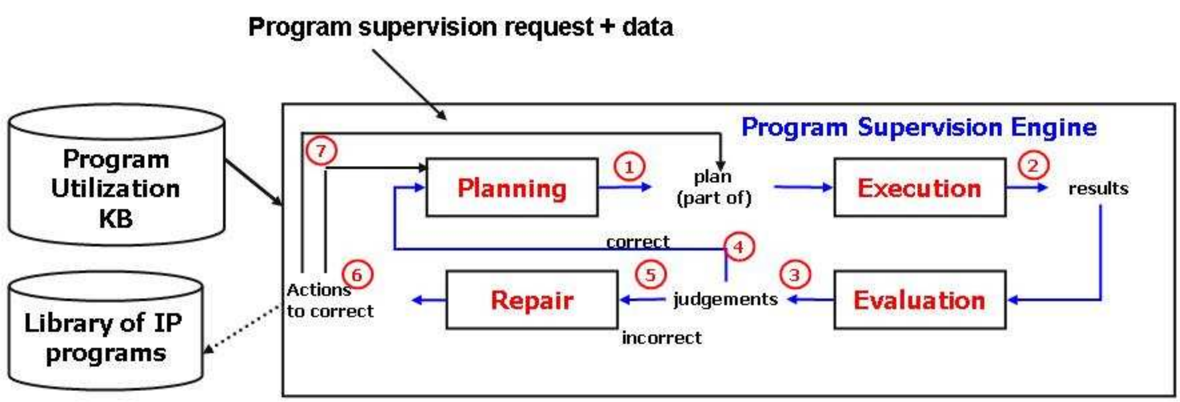


Figure 5.33: The different phases of a program supervision system

The PEGASE+ engine provides a HTN (Hierarchical Task Network) planner, an execution mode, some evaluation facilities and a repair mechanism using repair and adjustment criteria. The PEGASE+ engine has been tested on different applications

-1- global matching
While not all PS requests have been processed
-2- selection of the PS request to be processed
-3- classification of the operators ***(using choice criteria)***
while the PS request is not satisfied
-4- selection of the best operator
-5- execution of the operator ***(using initialization or adjustment criteria)***
*-6- assessment of the results (**using evaluation criteria**)*
While assessment is no satisfied
*-7- reparation of the operator (**using adjustment and repair criteria**)*

Figure 5.34: The Program Supervision Engine algorithm

[Thonnat et al., 1998b]: for medical imaging [Crubézy et al., 1997] or for satellites images [Mathieu-Marni et al., 1995]. The program supervision algorithm is described in figure 5.34.

5.4 Implementation of the Cognitive Vision Platform

We have implemented the proposed cognitive vision platform with the LAMA environment [Moisan, 1998]. The LAMA environment is a software platform devoted to the generation of knowledge based systems, i.e. knowledge base and inference engine design. It offers toolkits to build and to adapt all the software elements that compose a knowledge based system (inference engines, interfaces, knowledge based description languages, verification tools, ...). The platform both allows to design program supervision and automatic object recognition knowledge based systems and it facilitates the coupling between the two types of systems.

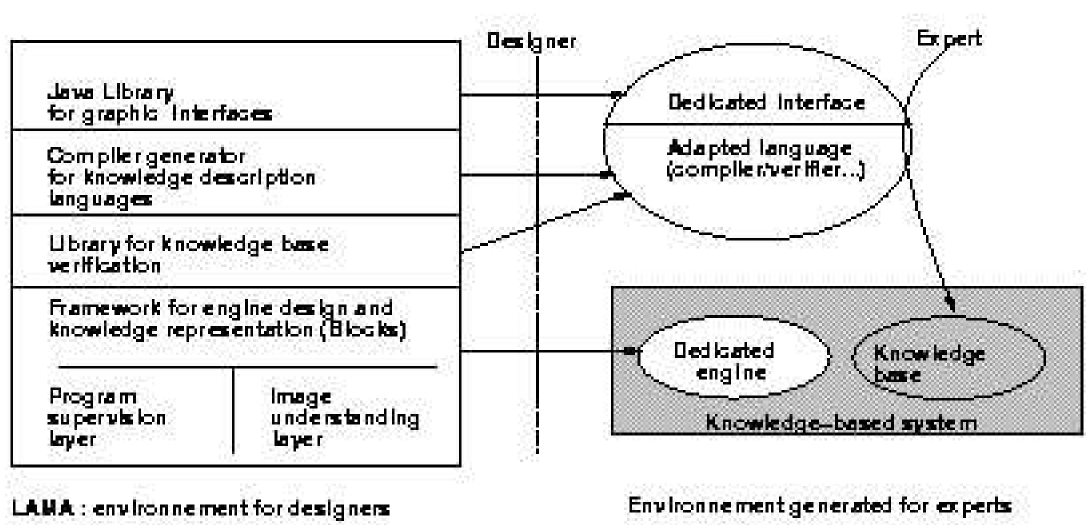


Figure 5.35: LAMA architecture and tools for engine design, knowledge base description, verification, and visualization

The global architecture of the LAMA platform is described in figure 5.35. The core of the platform is a framework of reusable components, called BLOCKS. It provides designers with a software framework: i.e. reusable and adaptable components that implement generic data structures and methods for the support of a particular problem. It also supplies the knowledge concepts of a particular task (e.g. program supervision) to build knowledge bases. Dedicated description languages that operationalize the conceptual models of knowledge can be developed. The LAMA platform provides a toolkit to design knowledge base editors and parsers, a knowledge base verification toolkit (adapted to the engine in use), a toolkit to develop graphical interfaces (both to visualize the contents of a knowledge base and to run the solving of a problem).

The LAMA platform is generic and customizable. To implement the cognitive vision platform:

- We have reused the PEGASE+ engine and the knowledge concepts of the program supervision task. It composes the program supervision module of the proposed cognitive vision platform.

- We have adapted an existing engine named TACLE dedicated to the object recognition problem to our semantic interpretation problem. The TACLE engine was based on a classification process: the classification of an unknown object using the taxonomy of classes. The new engine called TACLE++ takes as input a domain request, can manage multiple objects and can build hypotheses through visual data management requests.

  The knowledge concepts associated to TACLE were modified. The concept of *Class* was modified into *Domain Class* to cope with the description by visual concepts and by spatial relations. Context criteria were derived from existing concepts to represent rules. We have introduced some additional concepts: e.g. domain requests, domain context and domain acquisition.

- We have completely built by using BLOCKS an engine called VISDATMAN dedicated to visual data management and the conceptual knowledge model associated to it.

In the next chapter, we show how we have used the cognitive vision platform to build a semantic image interpretation system dedicated to rose disease diagnosis.

Chapter 6

Application: A Semantic Image Interpretation System for the Recognition of Biological Organisms

This chapter is dedicated to the validation of the cognitive vision platform by the building of a semantic interpretation system for the recognition of biological organisms. In particular, we are interested in the recognition of greenhouse rose leaf diseases. This application is a real world problem. We first present the biological problem and show how semantic image interpretation can solve it. Then we give a brief overview of the state of the art on the automatic recognition of biological organisms using image analysis and artificial intelligence techniques. The last section is dedicated to the detailed description of the rose disease interpretation system built with the cognitive vision platform.

6.1 A Major Biological Problem: the Early Detection of Plant Pathologies

6.1.1 A Major Challenge for Integrated Pest Management

Integrated Pest Management (IPM) is a knowledge based approach to crop protection. It is an important tool for the management of insects, pathogens, weeds and cultural problems in greenhouse in an economically and ecologically sound way. IPM involves the integration of cultural, physical, biological and chemical practices to grow crops with minimal use of pesticides.

In particular, early detection of plant diseases and plant health monitoring are keys for pest and disease management. Indeed, they can provide more accurate forecasts of diseases and make it possible to operate efficiently at the beginning of an infection to limit the plant damage. Thereby, they can reduce the amount of pesticide applications and thus reduce the control cost. Moreover, the early detection of plant disease enables the use of natural enemies when the threat of the damage is not imminent. Plant health monitoring is still carried out by humans. Therefore, there is a great interest in automating the monitoring with the goals of :

1. a more **accurate** (not subjective, not dependent on the human experience) and **earlier** (not limited to the human naked eye) diagnosis,

2. the capitalization of pathological knowledge.

6.1.2 A Semantic Image Interpretation Problem

Plant disease diagnosis is a **visual action** which aims at **inferring** the presence of plant diseases by the visual detection and analysis of disease **signs** and **symptoms**. **Symptoms** are observable changes which result from deviation from the normal physiological or morphological development of the plant. **Signs** are observable structures of the pathogen.

Due to the visual nature of the plant diagnosis, computer vision techniques seems to be well adapted for the automation of the perception task. Moreover, plant diagnosis involved reasoning strategies to infer the presence of the disease by the interpretation of signs and symptoms. To mimic the pathologist, the explicitation of pathological knowledge is needed and the system has to be able to see (extraction of the relevant information) and to reason (using the pathological knowledge) as illustrated in figure 6.1. Semantic image interpretation techniques seem to be a good way to automatically achieve the task of early plant diagnosis.

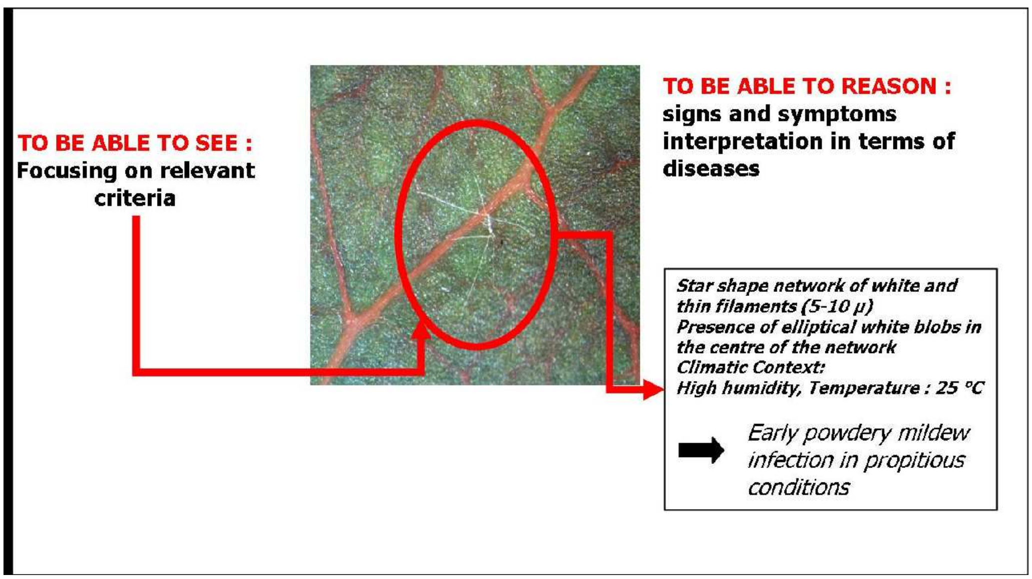


Figure 6.1: Illustration of an action of plant diagnosis. The image is a microscopic image of a rose leaf with an infection of powdery mildew. The mycelium (thin white lines) is about 10 micronmeters large.

Whatever is the targeted crop, plant disease diagnosis involves the recognition of various pathogens which can take various appearances according to their stage of evolution and to their vegetable support. Moreover, in greenhouse, various contexts have to be managed according to several external factors (season, climatic conditions, ...). The context can also change quickly in a unforeseeable manner (illumination changing,...). At last, several symptoms can occur in the same time on the same organs and the management of multiple object and spatial reasoning is needed. Therefore, plant disease diagnosis is a complex problem of semantic image interpretation and is a perfect real world application to validate our cognitive vision platform. The figure 6.2 illustrates the main issues of the rose disease application.

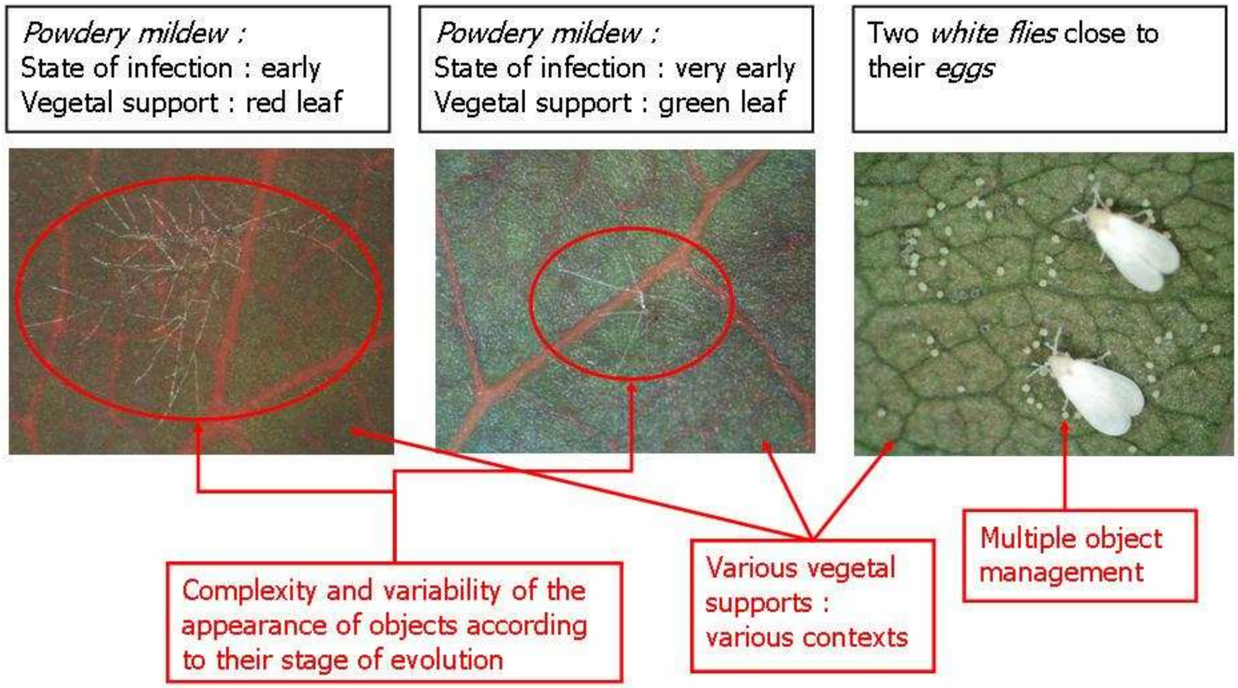


Figure 6.2: A complex problem of semantic image interpretation: illustration of the different issues. White flies are about 2 mm long. Powdery mildew is about 10 micronmeter large

6.1.3 State of the Art on Semantic Interpretation for Biological Purposes

In this section, we present some related works. These works are involved in the recognition of biological organisms. We review some works using image interpretation methodologies (in fact merely image analysis methodologies for the majority of these works) in crop production applications.

6.1.3.1 Image Interpretation Techniques to Manage Crop Production

In [Ehret et al., 2001], machine vision is presented as one of the current trends to monitor the crop status directly and to automate the plant health monitoring. Applications are numerous and various: fruit quality inspection system, plant growth monitoring, plant stress monitoring and automatic diagnosis of plant diseases.

1. **Crop sorting and crop quality grading**
   Machine vision techniques are widely used for crop sorting and crop quality grading [Jahns et al., 2001], [Urena et al., 2001], [Guyer et al., 1996],

[Shahin and Symons, 2001], [Steinmetz et al., 1994], [Unay and Gosselin, 2004]. Machine vision techniques are beneficial to crop quality grading because they enable: (1)to increase the speed of sorting, (2) to eliminate the human error and (3) to perform the grading without contact with the crop. Most of the techniques are based on the external visual appearance of the crops. The properties of quality are computed using image features like the size, the color and the shape of the crop. All the works on crop quality grading show good performances. Nevertheless, a general statement is their dependence on the application and on the concerned crop.

2. **Weed detection and characterization**
   As explained in [S. Christensen and Walter, 1996], information on weed spatial distribution and weed characterization is necessary to implement variable herbicide application and reduce their application. This task of weed detection and characterization is a labor-intensive task for a human operator. Therefore, many research studies have focused on weed detection and characterization using machine vision techniques. General approaches are either based on the geometric differences between the crop and the weeds (as the leaf shape) [Tian et al., 2000] or on their spectral reflectance differences (color indexes) [El-Faki et al., 2000]. Some other approaches used color, shape and/or texture features [Perez et al., 1997], [Zhang and chaisattapagon, 1995]. The location of the crop compared to the weed is used in [JunWei et al., 2000].

   In recent works [Mezzo et al., 2003] and [Manh et al., 2001], the authors emphasize remaining difficulties in the weed detection process due to the complexity of outdoor scenes and due to the variability of appearance of plants. They propose to introduce a priori knowledge about the shape of the weed leaf to enhance the weed leaf segmentation process. In particular, the method presented in [Mezzo et al., 2003] enables to manage the case of occluded leaves.

   All these works are interesting and give correct results even in outdoor conditions with variations of illuminations. Nevertheless a general comment is that most of these methods are application specific (for example detection of onion weeds in [JunWei et al., 2000]). Only few works are interested in the genericity of the method. Exceptions are [Mezzo et al., 2003], [Manh et al., 2001] but the validation of their method was performed only on few sequences.

3. **Recognition of biological organisms**

   - **Recognition of spores**
     In [Bernier and Landry, 2000], a method for the recognition of pathogenic fungal spore is proposed. This research takes part in a large-scale research program ultimately aimed at reducing fungicide application. The authors address the problem of early disease detection by the automatic detection and counting of fungal spores. The main issue is that spores appear as irregular objects, in multiple physical orientation and they may be obscured and occluded by other objects. They propose a robust algorithm based on knowledge about the visual appearance of spores. It enables the detection of spores even in occlusion cases but not the identification of spore species.

     Some similar works about the recognition of airborne fungi spores with the motivation of the monitoring of biological working substances which have an influence to human health can be found in [Perner et al., 2003b] and [Benyon et al., 1999].

- **Recognition of filamentous biological organisms**
  There are several works dealing with the image analysis of filamentous organisms [Pons and Vivier, 1999], [Paul and Thomas, 1998], [T. Dorge and Frisvad, 2001]. Their aim is the biomass quantification of microorganisms with various biotechnological purposes: nutritional or pharmaceutical substance production, wastewater treatment. All these works are based on the morphology of filamentous fungi [Kossen, 1989]. Various forms of filamentous fungi and their shape descriptors have been identified : *almost spherical ungerminated spores, germinating spores with one or more germs tubes of different lengths, filaments with various degree of branching called **flocks** and entanglement of one or more filaments called **pellets***. These different forms of filamentous fungi correspond to different growing stages of the fungi. They are represented on figure 6.3.

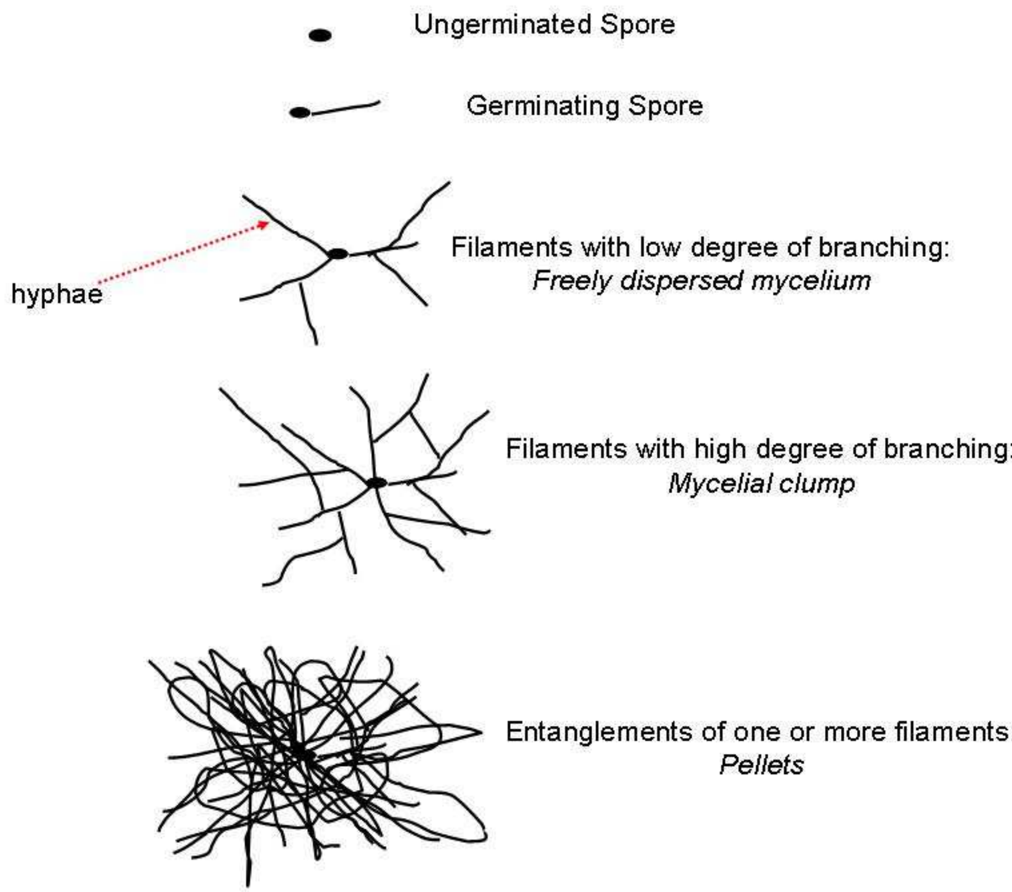


Figure 6.3: The different growing stages of a fungi

4. **Automatic diagnosis of plant diseases**
   Knowledge based system techniques were identified as efficient tools to manage the automation of plant disease diagnosis. They enable to combine the experimental and experiential knowledge with the intuitive reasoning skills of a multitude of pathologists. There are many knowledge based systems for plant pathologies

[El-Azhary et al., 2000], [Mahaman et al., 2003], [Nunez et al., 1996]. All these conventional knowledge based systems are based on textual input. They depend on the end user to understand abnormal symptoms on the plant. To reduce this weakness, in [El-Helly et al., 2004], a generic approach based on a knowledge based vision system is presented. They propose a generic diagnostic model integrated with an image analyzer. The aim of the image analyzer is to extract the abnormal symptoms and to classify these symptoms to a hypothesized disease through four generic steps : (1) image enhancement to make easier the segmentation, (2) the segmentation step, (3) the feature extraction step (4) a classification step. Then a differentiation phase consists in the confirmation or the rejection of the hypothesis using additional observations and a causal model. The proposed method was applied on a real world example: the diagnosis of cucumber diseases using leaf images. In this system, only the diagnosis model is generic.

6.2 A Semantic Image Interpretation System for the Recognition of Rose Diseases : ROSESIM

6.2.1 Context

This part of the work consists in a research cooperation between the Orion team of INRIA Sophia Antipolis (French National Institute for the Research in Computer Science and Control) and the Integrated Research in Horticulture Unit (URIH) of INRA Sophia Antipolis (National Institute for Agricultural Research). Our work takes place in a large-scale and multidisciplinary research program ultimately aimed at reducing pesticide application: *"Action Transversale INRA : Protection Integrée des Cultures 2000-2003. Production Intégrée sous serre lourde. Application sur Tomate et Rosier"*.

Moreover, the context of this work is also the region Provence Alpes Côte d'Azur (PACA). The region PACA is the leadind horticultural region of France. It represents a quarter of the national horticulture turnover. Roses are widely produced in PACA. Early disease detection was classified as a major challenge. Our work has received funding from the region PACA.

6.2.2 Main Targeted Rose Diseases

There is a considerable amount of rose diseases. The disease management is complicated by the presence of multiple types of pathogens: fungi, virus, bacteria, pests and nematodes.

We are not interested in building an exhaustive knowledge based system of rose diseases. The building of ROSESIM is motivated by both the biological goal of the application: i.e. the early detection problem and the context of the application: i.e. greenhouse roses.

In particular, the context of the application is greenhouse rose production in the Mediterranean Basin. Therefore, the study is limited to rose diseases which can appear in such conditions. Moreover, we have limited our system to foliar greenhouse diseases.

Moreover, the major motivation of the end users is the early detection and recognition of the major and most damaging rose diseases : powdery mildew and pests. As we are interested in the early detection of disease, we are first interested in the signs of the disease and in early symptoms. The symptoms visible to the naked eye are not taken into account. We are not interested in severe damage on leaves. In particular we will focus on the following diseases:

1. **Powdery Mildew**
   Powdery Mildew is a fungal disease. The pathogen responsible for powdery mildew is *Sphaerotheca pannosa*. It is one of the most widespread and destructive diseases of roses in the world. Powdery mildew fungi belongs to the group of plant pathogens called obligate parasites. It means that it can only grow and reproduce on a living host plant. It can infect any green tissue of roses. It first appears on new foliage but it can also be found on green stems and flower parts.

   - **Life cycle**
     Powdery mildew is always present in greenhouses. It survives during the winter months as **mycelium** in protected leaves of buds or in inner bud scales. The first infection on new growth arises from the previous year infection. Spores, called **Conidia**, produced by the surviving **mycelium** are spread to healthy leaves, drifting by the wind or carried by greenhouse human agents. When the right conditions (high day and low night temperatures with high humidity) to germinate are present, the conidia germinates by pushing out a **germ tube** across the leaf surface. As many fungi, it grows as tubular filaments called **hyphae**. The mass of branched filaments is the **mycelium**. Conidiophores are produced vertically from the mycelium and they bear chains of conidia. These conidia are carried by the wind or other means to healthy rose tissues. A new disease cycle is initiated. The life cycle of powdery mildew is depicted in figure 6.4. These powdery mildew early signs (**conidia**, **mycelium**, **hyphae**, **conidiophore**) are not visible to the naked eye. To observe them, microscopic device is needed.

     Powdery mildew can spread quickly since the disease cycle can be completed in as little as 72 hours. It commonly takes 7 to 10 days from the time of infection to the development of symptoms and secondary spore production.
   - **Main symptoms**
     First symptoms of the powdery mildew are irregular, light green to reddish areas on the upper surface of young leaves. Then, powdery grayish-white spots appears on the upper surface of the leaf. These spots represent the dense and concentrated growth of the fungi into mycelium, conidiophores and conidia. The powdery aspect results from the chains of conidia. Infected young leaves become curled or irregularly twisted. When the infection is severe the young leaves turn to yellow and may drop prematurely. Older leaves are not distorted but they develop round to irregular white areas.

2. **Pests**

   (a) **Greenhouse White Fly**
   Greenhouse white flies or *Trialeurodes vaporarium* are common pests of greenhouse plants. This insect has a host range of more than 250 ornamental and vegetable plants: rose, poinsettia, begonia, nicotiana, aster, calendula, cucumber, lantana, tomato, grape, ageratum, bean and hibiscus are commonly infested.

   It exists on the upper and lower surfaces on leaves under different stages of development (e.g.. figure 6.5).

   - **Stages of development**
     - **Adults** (A on figure 6.5) are small, winged, white insects about 1.5-2 mm long. They have four wings and a yellowish body. The wings have

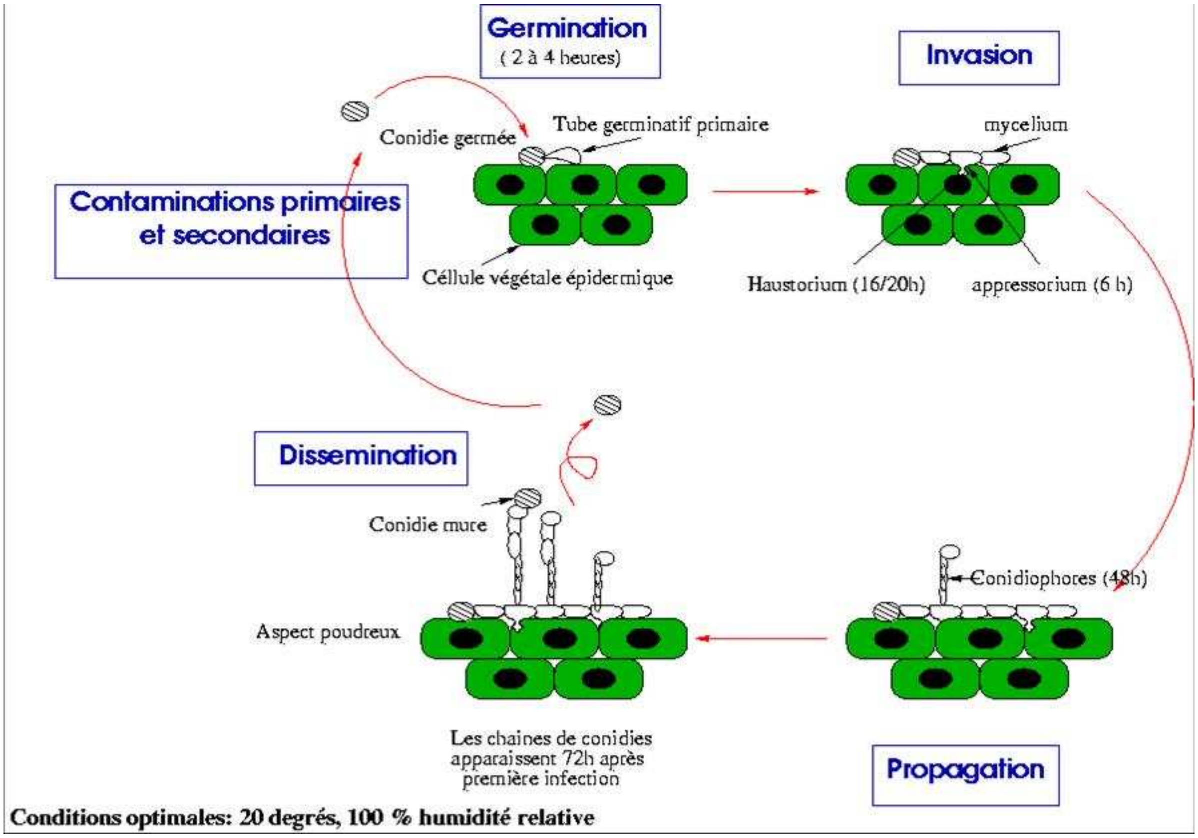


Figure 6.4: The leaf cycle of a powdery mildew infection (in french)

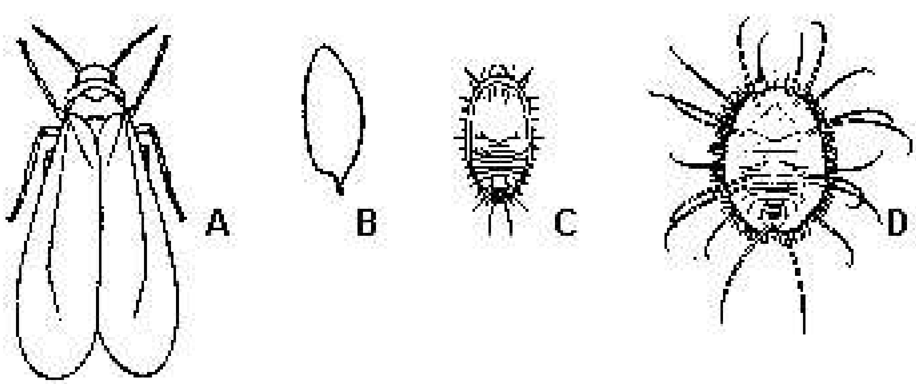


Figure 6.5: The different stages of development of the greenhouse white fly. From http://www.entomology.umn.edu/cues/inter/inmine/Whiteff.html

the appearance of having been dusted with a very fine white material. As the wings covered the body and are more or less parallel to the leaf surface, an adult white fly appears as rounded and triangular in shape. Newly emerged adults may be pale green or yellow, but soon become the more familiar white. An adult white fly has four or possibly five leg segments and two to three antenna segments. Nevertheless, most specimens appear to have only three leg segments and two antenna segments.

- **Eggs** (B on figure 6.5) are elliptical oblong shaped and white to yellow when they are young. They turn to gray after two days. They are laid in a circle or a crescent in the lower leaf surface. Each egg is about 0.2 mm long.
- **Nymphs**(C on figure 6.5) or **crawler** are usually pale green and scale-like in appearance. They possess antennas and have hair-like projections along the periphery of their bodies.
- **Pupae**(D on figure 6.5) are elliptical shaped, pale green (normal) or black (parasitized) and about 0.75 mm long insect. The periphery of the insect is surrounded by a fringe of white waxy hair-like filaments or appendages. Several pairs of filaments may project up from the upper surface of the pupa. After the adult has emerged, a white, almost transparent pupal skin is left behind.

- **Life cycle**
  The life cycle and number of eggs laid by the greenhouse white fly vary with the temperature and the species of the host plant. Greenhouse white flies reproduce relatively slowly (one generation every 30 to 45 days), but each female lays about 250 eggs and lives as long as two months. Adults usually are found on the lower surface of new leaves. They lay their eggs which hatch 5 to 7 days later. New nymphs move about the plant for a day or two, often from leaf to leaf, before inserting their mouth-parts to feed. Once this occurs, they probably do not move again until mature. The crawlers molt into later nymphal stages and then into pupae. Finally, a new generation of whitish-yellow adults emerges. They are covered with a white waxy bloom.
- **Main symptoms**
  The presence of white flies in greenhouse may cause severe damages. The main sign to detect the presence on white fly is the detection of the different stages of development of the insect. The main symptoms result from the white fly feeding. Spotting and chlorosis appear on the foliage. Plants infested with white flies lack vigor, wilt, turn yellow, and may die. In addition, heavily infested plants are coated with a sticky material called honeydew which reduces the attractiveness and scalability of the plant. Heavy concentrations of honeydew will promote the growth of a black sooty mold which interferes with photo-synthesis. Sooty mold may also interfere with production or harvest operations. Most important, some species of whiteflies are known to vector plant viruses.

(b) **Aphid**
Several different species of aphids attack roses. We are interested in the main species called *Macrosiphum rosae*. They are usually more numerous in late spring/early summer, although populations might also resurged in autumn. Aphids or plant lice are small, soft-bodied, sluggish insects that cluster in colonies on the leaves and stems of the host plants. They are sucking insects that insert their beaks into a leaf or stem to extract plant sap. They are usually found on and under the youngest leaves, and, in general, prefer to feed on tender, young growth.

- **Stages of development**
  - **Adults** (figure 6.6) are generally 1-5mm long with a soft body, long legs and antennae, and they usually have a prominent pair of tube-like structures at the end of the abdomen. These tube-like structures are called cornicles. Adults are rarely isolated but are found in colonies.

Figure 6.6: Picture of an aphid

  - **Nymphs** look like wing-less adults. They are smaller than adults. Both green and pink forms occur in the nymphal stages
- **Life Cycle**
  Many species of aphid have more than one host plant, and will over-winter on different plant species from the ones they live on in the summer. Life cycles are generally complex and many generations are produced each year. They usually survive the winter at the egg stage, although in mild weather the adults can survive and remain active. In spring, populations rise rapidly as female aphids are able to give birth to live young as well as laying eggs, resulting in a rapid rate of reproduction as the temperature increases.
- **Main Symptoms**
  The first sign is the detection of the presence of aphids. As a group, aphids attack a plant by sucking its fluids from tender new growth. They are attracted to the concentrated nitrogen in these new growth areas. The results are deformed leaves and new bloom stems.
  In addition, they also exudate a substance called "honeydew" - a sweet, syrup-like material which appears on leaves and stems. It is a food source for many insects, both pest and beneficial. Sooty mold is a fungus which grows in the honeydew. It makes the leaves look dirty and black.

(c) **Spider mite**
Spider mites are small arachnids related to spiders. They are common pests on greenhouse roses. The most common spider mite on roses is the **two-spotted spider mite** called *Tetranychus urticae*. Tetranychus urticae is a serious pest of a variety of agricultural crops and ornamental plants, about 180 different plant species. They are members of the Order *Acarina*, which includes spiders, ticks and mites. Whereas insects have three pairs of legs and three body regions (head, thorax, abdomen), spider mites have 4 pairs of legs and one body region.

- **Stages of development**
  - **Adults** are elliptical in shape and may be brown or orange-red. The female is about 0.5 mm long. The male is smaller and slender, about 0.3 mm long. They have two typical dark spots on the back and they have four pair of legs.

  - **Eggs** are spherical in shape and less than 0.1 mm in diameter. They are translucent. They are usually near the leaf veins
  - **Larva** are more or less identical to adults but of reduced size. They have only three pair of legs.

- **Life Cycle**
  The typical life cycle of spider mite is composed of the egg, the larva, two nymphal stages and the adult. Under optimum conditions, the development takes five to twenty days. The adult female lives two to four weeks and is capable of laying several hundred eggs during its life. Its development is optimal between 23 and 30 degree Celsius and at a relative humidity of less than 50% .

- **Main Symptoms**
  Direct damage is due to feeding punctures. Early symptoms appear on the upper surface of leaves as pale-colored spots. If attacks are heavy, the plant may die.

6.2.3 Data Acquisition

A first step in the design of ROSESIM was the definition of an image acquisition system. Two aspects should be discussed: the type of sensor and the device for image acquisition (in-situ non destructive or in situ destructive). The main requirements to define the acquisition system were: (1) the ability to detect early symptoms (not visible by the human naked eye), (2) a good image quality to be able to process and to interpret it, (3) the ability to shift the plant organ. Several sensors were tested and different experiments in greenhouses, in laboratory were conducted to define the acquisition process [Hudelot, 2002]. We define an acquisition system including: (1) a binocular microscope (Olympus SZH) equipped with a DP PLAN 1X lens (magnifying range from 7.5x to 64x), (2) a mono CCD color camera (Olympus DP 10, 1280*1024 pix, 8*3 bits), (3) and a manual microscopic positioning device to shift the plant organ.

The data we are working on are 2D microscopic images of rose leaves. These leaves were collected on three rose cultivars produced in a plastic twin-tunnel greenhouse (with climate and fertirrigated automatic control). The process of collection of the leaves were defined according to current sampling procedures in greenhouse for diagnosis purpose. To respect the life cycle of the disease, the data acquisition was performed regularly (two times a week) during long periods. Each acquired image was annotated with domain contextual and acquisition contextual information. This domain contextual information includes the climatic conditions of the greenhouse, the location of the plant in the greenhouse, the location of the organ on the plant, the date and the season of the sampling, the name of the rose cultivars. The acquisition contextual information includes the type of the sensor used to acquire data (in our case, we used only one sensor), the magnification rate, the resolution of the sensor.

The greenhouse simulates real conditions for the integrated production of roses. As a consequence, the presence of pathogens on the collected image was dependent on some external factors. In particular, it was dependent on the treatments in the greenhouse. We were not able to collect a significant amount of images for all the stages of the pest or pathogens previously presented. In agreement with the pathologists, we have focused on the early stages of powdery mildew and on the recognition of pests.

6.2.4 The ROSESIM Interpretation Knowledge Base

The ROSESIM interpretation knowledge base contains knowledge of early signs and symptoms of the targeted diseases described previously. We are interested in the association between organs and symptoms. This association is called a *syndrome*. The knowledge base contains a hierarchy of specialization of leaf syndromes of roses. We consider as early all the signs and symptoms which are not visible to the naked eye. Symptoms which correspond to severe damages are not included in the knowledge base.

6.2.4.1 Pathological Knowledge Acquisition

The domain knowledge acquisition was performed according to the ontology guided acquisition process described in the chapter 4. The pathological knowledge acquisition was performed in different stages. First, several interviews of experts in rose pathology were performed. In particular, Marc Bardin and Philippe Nicot, fungi experts, researchers in the INRA Avignon (France) and Roger Boll, expert in pests in the INRA Sophia Antipolis were interviewed. Two dedicated acquisition tools designed in the Orion team were used during this step : Annotate (c.f. figure 6.7) and Ontovis (c.f. figure 6.8 [Maillot et al., 2003a]). This step was necessary due to the lack of structured and formalized expertise concerning the early symptoms of rose diseases. This work enabled us to identify the important biological concepts for the recognition of early symptoms. A taxonomy of rose leaf early symptoms was built.

Then, the domain knowledge base was built by using the **visual concept ontology** and the previous taxonomy. Each domain class was described by visual concepts with respect to conceptual knowledge base model of the semantic interpretation module. Pathologist experts can used the high level language SIKL++ to describe concepts of their domain in a natural way.

To tackle the problem of the early diagnosis of greenhouse rose diseases, a domain knowledge base which consists in 45 domain classes and 10 context criteria was built.

6.2.4.2 Hierarchy of Domain Classes

The main part of the hierarchy of domain classes is depicted on figure 6.10. This figure shows the taxonomy of domain classes. Sub-part domain classes and corresponding subpart taxonomies are represented in colored bounding boxes.

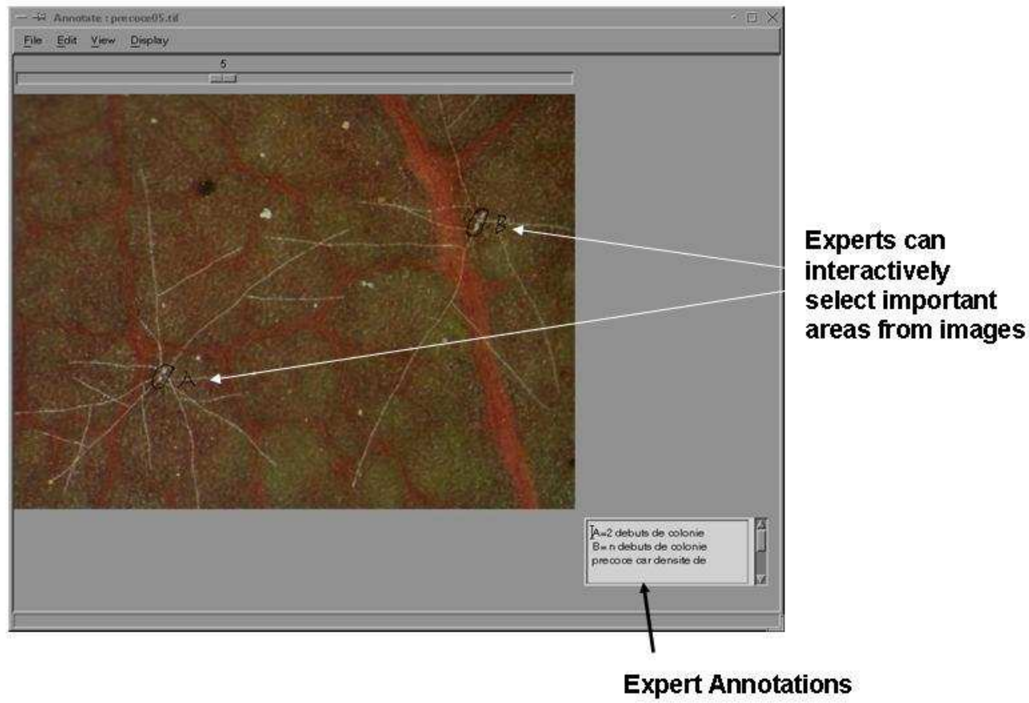


Figure 6.7: The tool Annotate to help the acquisition process

The typical representation of a domain class (in this example, the domain class HYPHAE) according to the representation formalism described in chapter 5 is :

6.2.4.3 Domain Context and Acquisition Context

At the interpretation level, the contextual information refers to the domain application context and the acquisition context. In the case of the rose disease diagnosis application, the domain contextual information represents all the non visual additional biological information which helps for the diagnosis decision process. This biological information represents the different risk factors. They are:

- The relative humidity rate;
- The mean temperature of the greenhouse;
- The season of the sampling;
- The location of the rose plant in the greenhouse;
- The location of the organ on the plant;

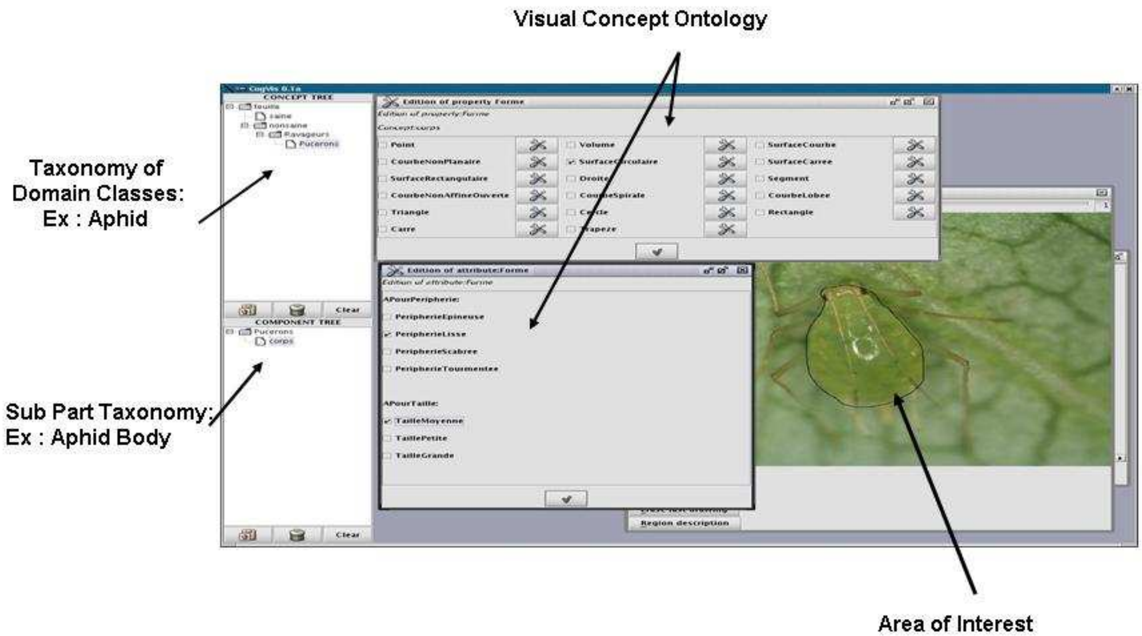


Figure 6.8: The Acquisition knowledge tool Ontovis. The knowledge acquisition is guided by a visual concept ontology

- The variety of the rose;
- The age of the organ.

The ROSESIM domain context is depicted in figure 6.11

The acquisition context represents the information about image acquisition context. The ROSESIM acquisition context is depicted in figure 6.12

6.2.4.4 Context Criteria

The ROSESIM interpretation knowledge base contains ten **context criteria**.

Initialization interpretation criteria enable to initialize the semantic interpretation knowledge base according to the current contextual information. For example, the following criteria infers the importance of the fungal infection sub-classes according to the climatic contextual information.

- Criteria **name** Climatic_fungi
  type : Initialization Interpretation Criteria
  Domain Class : FUNGAL INFECTION

DomainClass {
name HYPHAE
comment *"A thread-like, tubular filamentous fungal structure"*
SuperClass FUNGISYMPTOM
Visual Description
Spatial Description
Geometry **name** hyphae_geometry
range [*Curve Line Segment*]
Thickness **name** hyphae_thickness
range [*Very_Thin Thin*]
Straightness **name** hyphae_straightness
range [*Almost_Straight*]
Color Description
Neutral_Color **name** hyphae_color
range [*White Gray*]
Lightness **name** hyphae_lightness
range [*Very_Light Light*]
Spatial Relation Description
ProperPartOf **name** hyphae_properpart_relation
range [LEAF]
}

Figure 6.9: Representation of the domain class HYPHAE

Let c a ROSESIM_Domain_Context
If : c.Relative_humidity >= 80 and c.Greenhouse_temperature $\in [25, 30]$
Then :CONIDIA_GERMINATION.setImportance(0.9), MYCELIUM_INSTALLATION.setImportance(1), SPREADING_POWDERY_MILDEW.setImportance(1),
VERY_EARLY_POWDERY_MILDEW.setImportance(1)

Post interpretation criteria enable to confirm or to refine the interpretation results according to the contextual information. They enable to take into account the risk factors of the disease to make an accurate diagnosis. For example, the presence of early powdery mildew with climatic conditions contrary to the powdery mildew development ones is of less importance than the presence of early powdery mildew in favorable conditions for its growing.

To pursue with the example of Powdery Mildew, a post interpretation criteria is:

- Criteria **name** Powdery_mildew_criteria1
  type : Post Interpretation Criteria
  Domain Class : VERY_EARLY_POWDERY_MILDEW
  Let c a ROSESIM_Domain_Context
  If : true and c.Relative_humidity >= 80 and c.Greenhouse_temperature $\in [25, 30]$ and c.Season $\in [Spring, Automn]$
  Then :print ALERT : *very Early Powdery Mildew in favorable conditions. A treatment is needed*

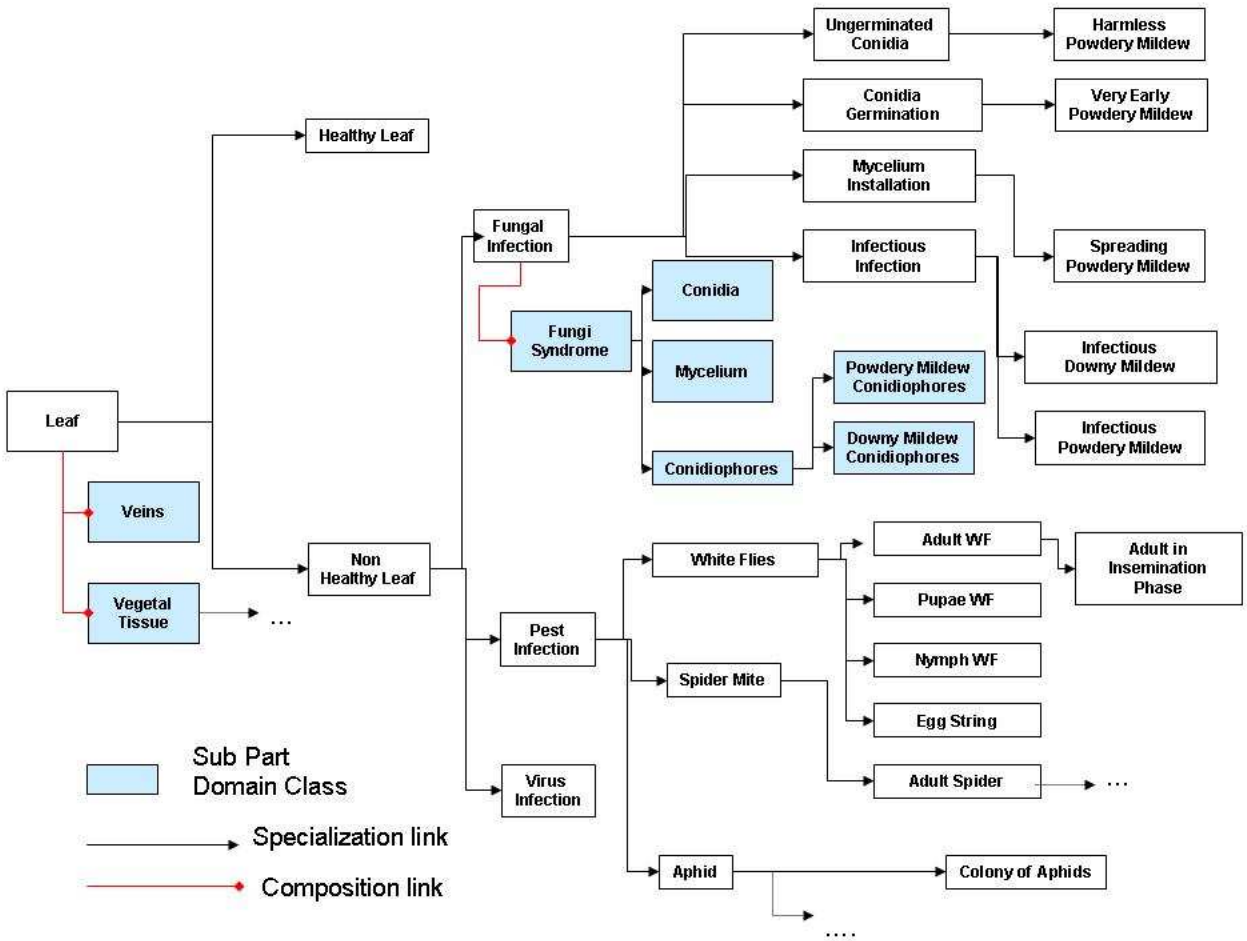


Figure 6.10: Overview of a part of the hierarchy of domain classes

6.2.5 The ROSESIM Semantic Interpretation System

The ROSESIM interpretation knowledge base is gathered with the application independent SI engine presented in section 5.1.5. The resulting association is a semantic interpretation system dedicated to rose diseases.

6.2.6 The Visual Data Management Knowledge Base

The building of the ROSESIM visual data management knowledge base is on one hand guided by the visual concept ontology and spatial relation ontology and on the other hand guided by the image processing ontology. The knowledge acquisition consists in:

1. Building the description, from a visual data management point of view, of all the visual concepts and spatial relations used to describe the set of domain classes in the ROSESIM semantic interpretation system. The building of the descriptions consists in the building of the grounding link with image descriptors. The choice of image descriptors is constrained by the image processing ontology. For each visual concept used in the semantic interpretation knowledge base (or for each visual concept provided by the visual concept ontology), we first build its general description (**Visual**

```
DomainContext {
    name ROSESIM_Domain_Context
    comment The Domain context of the rose disease diagnosis application
    Attributes
        Float name Relative_humidity
            default no
            range [0 100]
        Float name Greenhouse_temperature
            default no
        Symbol  name Season
            range [Winter Spring Summer Autumn]
        Symbol name Plant_Location
            range [Entrance Middle back ]
        Symbol name Organ_Location
            range [Bottom Middle Top]
        Symbol name Rose_Variety
            range [Leonidas Texas FirstRed]
        Symbol name Leaf_Age
            range [Very_Young Young Mature Old]
    }
```

Figure 6.11: The domain context for the application of rose disease recognition

```
AcquisitionContext {
    name ROSESIM_Acquisition_Context
    comment The Acquisition context of the rose disease diagnosis application
    Attributes
        Symbol name Sensor_type
            default Binocular_microscope
            range [Binocular_microscope Video_microscope microscope]
        Float name Sensor_magnification
            default 64
            range [7.5 200]
        Float name Sensor_resolution
            range [0.25 0.35 0.5 1.0 4.0] 10 E-4
    }
```

Figure 6.12: The acquisition context for the application of rose disease recognition

Concept Taxonomy) then we build the instances corresponding to this visual concept and present as attributes of domain classes (**visual concept instances**). The knowledge of spatial relations is built according to the spatial relation ontology.

2. Then, a set of visual data management criteria is built and linked to the corresponding visual concept or spatial relation.

3. The set of generic image processing functionalities is described and the set of image entities interesting for our application is described.

A big part of the visual data management knowledge base is not specific to the rose disease application. This part composes the generic visual data management knowledge base. The visual data management also contains knowledge dependent on the ROSESIM application : i.e. the instances of visual concepts and of spatial relations related to the domain classes of the semantic interpretation knowledge base. In the next sections, we summarize the content of the visual data management knowledge base.

6.2.6.1 Visual Concept Taxonomy

The visual concept taxonomy contains an organized set of visual concepts. It is the description from the visual data management point of view of the visual concepts of the **visual concept ontology**: i.e. with their grounding link as described in figure 6.13, 6.14 and 6.15. In our case, the visual concept taxonomy corresponds to a restriction of the set of concepts provided by the visual concept ontology. Only the visual concepts and their sub-concepts which are useful to describe the application domain are included in the visual data management knowledge base. In the case of the ROSESIM application, texture visual concepts were not used and as a consequence not including in the visual data management knowledge base. The current ROSESIM visual data management knowledge base contains the generic description of 107 visual concepts. Some of them are described in appendix B.

VisualConcept {
  name *Thickness*
  SuperConcept *Size*
  Grounding Link
    Float name width
      Numerical range: [0..image_height*resolution]
      unit: [km, m, cm, mm ,um, nm]
  }

Figure 6.13: The representation of the visual concept *Thickness* in the visual data management knowledge base

The taxonomy of visual concepts in the visual data management knowledge base is the same as the visual concept ontology. The process of inheritance between the visual concept *Elliptical Surface* and *Circular Surface* is shown in figure 6.14 and 6.15.

6.2.6.2 Image Data and Descriptor Set

The visual data management knowledge base also contains knowledge about image data types. This knowledge is the description of image entities and image descriptors provided

VisualConcept {
  name *Elliptical_Surface*
  SuperConcept *Geometric Surface*
  Grounding Link
    Symbol name eccentricity
      comment *ratio of the length of the longest chord to the longest chord perpendicular to it*
      Linguistic range: [very_low low medium high very_high]
      FuzzySet
      $F_{verylow} = \{0.0, 0.0, 0.19, 0.21\}$
      $F_{low} = \{0.19, 0.21, 0.38, 0.42\}$
      $F_{medium} = \{0.38, 0.42, 0.575, 0.625\}$
      $F_{high} = \{0.575, 0.625, 0.76, 0.84\}$
      $F_{veryhigh} = \{0.76, 0.84, 1, 1\}$
      Domain: [0 1]
      unit: none
    Symbol name compactness
      comment *Measure of how the shape is closely-packed*
      ...
    Symbol name ellipticity
      comment *Euclidian ellipticity : distance between fitting ellipse and region boundary*
      ...
}

Figure 6.14: The representation of the visual concept *Elliptical_Surface* in the visual data management knowledge base

by the image processing ontology. In the current ROSESIM visual data management knowledge base, we have only described image data and descriptor sets relevant for our application and with regard to the image processing ontology. A typical example of an image data is described in figure 6.16.

6.2.6.3 Spatial Relations

The visual data management knowledge base contains knowledge of spatial relations. This knowledge corresponds to the description of the spatial relations provided by the spatial relation ontology according to the formalism described in chapter 5. The visual data management knowledge base contains 22 spatial relations. Example of the representation of the topological spatial relation *Proper_Part_Of* is given in 6.17.

6.2.6.4 Visual Data Management Criteria

- **Object Extraction Criteria**
  Object extraction criteria are linked to **visual concepts** or to **spatial relations**. They are used to guide information extraction on images. Currently, the minimal visual data management knowledge base contains 18 extraction criteria. They are

VisualConcept {
name *Circular_Surface*
SuperConcept *Elliptical_Surface*
Grounding Link
Symbol name eccentricity
comment *ratio of the length of the longest chord to the longest chord perpendicular to it*
Linguistic range: [high very_high]
FuzzySet
$F_{high} = \{0.575, 0.625, 0.76, 0.84\}$
$F_{veryhigh} = \{0.76, 0.84, 1, 1\}$
Domain: [0 1]
unit: none
Symbol name compactness
comment *Measure of how the shape is closely-packed*
Linguistic range: [high very_high]
...
Symbol name circularity
comment *Shape factor*
...
}

Figure 6.15: The representation of the visual concept *Circular_Surface* in the visual data management knowledge base

used to fill the values of the current **visual content context**.

A simple example of object extraction criteria is the following:

– Criteria **name** SurfacicShape_extraction_1
type : Object Extraction Criteria
Visual Concept : *Geometric_Surface*
Let **VO** a Visual Object and **context** a visual Content Context
If : **VO**.geometry has for expected values *Surface* or *Surface*.Subconcepts
Then : **context.Image_Entity_type**:= Region

In this case, the object extraction criteria is linked to the visual concept *Surfacic_Shape* but some object extraction criteria are also linked to spatial relations.

For example, the following criteria states about the discriminacy of color between two objects lies with the *ProperPart* relation. They are based on the complementary of hue concepts.

– Criteria **name** ProperPartColor_extraction_red
type : Object Extraction Criteria
Spatial Relation : Proper Part Of
Let VO1 and VO2 be the two visual objects in relation (VO1 is **Proper Part** of VO2) and **context** a visual Content Context

ImageData {
  name Image_Region
  comment *Set of connected pixels which have common properties*
  Subtype Of Image_Data
Attributes
  Image_Region name region_interior
  Image_Edge name region_boundary
  RegionShapeDescriptorSet name region_shape_description
  RegionColorDescriptorSet name region_color_description
  File name dataFile
Methods
  Load()
  Save()
  }

Figure 6.16: The representation of the Image Data *Image_Region* in the visual data management knowledge base

SpatialRelation {
  name *Proper_Part_Of*
  SuperRelation *TopologicalRelation*
  Inverse *Has_For_Proper_Part*
  Complement *Discrete*
  Symmetry *False*
  Objects_In_Relation
    VisualObject **name** object1
    VisualObject **name** object2
  Conditions
    Difference(Interior(object1), Interior(object2)):= empty
    Difference(Interior(object2), Interior(object1)) != not empty
    Intersection(Interior(object1), Interior(object2)) != not empty
  }

Figure 6.17: The representation of the spatial relation *Proper_Part_Of* in the visual data management knowledge base

If : **VO1**.GenericHue has expected value *Red*
and **VO2**.GenericHue has expected value *Green*
Then : **context.Discriminative_Object_Color**:= High

The following criteria uses the relation NTTP to define an area of interest on the image:

- Criteria **name** NNTP_constraint
  type : Object Extraction Criteria
  Spatial Relation : *TPP*
  Let VO1, VO2 be two visual objects and context a Visual Content Context
  If : TTP(VO1, VO2) is true
  Then : context.areaofInterest:= Interior(VO2)

- **Spatial Deduction Criteria**
  They are linked to spatial relations. The minimal visual data management knowledge based contains:

  - The 7 transitivity criteria corresponding to the spatial relations: *Left Of*, *Right Of*, *In Front Of*, *Behind*, *Equals* , *NTTP* and *NTTP-1*.
    The transitivity criteria for the *Right Of* orientation relation is:
    Criteria **name** RightOf_transitivity
    type : Spatial Relation Criteria
    Spatial Relation : *Right_Of*
    Let VO1, VO2, VO3 be three visual objects
    If : *Right_Of*(VO1, VO2) and *Right_Of*(VO2, VO3)
    Then : *Right_Of*(VO1, VO3) is true
  - The composition criteria corresponding to the composition table.
    All these composition criteria have not been implemented for the ROSESIM application. We have implemented only the criteria related to spatial relations effectively used to describe domain classes. For example, the composition of the spatial relation *NTPP* and *Left Of* is represented by the following criteria :
    Criteria **name** NNTP_LeftOf
    type : Spatial Relation Criteria
    Spatial Relation : *NTPP*
    Let VO1, VO2, VO3 be three visual objects
    If : *NTPP*(VO1, VO2) and *Left_Of*(VO2, VO3)
    Then : *Left_Of*(VO1, VO3) is true

- **Evaluation Criteria**
  Evaluation Criteria are represented independently in the visual data management knowledge base to evaluate the result of the program supervision module according to what was expected at the visual data management level. Currently, this evaluation is done interactively with the end user using an evaluation criteria which asks to the end user if the results of the image processing are : (1)correct, (2)incorrect, (3)under-segmented and (4)over-segmented.

6.2.6.5 Visual Concept Instances

This part of the visual data management knowledge base is dependent on the application domain. It corresponds to a set of instances of visual concepts related to domain classes of the semantic interpretation knowledge base. These instances define what we have called the **grounding relation**. Indeed, the grounding relation between visual concepts and image data descriptors is dependent of the domain class described by the visual concept. For example, in a road detection application, the visual concept *Thickness* used to describe a road has not the same grounding relation that the same visual concept used to describe the domain class HYPHAE. For each domain class and its associated visual concepts, the grounding relation is represented by instances of visual concepts. Theses instances are stored in the visual data management knowledge base as described in the figure 6.18.

VisualConcept Instance {
Thickness **name** hyphae_thickness
Grounding Link
Float name width
Numerical range: [1-10]
unit: um
}

VisualConcept Instance {
Elliptical_Surface **name** WFpupae_elliptical_surface
Grounding Link
Symbol name ellipticity
Linguistic range: [high very_high]
}

Figure 6.18: Instances of visual concepts in the visual data management knowledge base

6.2.7 The ROSESIM Program Supervision Knowledge Base

For the image processing part of our application, we have used a library of image processing programs called PANDORE [1]. PANDORE is a library of image processing operators. It is composed of a set of executable operators and of an environment of programming in C++.

The program supervision knowledge acquisition was guided by the image processing ontology. The generic functionalities of the image processing ontology are functionalities represented in the knowledge base. For each functionality, we have defined primitive and composite operators able to achieve these functionalities. Moreover, for interoperability, the arguments of supervision operators correspond to image data in the visual data management knowledge. The figure 6.19 represents arguments corresponding to the Image Data *Image_Region* represented before.

The minimal program supervision knowledge base contains:

1. The description of image data that can be processed by image processing programs. They are the data concepts of the image processing ontology (e.g. chapter 4). They are described within **data types** in a clear and structured way. This part of the knowledge base is important because the management of image processing programs is essentially data_driven.

[1] http://www.greyc.ensicaen.fr/~regis/Pandore/

Argument Type {
name *Image_Region*
comment *Set of connected pixels which have common properties*
Subtype Of Image_Data
Attributes
Image_Region name region_interior
Image_Edge name region_boundary
RegionShapeDescriptorSet name region_shape_description
RegionColorDescriptorSet name region_color_description
RegionTextureDescriptorSet name region_shape_description
Methods
Load(File file)
Save(File file)
}

Figure 6.19: The representation of an argument type of the program supervision knowledge base. The corresponding ImageData is described in the visual data management knowledge base.

2. The description of the generic image processing functionalities of the image processing ontology. They correspond to the set of functionalities used to build program supervision requests by the program supervision module and to the associated program supervision operator representation. These generic image processing functionalities are :

   - Object Extraction
     Its representation in the YAKL language is described in figure 6.20. The hierarchy of supervision operators associated to this functionality is represented in figure 6.21. The representation of associated supervision operators is given in the annex B.
   - Feature Extraction
     For each image descriptor of the image processing ontology a primitive or composite supervision operator which describes at least a method to measure the descriptor on images is included in the program supervision knowledge base.
   - Visual grouping
   - Visual splitting
   - Image Enhancement

For complex applications requiring specific image processing programs, an application specific knowledge base can be built incrementally from the generic minimal knowledge base.

Functionality {
  name *Object_Extraction*
  comment *Extraction of a visual object from image*
  Achieved by ObjectExtraction_operator
InputData
  Image name inputImage
  VisualContentContext name context
OutputData
  List of Image Data name outputData
  IPReport name reports
  }

Figure 6.20: Representation of the functionality Object extraction

The program supervision knowledge base was built by using the visual image processing ontology as a skeleton. This part of the program supervision knowledge base is a *minimal program supervision knowledge base*. Additional knowledge (additional supervision operators) were added to cope with our biological application. In the current implementation, the program supervision ROSESIM knowledge base contains 29 supervision operators which solve the 5 basic image processing functionalities and 35 criteria, mainly initialization and choice criteria.

6.2.8 Example of session

In this section, we illustrate the behavior of ROSESIM with a typical end-user request. The request called *Powdery_mildew_request* is a **single detection** request but has the particularity of involving composite objects and spatial relations.

6.2.8.1 A *Single Detection Request*

This section describes the behavior of the ROSESIM system in response to a *Single Detection Request*.

The global process begins with the specification of a **domain request** by the end user:

Domain Request {
  name Powdery_Mildew_Request
  comment *Is there an infection of powdery mildew on the rose leaf and at what stage?*
  InputImageData : = Leonidas0602
  DomainContext : = Leonidas0602DContext
  AcquisitionContext : = Leonidas0602AContext
  HighLevelGoal:= Single_Detection
  TargetedDomainClass:= FungalInfection
}

The description of the end user request, i.e. the image to interpret and its associated context is described in figure 6.22.

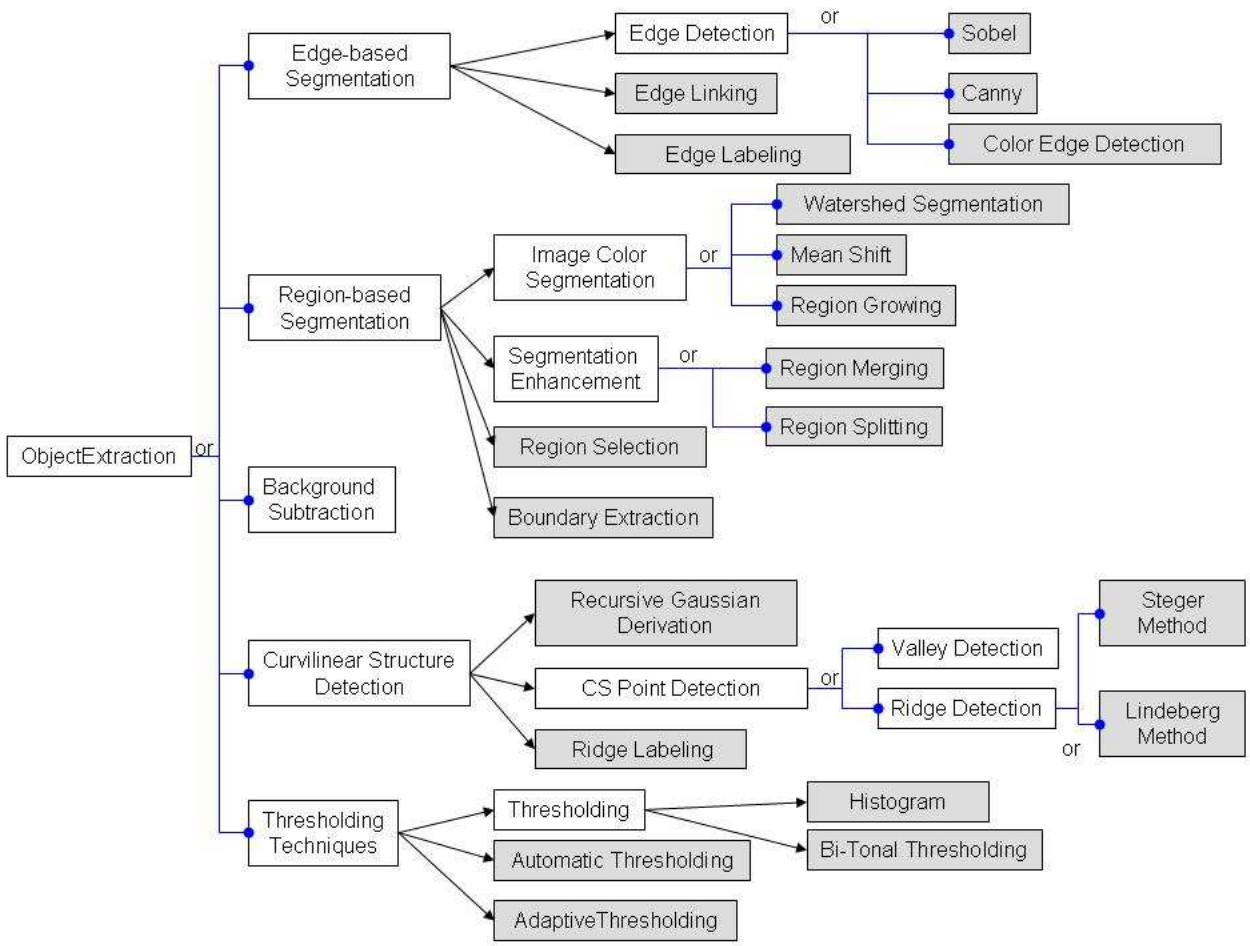


Figure 6.21: Hierarchy of supervision operators associated to the **functionality** *Object Extraction* in the ROSESIM Program Supervision Knowledge Base. Operators with a grey background are **Primitive Operators**. Operators with a white background are **Composite Operators**.

This domain request means that the end user is only interested in fungal infection. In particular, he wants to know if powdery mildew is present and in what stage of development.

This domain request represents a real biological problem. Indeed, powdery mildew causes severe damages on rose leaves and the detection of powdery mildew in early stages is a real major challenge for pathologists.

Moreover, from the point of view of the semantic image interpretation system, this request enables us to illustrate the management of complex composite objects.

At the beginning of the interpretation: DomainClassList is empty and Perceived Scene Description is empty.

1. *Semantic Interpretation Initialization Phase*
   The semantic interpretation engine takes as input the domain request and perform **Initialization**. The **Initialization** consists in loading the files corresponding to the

Leonidas0602

Domain Context

DomainContextInstance{
ROSESIM_Domain_Context **name** Leonidas0602DContext
Attributes
Relative_humidity := 70 %
Greenhouse_temperature :=21 ºC
Season :=Spring
Plant_location :=Entance
Organ_location :=Top
Rose_Variety:=Leonidas
Leaf_age:=Very_young
}

Acquisition Context

AcquisitionContextInstance{
ROSESIM_Acquisition_Context **name** Leonidas0602AContext
Attributes
Sensor_type := Binocular_microscope
Sensor_magnification := 64
Sensor_resolution:=0.00000025
}

Figure 6.22: Description of the input data of the end user Powdery_Mildew_Request

Domain and Acquisition Context. Only the part of the **domain taxonomy** from the **targeted domain class** of the domain request is taken into account. In our case, only the **domain classes** which specialize or compose FUNGAL INFECTION are taken into consideration. The **domain class** FUNGAL INFECTION is added in DomainClassList. A visual object (VO_background) corresponding to the root of the domain taxonomy: i.e. LEAF, is built and added in the perceived scene description. Then, some **Initialization Interpretation Criteria** are activated.

- Criteria **ContextAge_1**
  type: Initialization Interpretation Criteria
  Let : C a Domain context and V a Visual Object
  If : C.Leaf_Age == Very Young and C.Rose_Variety == Leonidas
  Then : V.Hue := Dark Red

Current Domain Class := FUNGAL_INFECTION (First class in the DomainClassList).

The semantic interpretation engine loads the description of the domain class FUNGAL_INFECTION in the knowledge base. This description is :

DomainClass {
  name FUNGAL_INFECTION

comment *An infection by a fungal pathogen characterized by the presence of a funny symptom*
SuperClass name NON_HEALTHY_LEAF
SubPart Description
FUNGI_SYMPTOM **name** fungal_infection_symptom
range [MYCELIUM CONIDIA CONIDIOPHORES]
facet at_least 1 }

2. *Analysis of the subpart of the current domain class*
According to the description of the domain class FUNGAL_INFECTION, the semantic interpretation deals with the sub-part domain classes. The **facet** at_least means that the presence of only one FUNGI_SYMPTOM is sufficient to validate the domain class FUNGAL_INFECTION. The first sub-part domain class studying is the first domain class of the range of the attribute fungal_infection_symptom, i.e. the domain class MYCELIUM.

CurrentDomainClass := MYCELIUM
The Description of MYCELIUM is:

DomainClass {
name MYCELIUM
comment *A group or mass of discrete hyphae, the vegetative structure of many fungi*
SuperClass name FUNGI_SYMPTOM
Visual Description
Spatial Description
NetworkOf **name** mycelium_network
range [(HYPHAE,*Connected*)]
atleast 1
Network_Density **name** mycelium_density
range [*Partially_Spaced Spaced*]
NetworkShape **name** mycelium_shape
range [*Star_Like*]
ProperPart **name** mycelium_proper_part_of
range [*Leaf*] }

3. *Visual data management request building*

The semantic image interpretation builds a **visual data management request** for the visual data management module according to the semantic knowledge of the domain class MYCELIUM and the high level goal of the request.

The resulting request is described just below using the semantic knowledge about the domain class MYCELIUM and the high level goal of the domain request.

VisualDataManagementRequest{
name mycelium
Attributes
CurrentVisualObject := Visual_Object_mycelium1

ObjectNumber := 1
}

According to the description of the domain class MYCELIUM, the visual object Visual_Object_1 is a composite visual object.

Its description is:

Composite Visual Object:
name := Visual_Object_mycelium1
state := hypothesized
AssociatedDomainClasses := Unknown
AssociatedImageData := Unknown
VisualAttributes
Network_Shape **name** network_shape
expectedValues := [*Star_Like_Network*]
PerceivedValues := Unknown (0)
Spatial Relation := Connected
PrimitiveVisualObject := Visual_Object_hyphae1
ObjectNumbermin:= 1
ObjectNumbermax:= Unknown

According to the description of the domain class HYPHAE which composes the domain class MYCELIUM, Visual_Object_hyphae1 is a visual object described by:

Primitive Visual Object:
name Visual_Object_hyphae1
state := hypothesized
AssociatedDomainClasses := Unknown
AssociatedImageData := Unknown
VisualAttributes
Shape **name** shape
expectedValues := [*Line Segment*]
PerceivedValues := Unknown (0)
Thickness **name** thickness
expectedValues := [*Thin Very_Thin*]
PerceivedValues := Unknown (0)
Straightness **name** straightness
expectedValues := [*Almost Straight*]
PerceivedValues := Unknown(0)
NeutralColor **name** neutralcolor
expectedValues := [*White Gray*]
PerceivedValues := Unknown(0)
Lightness **name** lightness
expectedValues := [*Very_Light Light*]
PerceivedValues := Unknown(0)

The spatial relational attribute *IsProperPart* named hyphae_properpart_relation and which has for value LEAF is processed. LEAF is a special domain class which represents all the visual scene and, as a consequence, all the image. It is always true. A corresponding visual object named VO_background has been added in the perceived scene description at the beginning of the process. An instance of the spatial relation *Proper Part of* between Visual_Object_hyphae1 and VO_background is created and added to the Perceived Scene Description. Due to the special domain class LEAF, this relation is true.

4. *Visual data management request processing*
   As the visual data management request is a composite visual object request, the visual data management engine first processes the primitive visual object Visual_Object_hyphae1.

   Current Visual Object:= Visual_Object_hyphae1

   As Visual_Object_hyphae1.AssociatedImageData == Unknown, the visual data management engine asks for object extraction in the images. It builds an Image Processing Request.

5. *Image Processing Request Building (Object Extraction)*
   It consists in the activation of **object extraction criteria**. The aim is to complete the fields of the visual content context corresponding to the image processing functionality. As object extraction criteria are linked to visual concepts or spatial relations, only the criteria linked to visual concepts which describe the visual object are activated. An object extraction criteria linked to a visual concept is automatically linked to its sub-concepts.

   - Criteria **Open_Curve_extraction_1**
     type : Object Extraction Criteria
     Visual Concept : Open Curved
     Let **VO** a Visual Object and **context** a visual Content Context
     If: **VO**.thickness.expectedValues $\ni$ *Thin* or **VO**.thickness.expectedValues $\ni$ *Very Thin*
     Then: **context.Image_Entity_type**:= Curvilinear Structure
   - Criteria **Thickness_1**
     type : Object Extraction Criteria
     Visual Concept : Thickness
     Let **VO** a Visual Object, **context** a visual Content Context, **acquisitioncontext** an Acquisition Context
     If: **VO**.thickness.expectedValues $\ni$ *Thin* or **VO**.thickness.expectedValues $\ni$ *Very Thin*
     Then: **context.relative_object_width**:= Thickness.getInstance().GroundingLink.width * acquisitioncontext.sensorResolution
   - Criteria **Thickness_2**
     type : Object Extraction Criteria
     Visual Concept : Thickness
     Let **VO** a Visual Object, **context** a visual Content Context
     If: **VO**.thickness.expectedValues $\ni$ *Thin* or **VO**.thickness.expectedValues $\ni$ *Very Thin*
     Then: **context.relative_object_length** >= 10* context.relative_object_width

- Criteria **ProperPartColor_extraction_2**
  type : Object Extraction Criteria
  Spatial Relation : Proper Part Of
  Let VO1 and VO2 be the two visual objects in Proper Part relation (VO1 is **Proper Part** of VO2) and **context** a visual Content Context
  If: **VO1**.lightness.expectedValues $\ni$ *Light* or **VO1**.lightness.expectedValues $\ni$ *Very Light*
  and **VO2**.lightness.expectedValues $\ni$ *Dark* or **VO2**.lightness.expectedValues $\ni$ *Very Dark*
  Then: **context.Discriminative_Object_Intensity**:= High

The visual data management system sends to the Program supervision system an image processing request corresponding to :

Request {
Object_Extraction **name** object_extraction1
Attributes
input_image := Leonidas0602
visual_content_context := context }

The Visual Content Context context is :

VisualContentContextInstance {
VisualContentContext **name** object_extraction_context1
Attributes
image_data_type := *Curvilinear_Structure*
discriminative_object_intensity := high
discriminative_object_color := true
discriminative_object_texture := no
relative_object_width := [5 10]
relative_object_length >= 50
relative_object_size := unknown
object_number >= 1 }

6. *Image processing request solving by the Program supervision module*

The processing of the program supervision request *object_extraction1* is shown in figure 6.23.

All the resulting image data are set in the ImageDataList

7. *Interactive Evaluation of Image Processing Results*
The aim of this evaluation phase is to make decisions on the necessity of a feedback to the program supervision system or if the image processing results are sufficient to pursue the interpretation process. It consists in the activation of the evaluation criteria of the visual data management knowledge base. In the present case, the evaluation is made interactively with the end user.

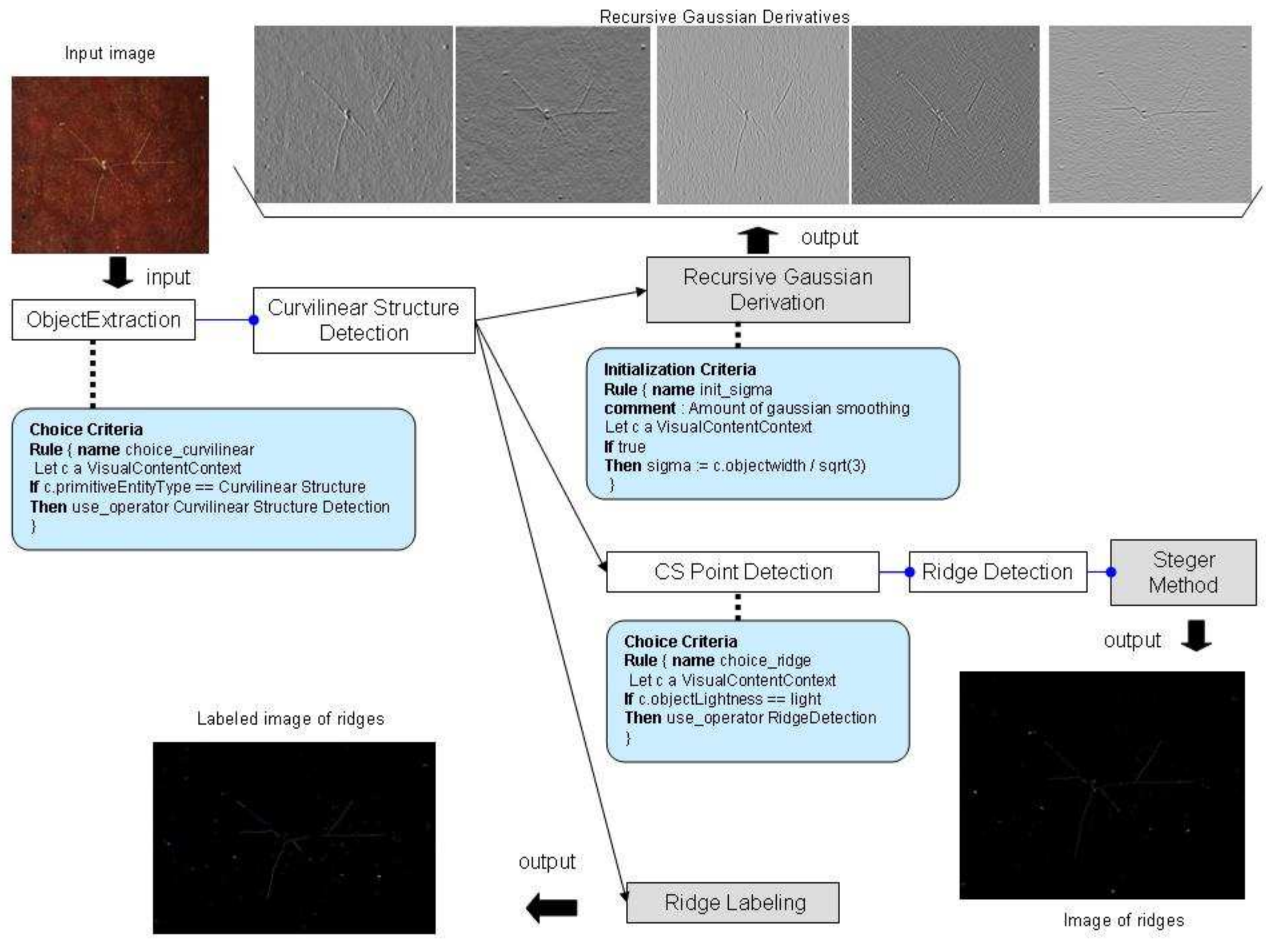


Figure 6.23: The processing of the request *object_extraction1* by the program supervision module. The criteria used during the process of the request are in oblong bounding boxes. The images in the figure are input or output of image processing programs. The complete output is a list of labeled ridge lines called ImageData_List1 with basic descriptors like the position, basic size descriptors, its interior and its boundary

- Criteria **evalseg_1**
  type : Object Extraction Evaluation Criteria
  If : true Then : assess_data_by_user [correct undersegmentation oversegmentation noisy]

In our case, the results are assessed correctly.

8. *Program Supervision Request Building (Feature Extraction)*
   The visual data management system asks for feature extraction.

   Request {
   Feature_Extraction **name** feature_extraction1
   Attributes
   input_data := ImageData_List
   visual_content_context := context2
   }

The visual content context is created by using the grounding link of the visual concept instances contained in the current visual object.

VisualContentContextInstance {
VisualContentContext **name** feature_extraction_context1
Attributes
image_data_type := *Curvilinear_Structure*
shapeFeatures := {elongation, compactness, circularity, eccentricity, orientation, line_strenghness}
sizeFeatures := width, length, area
colorFeatures := greylevelvalue, R, G, B
textureFeatures := none
}

9. *Image processing request solving by the Program supervision module*
The results are image data and their associated image data descriptors.

10. *Image Data Selection* The aim of this step is to remove from the ImageDataList all the image data that are useless for the following step of the process. The constraints of the Visual Content Context object_extraction_context1 are used to performed this selection. In our case, all the image data which have width and length values out of the range of the **relative_object_width** and **relative_object_length** are removed.

11. *Symbolic description generation* Each image data in ImageDataList is matched with the description of the primitive visual object *visual object hyphae1* previously defined. If the matching is correct, the *visual object hyphae1* is duplicated and completed according to the matching process. An example is:

Primitive Visual Object:
name Visual_Object_hyphae1
state := perceived
AssociatedDomainClasses := Unknown
AssociatedImageData := ImageDataList(i)
VisualAttributes
Shape **name** shape
expectedValues := [*Line Segment*]
PerceivedValues := Line(0.7)
Thickness **name** thickness
expectedValues := [*Thin Very_Thin*]
PerceivedValues := Thin (0.8) Very_Thin (0.2)
Straightness **name** straightness
expectedValues := [*Almost Straight*]
PerceivedValues := AlmostStraight (0.7)
NeutralColor **name** neutralcolor
expectedValues := [*White Gray*]
PerceivedValues := White (0.7) Gray (0.5)
Lightness **name** lightness
expectedValues := [*Very_Light Light*]

PerceivedValues := Light(0.8) VeryLight(0.4)

12. *Program Supervision Request Building (Top Down Visual Grouping)*
    The following request is sent to the program supervision system:

    Request {
    Top_Down_Visual_Grouping **name** visualgrouping1
    Attributes
    input_data := ImageData_List
    relation := Connected
    targetedstructure := network
    }

    The image processing result of the grouping process is shown in figure 6.24.

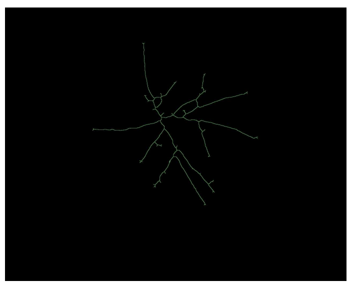
    

    Figure 6.24: The image resulting from the grouping process

    The result is set to ImageData_Graph

13. *Program Supervision Request Building (Feature Extraction)*
    The visual data management system asks for feature extraction.

    Request {
    Feature_Extraction **name** feature_extraction1
    Attributes
    input_data := ImageData_Graph
    visual_content_context := context3
    }

    The visual content context is created by using the grounding link of the visual concepts contained in the visual object.

    VisualContentContextInstance {
    VisualContentContext **name** feature_extraction_context2
    Attributes
    image_data_type := *graph*
    shapeFeatures := { graph_convexarea, graph_compactness, graph_circularity, graph_density }

sizeFeatures := total_length, total_area, graph_density
colorFeatures := none
textureFeatures := none
}

14. *Symbolic description generation*
The imageData_Graph and its descriptors are matched with the description of the Composite Visual Object :

Composite Visual Object:
name Visual_Object_mycelium1
state := perceived
AssociatedDomainClasses := Unknown
AssociatedImageData := imageData_Graph
VisualAttributes
Network_Shape **name** network_shape
expectedValues := [*Star_Like*]
PerceivedValues := Star_Like (0.7)
Network_Density **name** network_density
expectedValues := [*Very_Dense Dense Partially_Spaced Spaced*]
PerceivedValues := Partially Spaced (0.5) Spaced (0.8)
Spatial Relation := Connected
PrimitiveVisualObject := Visual_Object_hyphae1
ObjectNumbermin:= 1
ObjectNumber:= 20

15. *Semantic Matching (Visual_Object_mycelium1, CurrentDomainClass)*
The domain class MYCELIUM is recognized. The state of the Visual_Object_mycelium1 is changed to *recognized* and the name of the domain class MYCELIUM is set as a value in the slot AssociatedDomainClasses.

Current Domain Class := FUNGAL_INFECTION (its subpart MYCELIUM has been recognized).

As the domain class FUNGAL_INFECTION has not other attributes than its sub part attribute FUNGISYMPTOM, it is validated. The name of the domain class MYCELIUM is set as a possible semantic value of the current visual object (using the slot AssociatedDomainClasses).

16. *Refinement of the semantic interpretation*

The sub-classes of the domain class FUNGAL INFECTION are sorted out according to their priority order and add in the list of domain classes.

Current Domain Class := MYCELIUM_INSTALLATION

DomainClass {
name MYCELIUM_INSTALLATION
comment *Presence of mycelial clump*
SuperClass name FUNGAL_INFECTION

visual Description
Network_Density **name** mycelium_density
range [*Partially_Spaced Spaced*]
}

The domain class MYCELIUM_INSTALLATION is validated. The name of the domain class MYCELIUM_INSTALLATION is set as a value in the slot AssociatedDomainClasses.

Current Domain Class := SPREADING_POWDERY_MILDEW

DomainClass {
name SPREADING_POWDERY_MILDEW
comment *Presence of mycelial clump of powdery mildew*
SuperClass name SPREADING_POWDERY_MILDEW
Visual Description
Mycelium.Hyphae.Thickness **name** hyphae_thickness
range [*Thin*]
}

The domain class SPREADING_POWDERY_MILDEW is validated. The name of the domain class MYCELIUM_INSTALLATION is set as a value in the slot AssociatedDomainClasses.

It is a leaf of the tree of domain classes, the interpretation process is stopped.

17. *Post Interpretation phase* Activation of Post Interpretation criteria. ALERT : *very Early Powdery Mildew in favorable conditions. A treatment is needed*

6.3 Conclusion

In this section, we have presented the validation of the cognitive vision platform for a real world application. The cognitive vision platform was used to build the ROSESIM system dedicated to the recognition of several rose diseases. Using the cognitive vision platform, the building of the ROSESIM system only consists in the building of three dedicated knowledge bases. They contain 218 frames and 73 rules. They are provided in the appendix B. The semantic interpretation knowledge base and the visual data management knowledge base are the most developed. They both illustrate the re-usability of the visual concept ontology and of the spatial relation ontology. The session example with the *Powdery_mildew_request* has illustrated the different concepts and reasoning strategies presented in the chapter 5. The ROSESIM system was tested on about 50 images. The images represent either fungi diseases or insects. In most of the case, the ROSESIM system leads to a correct interpretation.

The ROSESIM system validates the using of the cognitive vision platform for the building of application specific interpretation system:

- The descriptive knowledge for the three sub-problems of semantic image interpretation can be represented in a natural way.
- The different knowledge bases can be easily modified and augmented by the different experts.

- The different engines have been tested.

The aim of the ROSESIM system was the validation of the cognitive vision platform. As a consequence, the biological validation of the ROSESIM system has still to be done. Nevertheless, according to the interpretation results of the tested images, we can conclude that the system is successful in recognizing the different types of diseases described in the knowledge base. Bad interpretations are often due to very noisy images for fungal infection. This weakness is due to a hard image processing problem: the extraction of thin filamentous objects in clustered background. The domain knowledge base can be increased and modified by application domain experts to cope with a wider range of rose diseases. A possible extension of the ROSESIM system could be the management of not only pest infection but also the management of the presence of their natural enemies and their relations.

Chapter 7

Conclusions and Perspectives

Despite an active and mature research in computer vision and artificial intelligence, the problem of semantic image interpretation is still a significant challenging problem. In this thesis, we address this problem under the point of view of the building of semantic image interpretation systems. This problem is important for a wide range of applications: visual surveillance, diagnosis, medical imaging, remote sensing, industrial inspection,... Current drawbacks of existing semantic image interpretation systems are their application domain dependence, their lack of robustness, flexibility and autonomy and their long design-cycle time.

Recently, the emerging discipline of cognitive vision tries to answer to these drawbacks. It gathers several scientific fields (computer vision, pattern recognition, artificial intelligence, cognitive science, machine learning, knowledge engineering) with the common attempt to achieve more *robust, resilient and adaptable computer vision systems by endowing them with cognitive faculties* [Vernon, 2004]. In particular, one of the scientific challenges of cognitive vision is the establishment of minimal architecture for cognitive vision systems. From the methodological point of view, some challenges are concerned with the utilization and the advancement of systems engineering methodologies.

Our cognitive vision platform is a contribution for the research in cognitive vision. Indeed, as a reusable and convenient environment of development for the design of semantic image interpretation systems, the cognitive vision platform considerably reduces the design-cycle time of systems. Moreover, it is a minimal functional architecture which defines the minimal set of processing modules and their inter-relationships necessary for semantic image interpretation systems. Our approach, based on knowledge based systems enables a natural interaction with the different experts involved in the building of a semantic image interpretation system.

The first part of this chapter is a review of our cognitive vision platform. In particular, we focus on how the proposed cognitive vision platform answers or not the requirements defined in chapter 3. In the second part, we present perspectives to improve the platform, in particular with machine learning techniques.

7.1 Review of the Cognitive Vision Platform and Contributions

7.1.1 A Reusable Cognitive Vision Platform

Our cognitive vision platform is a set of reusable tools for the design of semantic image interpretation systems.

1. **Re-usability using problem solving methods**
   Our approach based on knowledge based systems and problem solving methods enables the separation between the reasoning and the a priori knowledge used by it. This separation enables the re-usability of the reasoning strategies for different application domains. For each sub-problem of semantic image interpretation, the cognitive vision platform proposes (e.g. chapter 5):

   - a dedicated application independent engine
   - a generic conceptual knowledge model

   The modularity of the proposed cognitive vision platform enables the separation of the different types of expertise. The model proposed for each sub-problem provides for experts of the corresponding sub-problems a framework and a clear description of the structure of the knowledge involved in their sub-problem. By using the cognitive vision platform, the design of a semantic image interpretation system for a particular application is a cooperative knowledge acquisition work between application domain experts, cognitive vision experts and image processing experts. The engines are reused and the different experts have just to build their specific knowledge base.

2. **Re-usability using ontological engineering**
   As problem solving methods play an important role in the reuse of reasoning strategies, ontologies play an important role in knowledge sharing and reuse. The cognitive vision platform proposes two specific ontologies (i.e.. chapter 4):

   - A **visual concept ontology** which is application independent and which can be reused across application domains. We used a visual concept ontology proposed by [Maillot et al., 2003a] to reduce the domain knowledge acquisition bottleneck. Indeed, it provides a set of generic terms to describe concepts of a domain. In the cognitive vision platform, the **visual concept ontology** is also used as a common corpus for the inter-operability between the semantic interpretation module and the visual data management module. Moreover, additional visual concepts were added:
     - color concepts to manage grey level images
     - shape concepts, less geometric, to cope with the description of natural complex objects
     - spatial relation concepts (forming an independent ontology called **spatial relation ontology**) for the generic description of the spatial relations between objects
     - spatial structure concepts to describe spatial object configurations with a fixed structure
   - An **image processing ontology** which is application independent but dependent on the data structures of a library of programs. Similarly to the visual concept ontology, the image processing ontology enables the interoperability between the visual data management module and the program supervision module. This image processing ontology also reduces the program supervision knowledge acquisition bottleneck by providing a set of generic image processing functionalities to structure the program supervision knowledge base and a set of generic image data used to describe arguments of supervision operators.
     Our **image processing ontology** is not complete and additional terms could be added to improve it.

7.1.1.1 Semantic Interpretation

Concerning the semantic interpretation module of the platform, we were inspired by a similar approach designed in our team for single object recognition [Thonnat, 2002]. Our contribution consisted in upgrading and advancing this approach to situation recognition. Situations are fixed spatial configurations of multiple objects representing abstract notions.

The generic conceptual knowledge model proposed in [Thonnat, 2002] was modified:

- to cope with situations by a formalism to represent spatial configurations of objects,
- to integrate the use of the visual concept ontology in the knowledge representation formalism.

The semantic interpretation engine was inspired by the classification engine proposed in [Thonnat, 2002] but upgraded to be goal oriented. It is a request including the end user goal and the current context which conditions the semantic interpretation process. Moreover, the proposed semantic interpretation engine is based on an hypothesis and test cycle.

7.1.1.2 Visual Data Management

The visual data management module is the main contribution of this thesis. The main idea is to consider the problem of the visual data management as a problem as such, which has its proper expertise and its proper reasoning strategies. The visual data management problem includes symbol grounding (connection between the symbols used at the interpretation level and the perceptual data provided by the sensors) and spatial reasoning. We have outlined the main concepts and the main functionalities involved in a visual data management process. Our knowledge based approach proposes:

- An application independent visual data management engine based on top down and bottom up reasoning strategies.
- A visual data management knowledge based model

Our approach enables the capitalization of the knowledge about visual data management which is an important result.

7.1.1.3 Program Supervision

Concerning the program supervision module, our contribution was smaller. Our main contribution was to use and to integrate program supervision techniques to manage the image processing sub-problem of semantic image interpretation. We have used existing techniques to build this module. Indeed, program supervision by a knowledge based approach is a great expertise of the ORION team [Thonnat and Moisan, 1995]. In particular, we have used the particular program supervision engine named Pegase+. Some minor changes were made to support the image processing ontology and the interoperability with the data management module.

7.1.2 A convenient cognitive vision platform

The principle of the use of the cognitive vision platform states that experts of a specific domain are the best persons to deal with their domain and to make explicit their knowledge. The modularity of the cognitive vision platform enables experts to contribute only at their level of expertise. Nevertheless, even for experts, the building of knowledge bases for the different sub-problems is still an effort. The cognitive vision platform makes easier the knowledge acquisition process by providing specific knowledge description languages. The explicitation of knowledge is made is a natural manner with natural languages.

- The YAKL language developed in the ORION team for the program supervision knowledge acquisition
- The SIKL++ language derived from the SIKL language for the application domain and the visual data management knowledge.

7.1.3 The ROSESIM system

Using the cognitive vision platform, we have built a semantic interpretation system dedicated to rose disease diagnosis. We have used the three application independent engines described in the chapter 5 and we have built three knowledge bases: a knowledge base on rose disease early symptoms, a visual data management knowledge base and a program supervision knowledge base. We have also developed image processing programs to complete the PANDORE image processing library to add missing functionalities. In particular, we have developed robust image processing algorithms able to extract ridges on images.

7.1.4 A minimal semantic image interpretation system

By providing a minimal semantic image interpretation system, we go a step further in reusability. Indeed, the minimal semantic image interpretation system provides a minimal visual data management knowledge base and a minimal program supervision knowledge base. The design of a particular application image interpretation system is restricted to the building of the interpretation knowledge base and to the application dependent part of the visual data management knowledge base. This data management knowledge represents the symbol grounding relation between the instances of visual concepts used to describe domain classes and numerical image descriptors. It is one of the drawback of the current version of the cognitive vision platform: the domain knowledge expert is not always able to provide this visual data management knowledge. We propose in the following section a method to cover this drawback using machine learning techniques.

If the particular application required complex image processing programs or complex spatial reasoning, a particular visual data management knowledge base and a particular program supervision knowledge base can be built incrementally from the minimal ones. As a consequence, the use of a minimal system enables to considerably restrict the design-cycle time of a particular semantic image interpretation system.

From the point of view of the end user, a particular semantic interpretation system built with the cognitive vision platform has the properties of :

- **flexibility and self adaptation**
  At the level of semantic interpretation and visual data management, these properties result from the context awareness of the system. The explicit representation of the domain context and the acquisition context enables to take into account the conditions of the environment during the processing.

  Program supervision techniques with its mechanisms of initialization, choice, assessment and repair enable the self-adaptation and the autonomy of the image processing. This self-adaptation is made according to the image processing request built by the visual data management system.

- **convenience**
  The end user of a particular semantic image interpretation system has only to provide a domain request with input data and the current context of the particular application. The particular image interpretation system is more or less autonomous. The end user is not burdened by providing expertise he does not own or with technical details. Nevertheless, the end user may have to interactively assess the results of the image processing module (evaluation criteria of the visual data management knowledge base).

7.2 Short Term Perspectives

7.2.1 Improvement of the knowledge acquisition process

7.2.1.1 An interactive domain knowledge acquisition

To make more convenient the domain knowledge acquisition process than the writing of a knowledge base using the description language SIKL++, we have planned to make it interactive. We propose to integrate the tools Ontovis [Maillot et al., 2003a] to the cognitive vision platform. Currently, Ontovis outputs knowledge bases in XML and annotated images. The aim is to make an interface between the outputs of Ontovis in XML and the SIKL++ language. This interactive domain knowledge acquisition should be more attractive and easier for experts not trained in writing knowledge bases.

7.2.1.2 Learning of the symbol grounding relation between symbolic and image data

Currently, one of the big drawback of the current cognitive vision platform is that the symbol grounding relations between (1) the instances of the visual concepts associated to a domain class and (2) the numerical image descriptors, are hand coded. This symbol grounding knowledge is hand coded by the application domain expert during the domain knowledge acquisition. If visual concepts are really close to real physically and quantitatively measurable properties, the application domain expert can provide this knowledge. Most of the time, the creation of the symbol grounding relation is not in the skills of application domain experts. As shown in [Maillot et al., 2003a], machine learning techniques are good methods to answer to this big drawback and to reduce the semantic gap between symbolic data and numerical data. This work is the subject of the Ph.D. thesis of Nicolas Maillot in the Orion Team. This work proposes to learn, from representative samples, visual concept detectors. The visual concept learning is composed of three steps: training set building, feature selection and training.

We propose to add the visual concept learning module in the cognitive vision platform and to use it for the domain dependent knowledge part acquisition. The knowledge acquisition process is then composed of two steps :

- The domain knowledge acquisition guided by the visual concept ontology to build the domain knowledge base and to build a set of annotated and manually segmented images.
  This part of the knowledge acquisition is performed by domain experts and is depicted in figure 7.1.

- The automatic building of the symbol grounding relations between the visual concepts describing domain classes and image descriptor values. This part of the knowl-

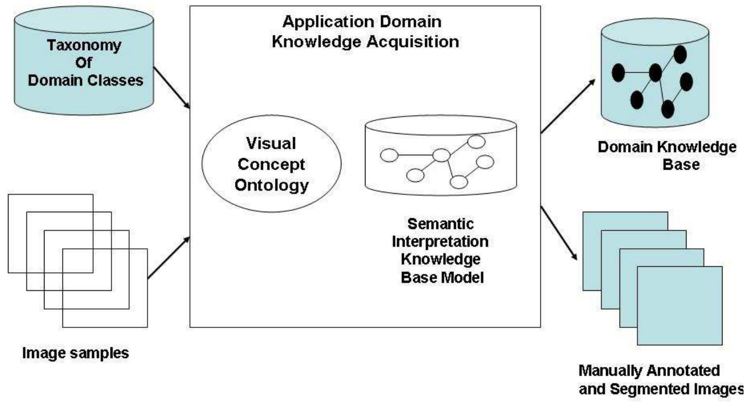


Figure 7.1: The domain knowledge acquisition process

edge acquisition is described in figure 7.2. In [Maillot et al., 2003a], the result of this learning phase is represented as an **augmented domain knowledge base**. We state that this knowledge does not belong to the domain knowledge base but to the visual data management knowledge base.

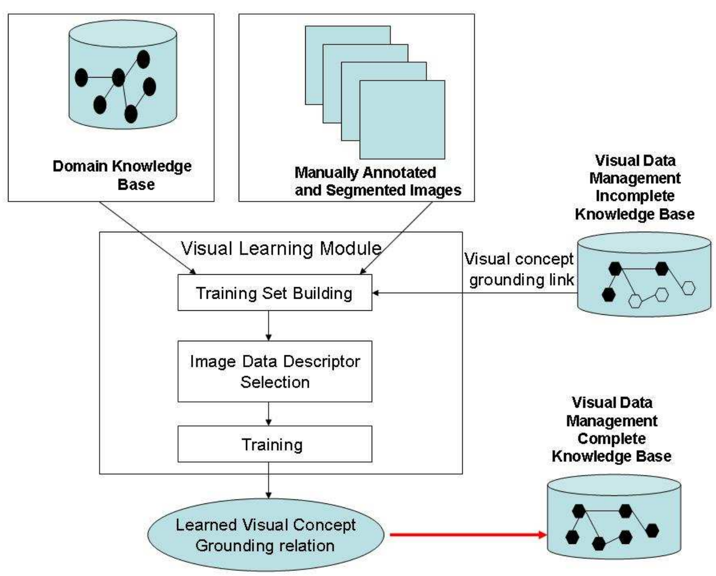


Figure 7.2: A visual concept ontology based learning method to automatically built symbol grounding relations

7.2.2 Machine Learning Techniques for Image Segmentation

In his master thesis [Martin, 2004] in the Orion team, Vincent Martin proposes supervised machine learning techniques for the image segmentation process. In particular, he proposes to use machine learning for the automatic tuning of segmentation algorithms. Another perspective to improve the robustness and the autonomy of semantic image interpretation systems is to integrate these works in the cognitive vision platform. With such a method, the program supervision knowledge is reduced. Initialization criteria are not necessary anymore.

7.2.3 Automatic Evaluation of Image processing results

Another drawback of the proposed minimal system concerns the evaluation of the image processing results by the visual data management system. Indeed, we have stressed the importance of the evaluation of the image processing results at this level of the interpretation level but the proposed solution is based on an interaction with the end user. Our minimal system contains only one evaluation criteria. To achieve an automatic evaluation, additional evaluation criteria could be added in the visual data management knowledge base. Works on the evaluation of image processing results according to a high level goal have to be studied to build these evaluation criteria.

7.3 Long Term Perspectives

7.3.1 Introduction of the Temporal Dimension

A long term objective for the evolution of the cognitive vision platform is to take into account the temporal dimension. Currently, the cognitive vision platform is a tool for the building of semantic interpretation system of purely static images. It does not enable the recognition of events or scenarios. As the Orion team has a great experience in semantic video interpretation, we aim at extending our cognitive vision platform in a generic and reusable framework for the semantic interpretation of static images as well as for sequences of images.

7.3.2 An opportunistic behavior

Some improvements can also be made by the integration of opportunistic reasoning. Currently, all the components of the cognitive vision platform are guided by requests coming from higher level module. To make the platform more flexible, the modules could also be directed by events as in [Sandakly, 1995]. Moreover, the global process could be initialized by the low levels.

7.3.3 Dynamic knowledge bases

In the framework of the proposed cognitive vision platform, we have no answer to the problem of the *close world assumption*. Indeed, the knowledge bases are static and they can not be modified during the reasoning to take into account new contexts. To make the cognitive vision platform more adaptable, evolving knowledge bases, able to adapt themselves by the creation of new concepts or by the modification of existing concepts, could be a great improvement for the cognitive vision platform.

Chapter 8

French Extended Abstract

This chapter presents a translation in french of the introduction and of the conclusion and gives a description of the cognitive vision platform.

8.1 Introduction

Dans cette thèse, nous nous intéressons au problème de l'interprétation sémantique d'images. Il s'agit d'un problème de perception visuelle, c'est à dire la perception du monde réel par des capteurs visuels (système visuel humain, caméra, ...). Dans nos travaux, nous abordons ce problème sous l'angle de la **construction de systèmes automatiques d'interprétation d'images**.

8.1.1 Problématique

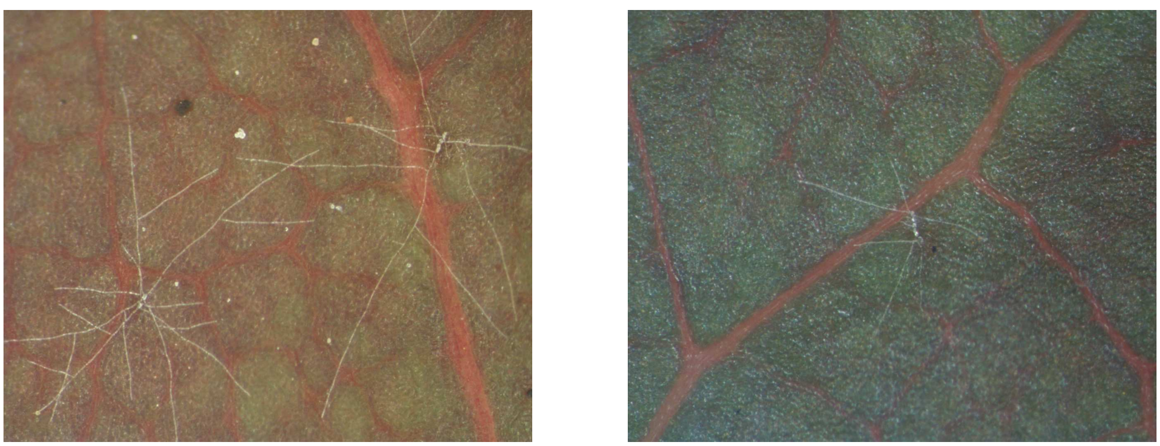


Figure 8.1: Deux exemples d'images à interpréter

Le problème de l'interprétation sémantique d'images peut être illustré très simplement à l'aide des deux images de la figure 8.1. Si on regarde ces deux images, il s'agit de répondre à la question: quel est le contenu sémantique de ces deux images ou de manière plus simple, que représentent t'elles? Les réponses peuvent être multiples. Par exemple pour l'image gauche de la figure 8.1, des interprétations possibles sont:

- deux réseaux de fines lignes blanches sur un fond vert texturé,

- des anomalies sur un objet végétal,
- deux apparitions d'un champignon microscopique sur une feuille végétale,
- deux apparitions très précoces d'oïdium sur une jeune feuille de rosier.

Cet exemple montre que l'interprétation sémantique d'une image dépend de la connaissance à priori de l'interpréteur. L'image droite de la figure 8.1 est très similaire à celle de gauche et peut être interprétée de la même manière. Cependant, des interprétations possibles peuvent aussi être:

- un réseau d'épaisses lignes rouges sur un fond vert,
- un réseau de routes dans une image aérienne,
- un réseau de routes dans une zone forestière.

Sans aucune autre information que l'image en elle même, on peut considérer que ces interprétations sont correctes. Il n'existe pas de réponse unique au problème de l'interprétation sémantique d'images.

- **Importance de la connaissance a priori**
  Nous avons montré, avec les images de la figure 8.1, que la réponse à un problème d'interprétation sémantique dépend très fortement du niveau de connaissance a priori de l'interpréteur. La sémantique n'est pas dans l'image elle même. En effet, sans aucune connaissance sur les pathologies végétales et en particulier les pathologies des rosiers, il est impossible d'interpréter les réseaux de lignes blanches (figure 8.1) comme une apparition précoce de l'oïdium du rosier. Par conséquent, si l'interprétation sémantique d'une image **dépend très fortement de la connaissance a priori de l'interpréteur**, il est donc nécessaire qu'un système automatique d'interprétation d'images dispose d'une connaissance plus ou moins sophistiquée.

- **Importance de l'information contextuelle**
  Par information textuelle, nous faisons référence à toute l'information additionnelle, non visuelle, qui peut influencer la manière dont la scène est perçue. Par exemple, sans savoir que les images de la figure 8.1 sont des images biologiques microscopiques, les interprétations faisant référence à des routes dans des images aériennes peuvent être prises en considération. Le terme *biologique* fait référence au contexte du domaine d'application tandis que le terme *microscopique* fait référence à l'acquisition des images. L'information contextuelle est donc primordiale pour le problème de l'interprétation sémantique d'images et la représentation et l'utilisation du contexte dans un système d'interprétation d'images peut améliorer de manière importante l'efficacité et la performance d'un système.

- **Importance du but de l'interprétation**
  L'interprétation sémantique d'une scène est un problème dirigé par le but. En effet, reprenons l'exemple de l'image de gauche de la figure 8.1. Si le but de l'interprétation est de détecter la présence ou non de champignons sur une feuille de rose alors il suffira d'extraire les fines lignes blanches dans l'image. Par contre, si le but de l'interprétation est de faire un bilan sanitaire complet de la feuille alors il faudra extraire et interpréter tous les objets présents dans l'image (les fines lignes blanches et les petites régions blanches). La prise en compte du but de l'interprétation est importante car elle permet de se focaliser sur ce qui est pertinent pour ce but. La

stratégie d'interprétation du système doit donc être guidée par son but : il s'agira de trouver les objets ou les informations visuelles qui aident au mieux à la réalisation du but.

- **Importance de l'information sur la scène**
  Dans le monde réel, un objet est rarement isolé. Il existe dans le contexte de son environnement. Il existe une relation très forte entre un objet et son environnement et cette relation est très importante pour l'interprétation. La connaissance sur la scène, c'est à dire sur les différents objets pouvant coexister dans une scène et sur les relations spatiales entre ces différents objets est donc primordiale. Nous utilisons de manière considérable les relations spatiales pour décrire les objets, pour les détecter et les reconnaître. Dans [Neumann and Weiss, 2003], l' interprétation sémantique est définie comme un problème "au delà de la reconnaissance d'objets isolés". Elle implique la reconnaissance de situations, d'évènements ou scénarios pour des scènes dynamiques.

- **Importance d'un traitement intelligent des images**
  Les deux images de la figure 8.1 montrent que plusieurs apparences et plusieurs contextes (les différents supports végétaux) peuvent exister pour la même classe sémantique d'objets (l'oïdium). Un programme spécialisé, même sophistiqué, n'est pas suffisant. Les traitements bas niveau doivent être flexibles et doivent pouvoir s'adapter à différents contextes.

La construction d'un système d'interprétation automatique d'images consiste à doter les ordinateurs d'un système visuel leur permettant de percevoir et de comprendre leur environnement. Pendant les cinquante dernières années, plusieurs approches différentes ont été proposées pour résoudre ce problème. Elles ont donné lieu à un grand nombre de systèmes d'interprétation, avec autant de modélisations et de motivations différentes. Cependant le constat général commun sur ces systèmes est que, quelle que soit leur approche de construction, ils manquent souvent de robustesse, d'adaptabilité et ils sont souvent très dépendants du domaine d'application. Après cinquante années de recherche dynamique, il n'existe toujours pas de système générique d'interprétation d'images capable de traiter des tâches et des images diverses comme le système visuel humain.

La **vision cognitive** est née de ce constat. C'est une nouvelle discipline de recherche qui rassemble des domaines de recherche variés, entre autres, la vision par ordinateur, la reconnaissance de forme, l'intelligence artificielle, la robotique, l'apprentissage et les sciences cognitives. Un plan de recherche sur 20 ans a été proposé par le réseau de recherche européen ECVision[1] dans [Vernon, 2004]. L'idée principale de la vision cognitive est de rendre les systèmes de vision plus robustes, plus résistants et plus adaptables en les dotant de facultés cognitives : "savoir", "comprendre", "raisonner" et " apprendre". Un système de vision cognitive est *un système capable d'apprentissage, d'adaptation, de faire un choix entre plusieurs alternatives et de développer des nouvelles stratégies d'analyse et d'interprétation.* Un système de vision cognitive doit s'adapter à la réalisation d'un grand nombre de tâches et il doit être capable de s'adapter à son environnement courant. Un grande motivation de la vision cognitive est de concevoir des systèmes pouvant évoluer dans et en forte interaction avec le monde réel.

[1] The European research Network for Cognitive Computer Vision Systems, www.ecvsion.org

8.1.1.1 Notre objectif

L'objectif de cette thèse est de faire des avancées dans le domaine de la vision cognitive par la conception d'une plate forme générique et réutilisable pour la résolution de problèmes d'interprétation sémantique d'images. Il ne s'agit pas de construire un système complet spécifique à une application mais plutôt de fournir des outils génériques et réutilisables pour la conception de tels systèmes.

La plate forme de vision cognitive proposée est un environnement unifié pour la construction de systèmes complets d'interprétation d'images. Nous nous intéressons à la fois aux problèmes de génie logiciel et aux problèmes cognitifs impliqués dans la conception d'une telle plate forme. En particulier, nous nous sommes focalisés sur les propriétés de réutilisabilité et de commodité pour l'utilisateur.

La plate forme proposée consiste en :

- Une architecture fonctionnelle minimale pour les systèmes d'interprétation sémantique d'images. Cette architecture définit quels sont les modules nécessaires pour un système d'interprétation, quel est leur rôle et quelles sont leurs interactions.

- Une formalisation et une explicitation des différents types de connaissances et de raisonnements impliqués dans le problème global de l'interprétation sémantique d'images.

- Un environnement de développement pour la conception de systèmes d'interprétation.

L'objectif de cette thèse est double:

- Premièrement, il s'agit de définir et de concevoir cette plate forme de vision cognitive pour faciliter la construction de systèmes d'interprétation sémantique d'images.

- Un second travail consistera à valider et à tester la plate forme avec une application concrète: la reconnaissance d'organismes biologiques dans leur environnement naturel.

8.1.1.2 Contexte de l'étude

Ces travaux de thèse prennent place dans les thématiques de recherche de l'équipe ORION de l'INRIA Sophia Antipolis. Orion est une équipe pluridisciplinaire à la frontière des domaines de la vision par ordinateur, des systèmes à base de connaissances et du génie logiciel. Ces travaux ont donc bénéficié d'une grande expertise dans les domaines de l'interprétation d'images et la reconnaissance d'objets complexes et dans les domaines du génie logiciel pour la réutilisation de systèmes intelligents.

En particulier, plusieurs travaux de l'équipe ORION ont prouvé l'efficacité de l'explicitation des connaissances a priori pour la résolution de problèmes complexes d'analyse d'images. Les problèmes abordés ont été l'automatisation de l'utilisation d'une bibliothèque de programmes de traitement d'images [Moisan and Thonnat, 1995], [Moisan and Thonnat, 2000] et la reconnaissance automatique d'objets [Thonnat, 2002]. Dans [Ossola, 1996], une approche pour la conception de systèmes de reconnaissance automatique d'objets isolés complexes est proposée. Elle se base sur la coopération de systèmes à base de connaissances. Nos travaux de thèse s'inscrivent dans la suite de ces travaux.

De plus, la conception de systèmes intelligents réutilisables est un autre thème de recherche actif de l'équipe ORION. Des méthodes de résolution de problèmes ont été utilisées pour concevoir des moteurs indépendants d'une expertise particulière, mais cependant dédiés à une classe de problèmes. Ils facilitent la construction de systèmes à base de connaissances. En particulier, la plate-forme logicielle LAMA [Moisan, 1998] fournit un environnement unifié pour construire non seulement des bases de connaissances expertes, mais aussi des variantes de moteurs et des outils annexes. Elle regroupe des boîtes à outils pour construire et adapter tous les éléments logiciels nécessaires à la réalisation de systèmes à base de connaissances. Nous avons utilisé LAMA pour l'implémentation de la plate forme de vision cognitive.

8.1.1.3 Approche proposée

Le problème complexe de l'interprétation sémantique d'images peut être divisé en trois sous problèmes plus faciles à résoudre en tant que problème indépendant:

- **Le traitement d'images**, pour l'extraction et la description numérique des objets d'intérêts dans l'image.
- **La mise en correspondance** entre les représentations de haut niveau *qualitatives* de la scène et l'information *numérique* extraite des images.
- **L'interprétation sémantique**, c'est à dire la compréhension de la scène à l'aide de la terminologie du domaine d'application. La scène est décrite avec des concepts propres au domaine d'application.

Chacun de ces sous problèmes est un problème en lui même possédant une expertise propre. Pour gérer et pour séparer les différentes sources de connaissances et les différents types de raisonnements impliqués, nous proposons une architecture minimale distribuée qui se base sur la coopération de **trois systèmes à base de connaissances**. Chaque SBC (Système à Base de Connaissances) est dédié à l'un des sous problèmes de l'interprétation sémantique d'images. L'architecture est donc composée d'un **SBC dédié au pilotage de programmes de traitement d'images**, d'un **SBC de gestion de données visuelles** dédié à l'ancrage de symboles et au raisonnement spatial et enfin d'un **SBC d'interprétation**.

Nous nous intéressons à des solutions génériques et réutilisables. Les trois sous problèmes ont donc été étudiés sous les angles du génie logiciel et de l'ingénierie des connaissances. Pour chaque sous problème, nous proposons un modèle composé d'un moteur spécifique et d'un cadre conceptuel pour modéliser la connaissance propre au sous problème. La plate forme de vision cognitive se compose de ces trois modèles et de deux ontologies pour faciliter leur interopérabilité.

Pour valider la plate forme de vision cognitive, nous avons choisi un problème d'interprétation sémantique complexe: le diagnostic précoce des maladies foliaires du rosier de serre. Ce travail a été effectué avec l'URIH (Unité de Recherches Intégrées en Horticulture) de l'INRA (Institut National de Recherche en Agronomie) de Sophia Antipolis.

8.1.1.4 Plan du mémoire

Ce document est organisé de la manière suivante:

- Le **chapitre 2** dresse un état de l'art non exhaustif sur les systèmes d'interprétation d'images. Nous nous focaliserons sur les travaux jugés les plus importants et les plus pertinents pour nos travaux. Différents courants de pensée pour la conception de systèmes d'interprétation sont présentés. Nous analysons ensuite différents systèmes d'interprétation d'images et nous introduisons la discipline de la vision cognitive.

- Le **chapitre 3** décrit de manière précise nos motivations et propose une description globale de la plate forme de vision cognitive: l'approche choisie, sa composition et son principe d'utilisation.
- Dans le **chapitre 4**, nous présentons succinctement le génie ontologique et son utilisation pour la plate forme de vision cognitive. En particulier, nous présentons les deux ontologies utilisées pour l'interopérabilité entre les différents composants de la plate forme: une ontologie de concepts visuels et une ontologie de traitement d'images.
- Le **chapitre 5** est dédié à la description détaillée des différents composants de la plate forme. Pour chaque composant de la plate forme et donc pour chaque sous problème de l'interprétation sémantique d'images nous proposons une base conceptuelle pour la représentation des connaissances propres au sous problème et un moteur générique pour la résolution du sous problème.
- Le **chapitre 6** décrit l'application concrète utilisée pour valider la plate forme de vision cognitive proposée. Nous présentons d'abord la problématique et les objectifs biologiques. Après un bref état de l'art sur le problème biologique nous illustrons sa résolution à l'aide de la plate forme de vision cognitive. En particulier, nous montrons comment nous construisons un système dédié à l'interprétation sémantique d'images microscopiques de feuilles de rosiers à l'aide de la plate forme de vision cognitive.

8.2 Description globale de la plate forme de vision cognitive

Cette section présente, de manière globale, la plate forme de vision cognitive. Dans une première partie nous rappelons nos objectifs et nous définissons une liste de propriétés devant être satisfaites par la plate forme de vision cognitive. Nous présentons et justifions, dans la section suivante, les solutions choisies.

8.2.1 Motivations

Notre objectif est de faire des avancées dans le domaine de la vision cognitive en concevant une plate forme générique et réutilisable pour faciliter la construction de systèmes d'interprétation sémantique d'images. La plate forme proposée est un environnement unifié pour la conception de tels systèmes. Elle peut être définie à la fois comme une architecture fonctionnelle minimale qui définit quels sont les modules nécessaires pour un système d'interprétation sémantique d'images et à la fois comme un environnement de développement dédié à la construction de systèmes d'interprétation d'images. Nous nous intéressons donc non seulement aux problèmes cognitifs mais aussi aux problèmes de génie logiciel impliqués dans la conception d'une telle plate forme.

Par *cognitifs*, nous entendons que la plate forme doit proposer des solutions intelligentes pour chacune des fonctionnalités d'un système d'interprétation d'images, c'est à dire pour l'interprétation, la reconnaissance et la détection des objets d'intérêts de l'image.

En tant qu'environnement unifié pour la construction de systèmes d'interprétation sémantique d'images, la plate forme proposée doit satisfaire un certain nombre de propriétés. Ces propriétés sont:

1. **La réutilisabilité**
   La réutilisabilité est une propriété fondamentale du génie logiciel. Elle représente la propriété d'un logiciel d'être tout ou partiellement réutilisée pour de nouvelles applications. La conception de solutions indépendantes du domaine d'application est un moyen de favoriser cette réutilisabilité. Il s'agit donc de déterminer quels sont les connaissances génériques d'un système d'interprétation.

2. **La modularité**
   La propriété de modularité fait référence au principe du "*Diviser pour régner*", c'est à dire la décomposition d'un problème complexe en sous problèmes (modules) plus simples et pouvant être traités en tant que problèmes indépendants. Cette propriété permet un développement plus facile, elle facilite la maintenance et l'évolution et elle permet aussi d'avoir des modules réutilisables.

3. **La facilité d'utilisation**
   Cette propriété est un requis à la fois pour l'utilisation de la plate forme de vision cognitive et à la fois pour l'utilisation des systèmes d'interprétation sémantique construits à l'aide de la plate forme. Cette propriété implique de fournir des outils pour que les différents experts interviennent à leur niveau et seulement à leur niveau de compétences. En particulier, ils doivent pouvoir exprimer leur connaissance d'une manière naturelle. De plus, du point de vue des systèmes construits avec la plate forme, ces derniers doivent avoir un degré d'autonomie suffisant pour pouvoir être utilisés par un utilisateur final pas nécessairement expérimenté.

Comme nous l'avons mentionné précédemment, notre objectif est de faire des avancées dans le domaine de la vision cognitive. Notre objectif n'est pas de proposer une plate forme de vision cognitive complète et générique pour l'ensemble des problèmes d'interprétation sémantique d'images. Cet objectif est l'objectif ultime à long terme de la vision cognitive et il serait présomptueux de penser y répondre dans le cadre d'une thèse. Nous proposons une plate forme de vision cognitive qui a bien sûr une portée limitée. Nous avons donc posé un ensemble d'hypothèses fortes pour faciliter la conception de cette plate forme.

- Nous nous intéressons à des scènes statiques pour lesquelles l'information 3D n'est pas primordiale.
- Les scènes étudiées sont perçues par un unique dispositif d'acquisition d'images.
- Les domaines d'application concernés sont des domaines pour lesquels il existe une connaissance propre et des experts.
- Bien qu'une des fonctionnalités primordiales d'un système de vision cognitive, nous ne nous sommes pas intéressés aux techniques d'apprentissage dans le cadre de ce travail. Nous décrirons néanmoins comment certaines techniques d'apprentissage développées dans l'équipe Orion [Maillot et al., 2004a] pourront être intégrées dans la plate forme.

8.2.2 Approche proposée: description globale de la plate forme

Nos travaux s'inscrivent dans la suite des travaux de notre laboratoire d'accueil dans le domaine de l'interprétation d'images. Ils ont bénéficié de l'expérience de l'équipe Orion dans ce domaine. En particulier, dans [Ossola, 1996], une architecture distribuée se basant sur la coopération de deux systèmes à base de connaissances est proposée pour la reconnaissance d'objets isolés complexes. Cette expérience nous a permis d'identifier les difficultés du problème d'interprétation sémantique d'images et les limites de cette première architecture. En particulier, nous lui reprochons la non prise en compte de la connaissance sur la scène, l'absence de raisonnement spatial et l'encapsulation du problème de mise en correspondance entre les données images et les symboles dans des règles d'abstraction de données dépendantes du domaine d'application.

Du point de vue architectural, nous proposons une architecture distribuée qui se base sur la coopération de trois systèmes à base de connaissances:

- **Un système dédié à l'interprétation sémantique**, c'est à dire dédié à l'interprétation sémantique de la scène à l'aide de la terminologie du domaine d'application.
- **Un système dédié à la mise en correspondance** entre les données issues des capteurs et les symboles servant à décrire de manière abstraite et qualitative les objets et les situations du monde réel.
- **Un système de pilotage de programmes de traitement d'images**.

Un système à base de connaissances est un logiciel qui permet de traiter des problèmes complexes en se servant de connaissances décrites de manière déclarative. Les caractéristiques majeures des systèmes à base de connaissance sont:

- La séparation de la connaissance et du raisonnement,
- La représentation d'une connaissance experte et la reproduction du raisonnement des experts du domaine,
- Leur facilité de maintenance et d'évolution,
- La gestion de l'incertitude.

La plate forme de vision cognitive est donc composée de trois modules. Dans l'optique de la réutilisabilité, chaque module est un générateur de systèmes à base de connaissances pour chacun de ses sous problèmes, c'est à dire un moteur générique et un modèle conceptuel de représentation des connaissances. La gestion de la communication et l'interopérabilité entre les différents modules de la plate forme sont assurées par deux ontologies. L'architecture globale de la plate forme de vision cognitive est représentée dans la figure 8.2.

8.2.3 Apport du génie ontologique pour la plate forme de vision cognitive

La modularité de l'architecture proposée implique une gestion de la communication et du partage de l'information (connaissances et données). Pour faciliter la communication et le partage d'informations entre les différents modules de la plate forme mais aussi pour faciliter l'acquisition de la connaissance, nous proposons d'utiliser de récents progrès en ingénierie des connaissances: le génie ontologique. Une très bonne introduction sur le génie ontologique peut être trouvée dans [Gandon, 2002]. En particulier, la plate forme de vision cognitive se compose de deux ontologies:

- une ontologie de concepts visuels
- une ontologie de concepts de traitement d'images

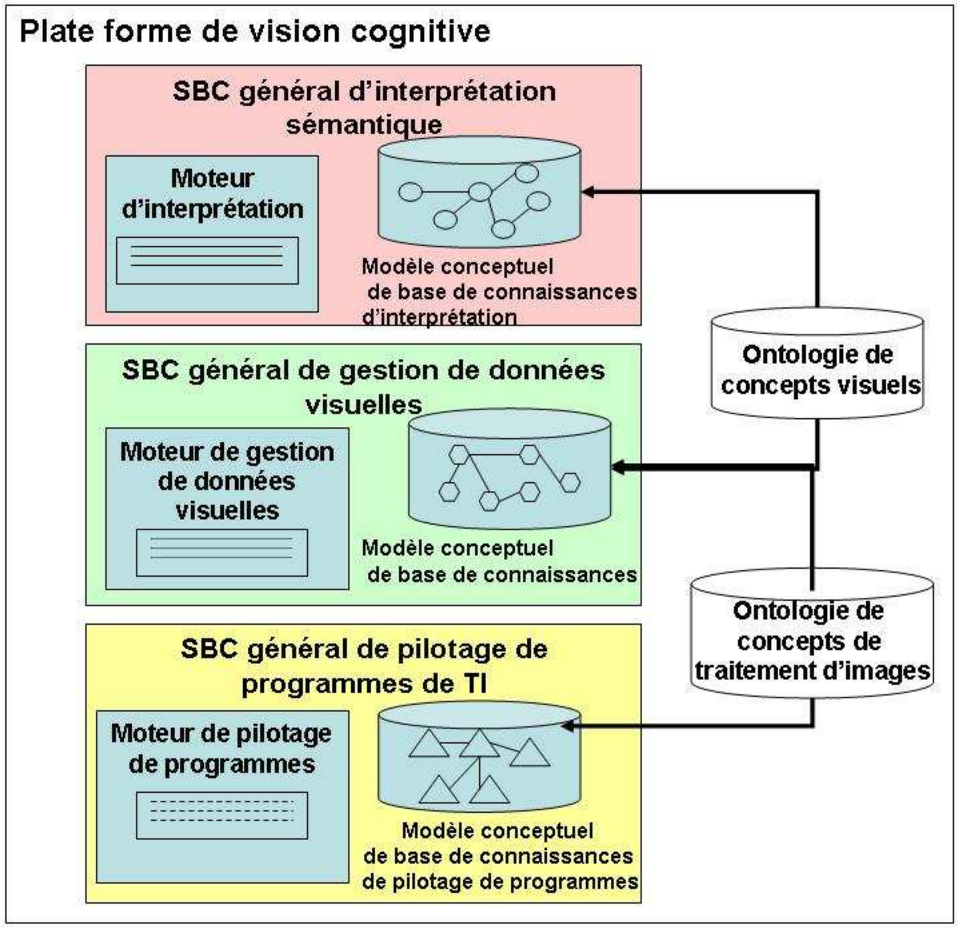


Figure 8.2: Architecture globale de la plate forme de vision cognitive

8.2.3.1 Une ontologie de concepts visuels

Pour faciliter la communication entre le module d'interprétation et le module de gestion de données visuelles, nous proposons d'utiliser une ontologie de concepts visuels. L'ontologie de concepts visuels utilisée repose sur une ontologie existante construite dans l'équipe Orion par Nicolas Maillot [Maillot et al., 2003a]. Elle est née du constat que les experts de différents domaines utilisent et partagent un vocabulaire visuel générique pour décrire les concepts de leur domaine. L'ontologie proposée est une hiérarchie de concepts qui s'organise en:

- concepts spatio-temporels (concepts géométriques) (circulaire, rectangulaire,...)
- concepts de couleur (teinte, bleu, clair,...)
- concepts de texture (texture régulière, texture maillée,...)

Nous avons adapté cette ontologie au contexte de la plate forme de vision cognitive. En particulier, nous ne prenons pas en compte les concepts temporels ou les concepts géométriques 3D de cette ontologie. Des concepts spatiaux permettant de décrire les formes de manière qualitative et pas seulement géométrique ont été ajoutés à l'ontologie de concepts visuels.

De plus, des concepts permettant de décrire des relations spatiales binaires et des configurations spatiales ont aussi été ajoutés à l'ontologie existante. Cette ontologie:

- facilite l'acquisition de la connaissance sémantique. En effet, elle permet de guider la description des différents concepts du domaine d'application en fournissant aux experts un vocabulaire de description visuelle prédéfini.
- En tant que corpus commun entre le module d'interprétation et le module de gestion de données visuelles, elle facilite la communication entre ces deux modules.
- Elle permet de réduire le fossé sémantique entre les concepts du monde réel et les données images.

8.2.3.2 Une ontologie de concepts de traitement d'images

De manière similaire une ontologie de concepts de traitement d'images permet de faciliter la communication entre le module de gestion de données visuelles et le module de pilotage de programmes de traitement d'images. Elle est définie comme l'ensemble des concepts génériques qui permettent de décrire les problèmes de traitement d'images et leur résultats. Cette ontologie est une hiérarchie de concepts structurée en:

- concepts d'entités image: ils représentent les différentes structures de données (ou primitives) qui peuvent être extraites des images (région, contour,...)
- concepts de descripteurs d'images: ils représentent les différentes caractéristiques pouvant être mesurées dans l'image (R, G, B, compacité, aire, excentricité,...)
- concepts de fonctionnalités de traitement d'images: il représentent des buts génériques de traitement d'images (segmentation, extraction de caractéristiques, ...)

Les principaux apports de l'ontologie de concepts de traitement d'images pour la plate forme de vision cognitive sont:

- l'interopérabilité entre le module de gestion des données visuelles et le module de pilotage de programme de traitement d'images. Elle est en effet un corpus commun entre ces deux modules.
- l'acquisition des connaissances de pilotage de programmes de traitement d'images est guidée par l'ensemble des termes de l'ontologie de concepts de traitement d'images.

8.2.4 Description détaillée de la plate forme de vision cognitive

Nous avons vu précédemment que la plate forme de vision cognitive est composée de trois modules indépendants. Chaque module est dédié à l'un des sous problèmes du problème complexe de l'interprétation sémantique d'images. Nous avons choisi de nous baser sur le développement d'outils dédiés à une classe de problèmes mais indépendants des applications de ce problème.

8.2.4.1 Interprétation sémantique

Le rôle du module d'interprétation sémantique est de donner un sens sémantique à la description symbolique perçue de la scène. Cette sémantique fait référence à l'expertise et à la terminologie du domaine d'application. Nous pensons que les experts du domaine d'application sont les personnes les mieux placées pour reconnaître les objets de leur domaine. Le but du module d'interprétation est donc d'imiter les experts du domaine en utilisant leur propre taxonomie et en reproduisant leur raisonnement.

Le module d'interprétation sémantique contient toute la connaissance propre au domaine d'application stockée dans une base de connaissance d'interprétation sémantique.

1. **Modèle de représentation de la connaissance pour l'interprétation sémantique**
   La base de connaissances d'interprétation sémantique contient toute la connaissance propre au domaine d'application. Elle est écrite par les experts du domaine d'application. L'acquisition des connaissances est guidée par l'ontologie de concepts visuels et l'ontologie de relations spatiales. Le contenu de la base de la connaissance est donc propre au domaine d'application mais les concepts de représentation de la connaissance sont génériques. Cette connaissance est de deux types: une connaissance déclarative et une connaissance inférentielle. On présente ici les différents éléments qui permettent d'exprimer la connaissance d'interprétation sémantique.

   (a) Connaissance déclarative

   - Les **classes du domaine** sont utilisées pour décrire de manière explicite les différents objets physiques du domaine, les sous parties de ces objets ou des situations du domaine d'application. Nous définissons par situations des configurations spatiales de plusieurs objets du domaine ou sous parties d'objets du domaine qui ont une signification sémantique. Les classes du domaine sont définies par une liste de **propriétés** qui sont partagées par toutes les instances de la classe. Ces propriétés peuvent représenter des propriétés de composition (les sous parties), des propriétés d'apparence visuelle en terme de concepts visuels ou des propriétés de relations spatiales avec d'autres concepts.
   - Ces **classes du domaine** sont organisées de manière hiérarchique en **taxonomie du domaine**. Cette structure permet d'organiser la connaissance et elle reflète la hiérarchie de spécialisation du domaine d'application.
   - Le contexte propre au domaine (c'est à dire les informations non visuelles qui peuvent influencer l'interprétation) est représenté de manière explicite par une structure de connaissance appelée **contexte du domaine**
   - De la même manière le contexte d'acquisition d'images est représenté de manière explicite par la structure **contexte d'acquisition**.
   - Les **requêtes du domaine** expriment des requêtes de l'utilisateur. Elles expriment le problème initial d'interprétation.

   (b) Connaissance inférentielle
   Des **critères de contexte** sont utilisés pour décrire des prises de décisions pendant la résolution du problème. Ils représentent l'expertise sur comment prendre des décisions pour faciliter l'interprétation à partir du contexte. En particulier, les **critères d'initialisation de l'interprétation** représentent la connaissance sur comment initialiser l'interprétation à partir du contexte courant. Les **critères de post-interprétation** permettent de raffiner le résultat de l'interprétation à l'aide du contexte courant.

2. **Le moteur d'interprétation sémantique**
   L'idée principale est de réaliser l'interprétation sémantique de la même manière que

les experts du domaine, c'est à dire en utilisant leur propre terminologie et taxonomie. Le but principal est de donner un sens sémantique aux objets perçus dans la scène c'est à dire trouver les classes qui leur correspondent le mieux dans l'arbre représentant la taxonomie du domaine. Le raisonnement se base sur un parcours en profondeur d'abord de l'arbre des classes du domaine. Pour chaque noeud de l'arbre de classe, le moteur suit un cycle hypothèse/test:

- Le moteur d'interprétation construit des hypothèses d'objets visuels à l'aide de la description de la classe du domaine correspondant au noeud analysé. Ces hypothèses vont permettre de guider les traitements des niveaux inférieurs. Elles sont envoyés au module de gestion de données sous la forme d'une **requête de gestion de données**.
- Les instances des objets perçus crées par le module de gestion des données visuelles sont mises en correspondance avec la classe du domaine correspondant au noeud courant. Le résultat de cette mise en correspondance est la validation ou non de la classe comme interprétation possible des instances d'objets visuels.
- Une phase de raffinement a pour rôle de classer plus précisemment les instances d'objets visuels. Cette phase de raffinement consiste à analyser les classes filles de la classe courante si cette dernière est validée sinon à un retour arrière vers une classe précédemment sélectionnée.

8.2.4.2 Gestion des données visuelles

Le rôle principal du module de gestion des données visuelles est de faire la mise en correspondance entre les symboles servant à décrire de manière abstraite et qualitative les objets du monde réel et les données perceptuelles issues des capteurs. Le problème de gestion des données visuelles fait référence au problème **d'ancrage de symboles** de l'intelligence artificielle fortement étudié dans la domaine de la robotique [Coradeschi, 1999], [Coradeschi and Saffiotti, 2003], [Bredeche, 2002]. C'est un problème considéré comme primordial pour la création du sens dans un système intelligent artificiel. Ce problème a rarement été considéré comme un problème en tant que tel en interprétation sémantique d'images. Ainsi, les solutions proposées pour résoudre ce problème sont souvent fortement dépendantes du domaine d'application [Ossola, 1996]. Nous proposons donc de considérer ce problème comme un problème indépendant, possédant sa propre expertise et ses propres stratégies de raisonnement.

En particulier, dans le cadre de l'analyse de scènes ou de la reconnaissance d'objets composés complexes, la représentation des relations spatiales et la mise en oeuvre de raisonnements spatiaux est un sous problème important de la gestion de données visuelles.

1. **Modèle de représentation de la connaissance pour la gestion de données visuelles**
   La base de connaissances d'un système de gestion de données visuelles contient la description des différentes données à gérer, symboliques et perceptuelles, la représentation des *liens d'ancrage* entre ces données, la description des différentes relations spatiales ainsi qu'un ensemble de critères décisionnels pour guider le raisonnement. Dans la suite, nous décrivons les principaux concepts d'une base de connaissances de gestion de données visuelles : les **concepts visuels**, les **données images**, les **relations spatiales** et les **critères de gestion de données visuelles**.

   - Les **concepts visuels** représentent les données symboliques visuelles. Ce sont les différents symboles à ancrer. Leur description contient un **lien**

d'ancrage avec des descripteurs numériques de bas niveau. Comme dans [Coradeschi et al., 2001], ce lien d'ancrage est représenté par une liste de descripteurs numériques modélisés comme des variables linguistiques à l'aide du formalisme de la logique floue.

- Les **données images** représentent les données perceptuelles (issues des capteurs). Leur description contient une liste de descripteurs numériques qui servent à les caractériser.
- Un ensemble de **fonctionnalités de traitement d'images**.
- Les **relations spatiales** sont utilisées pour représenter de manière explicite les différentes relations spatiales à prendre en compte et leur propriétés.
- **Les critères de gestion de données** sont représentés par des règles de production. Ils jouent un rôle important pendant le raisonnement
  - Les **règles d'extraction d'objets** sont utilisées pour construire et pour contraindre une requête de traitement d'images en fonction de la description symbolique de l'objet visuel recherché.
  - Les **règles de déduction spatiales**, associées aux relations spatiales, sont utilisées pour l'inférence de relations spatiales à partir d'autres relations spatiales.
  - Les **règles d'évaluation** sont utilisées pour diagnostiquer les résultats du traitement d'images du point de vue de l'objet visuel recherché.

2. **Le moteur de gestion de données visuelles**
   Le moteur de gestion de données visuelles a pour rôle:

   - La construction d'instances de requêtes de traitement d'images en utilisant la description des objets visuels. Le moteur utilise les critères d'extraction d'objets pour contraindre ces requêtes.
   - La gestion des données visuelles et la description symbolique des objets perçus dans l'image. Cette gestion se fait en plusieurs phases. Une première phase est l'évaluation des résultats du traitement d'images (à l'aide des règles d'évaluation).
     - Si le résultat de l'évaluation est correct, le moteur sélectionne et décrit de manière symbolique les données perçues dans l'image. Pour cela, il utilise le lien d'ancrage des concepts visuels et une mise en correspondance *floue*. Les instances d'objets visuels ainsi crées sont ensuite envoyées au module d'interprétation pour leur interprétation sémantique.
     - Si le résultat de l'évaluation est incorrect, une nouvelle requête de traitement d'images est crée.
   - L'activation de raisonnements spatiaux dans le cas de la gestion de plusieurs objets visuels (activation des règles de déduction spatiale).

8.2.4.3 Le module de pilotage de programmes de traitement d'images

Le rôle du module de traitement d'images est l'extraction des objets d'intérêts de l'image et de leur description numérique. Pour un traitement intelligent des images, nous nous appuyons sur des techniques de pilotage de programmes existantes dans l'équipe Orion [Clement and Thonnat, 1993b] et qui ont été validées sur plusieurs applications [Thonnat et al., 1998b], en particulier pour le traitement d'images médicales

[Crubézy et al., 1997] et le traitement d'images aériennes [Mathieu-Marni et al., 1995]. De plus, le pilotage de programmes est une technique qui favorise la réutilisabilité [Moisan and Thonnat, 2000]. La conception de ce module ne fait pas partie des contributions de ces travaux de thèse. Nous avons réutilisé le moteur PEGASE et le modèle de représentation de connaissances associé. Notre travail a consisté à intégrer ces solutions existantes dans la plate forme de vision cognitive. Une présentation claire des solutions proposées peut être trouvée dans [Thonnat, 2002].

8.2.5 Principe d'utilisation de la plate forme de vision cognitive

Le principe d'utilisation de la plate forme est représenté sur la figure 8.3. Il s'agit d'un travail de coopération entre différents experts qui n'agissent qu'à leur propre niveau d'expertise.

- L'expert du domaine utilise l'ontologie de concepts visuels pour décrire les concepts de son domaine (**classes du domaine**), décrit quelles sont les informations contextuelles à prendre en compte (**contexte du domaine et contexte d'acquisition**) et écrit des règles de décision portant sur ces informations contextuelles (critères de contexte).

- L'expert en gestion de données visuelles construit la base de connaissance du module à l'aide de l'ontologie de concepts visuels et de l'ontologie de concepts de traitement d'images. Les concepts visuels sont décrits de manière générique en construisant le lien d'ancrage non contraint avec des descripteurs images et des règles d'extraction d'objets associés, de même que les relations spatiales et les critères de déduction spatiale. Les concepts d'entités image fournis par l'ontologie de concepts de traitement d'images sont décrit par une liste de descripteurs image qui servent à les caractériser. Cet ensemble de connaissances peut être réutilisé et complété de manière incrémentale pour un ensemble d'applications respectant les hypothèses fortes décrites dans la section 8.2.1. La partie de la base de connaissances de gestion visuelle qui est propre à l'application particulière est la construction des instances de concepts visuels (et des contraintes sur leur lien d'ancrage) et des relations spatiales utilisées pour décrire les concepts du domaine.

- L'expert en traitement d'images choisit une bibliothèque générique de programmes de traitement d'images et écrit une base de connaissances sur comment utiliser ces programmes en se basant sur l'ontologie de concepts de traitement d'images.

8.3 Application à la reconnaissance des pathologies végétale

Cette partie des travaux de thèse est l'objet d'une collaboration entre l'équipe Orion de l'INRIA et l'INRA (Institut national de Recherche en Agronomie) de Sophia Antipolis. L'objectif de la collaboration entre ces deux organismes, supportée par la région PACA (Provence Alpes Côte d'Azur) et le SCRADH (Syndicat du Centre Régional d'Application et de Démonstration Horticole), était de faire des avancées dans le domaine de la détection précoce de pathologies des cultures sous serre en utilisant les méthodes de la vision et de l'interprétation automatique d'images.

Un travail de recherche bibliographique a permis de montrer que la vision par ordinateur est une technique en vogue pour l'automatisation du contrôle de l'état sanitaire et de la production de plantes maraîchères ou ornementales. Les applications sont nombreuses

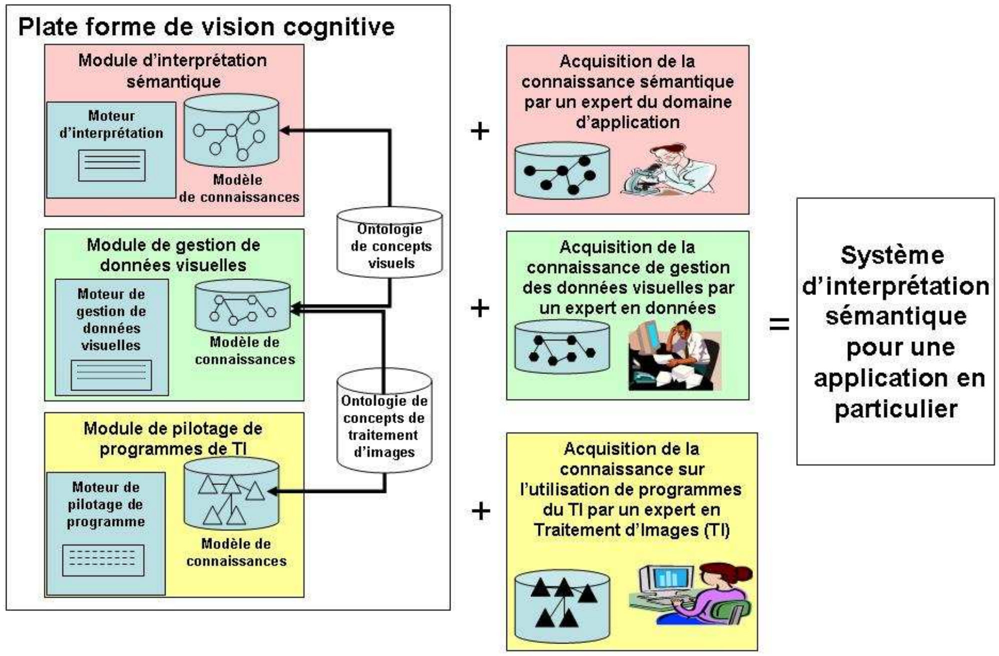


Figure 8.3: Principe d'utilisation de la plate forme de vision cognitive pour une application particulière

et variées: inspection de la qualité des fruits, contrôle de la croissance de la plante, contrôle du stress de la plante, diagnostic automatique de pathologies. Cependant, tous les systèmes proposés sont spécifiques et dépendant d'une application en particulier.

La plate forme de vision cognitive a été utilisée pour construire un système d'interprétation d'images microscopiques de feuilles de rosier pour le diagnostic des maladies du rosier de serre. A l'aide de la plate forme de vision cognitive, il s'agissait de:

- La construction d'une base de connaissances dédiée au domaine d'application: les pathologies foliaires du rosier de serre.
- La construction d'une base de connaissances de gestion des données visuelles, c'est à dire la construction du lien d'ancrage entre les concepts visuels utilisés pour décrire le domaine et des descripteurs image de l'ontologie de concepts de traitement d'images et la définition de critères de gestion de données visuelles.
- La construction d'une base de connaissances de pilotage de programmes de traitement d'images. Nous avons utilisé la bibliothèque de traitements d'images PANDORE[2] à laquelle nous avons rajouté quelques fonctionnalités en particulier la détection de structures curvilignes dans l'image [Steger, 1998].

Le travail d'acquisition de la connaissance du domaine d'application a été effectué par interviews et à l'aide d'outils spécifiques, Annotate et Ontovis, avec des experts pathologistes de l'INRA d'Avignon. Une base de connaissances décrivant les signes et symptômes

[2] http://www.greyc.ensicaen.fr/ regis/Pandore/

précoces des pathologies du rosier de serre à l'aide de l' ontologie de concepts visuels a été développée. Pour le diagnostic, c'est l'association symptômes/organes qui est intéressante et la base de connaissance est donc propre à chaque organe.

Le système ROSESIM a été testé sur une cinquantaines d'images représentant soit des maladies fongiques au stade précoce soit des ravageurs. Ce travail de test ne constitue absolument pas une validation du système ROSESIM qui reste à faire mais elle a permis de tester le fonctionnement des différents moteurs de la plate forme et de tester leur coopération. L'INRA souhaite utiliser cette plate forme pour compléter les bases de connaissances développées et les valider.

8.4 Conclusions et perspectives

Malgré une recherche fructueuse et dynamique en vision par ordinateur et en intelligence artificielle, l'interprétation sémantique d'images est toujours un problème majeur. Dans cette thèse, nous abordons ce problème sous l'angle de la construction de systèmes automatiques d'interprétation sémantique d'images. Ce problème couvre un vaste champ d'applications: vidéo-surveillance, diagnostic, imagerie médicale, imagerie aérienne, inspection industrielle,...En particulier, il est souvent reproché aux systèmes d'interprétation d'images existants leur dépendance vis à vis de l'application, leur manque de robustesse et de flexibilité. Leur conception est souvent longue et coûteuse.

Depuis quelques années, une nouvelle discipline de recherche, la vision cognitive, tente de répondre à cette problématique.

8.4.1 Contributions de la plate forme de vision cognitive

8.4.1.1 Une plate forme réutilisable

La plate forme de vision cognitive proposée est un ensemble d'outils réutilisables pour la conception de systèmes d'interprétation d'images.

1. **Réutilisabilité à l'aide de techniques de résolution de problèmes**
   L'utilisation des techniques de systèmes à base de connaissances et de générateurs de systèmes à base de connaissances permet la construction de moteurs génériques dédiés à une classe de problèmes mais indépendants d'une expertise particulière. Ces techniques permettent la séparation entre la connaissance particulière, le modèle de représentation de cette connaissance et les stratégies de raisonnement utilisant cette connaissance. Cette séparation permet non seulement la réutilisation des stratégies de raisonnement mais aussi du modèle de représentation de la connaissance.

2. **Réutilisabilité à l'aide du génie ontologique**
   Le génie ontologique est un moyen pour le partage et pour la réutilisation des connaissances. La plate forme de vision cognitive est composée de deux ontologies:

   - **Une ontologie de concepts visuels** indépendante de tout domaine d'application. Cette ontologie permet de faciliter l'acquisition de la connaissance sémantique en fournissant aux experts des domaines d'application un ensemble de termes visuels génériques pour décrire les concepts de leur domaine. En tant que corpus commun entre le module de gestion de données visuelles et le module d'interprétation sémantique, cette ontologie permet de réduire le fossé sémantique entre les concepts de haut niveau et les données images et elle permet la communication entre ces deux modules.

- **Une ontologie de concepts de traitement d'images** permettant la communication entre le module de pilotage de programmes et le module de gestion des données visuelles. Cette ontologie contient un ensemble de concepts structurés en entités, descripteurs et fonctionnalités permettant de décrire les problèmes de traitement d'images et leurs résultats. Cette ontologie permet de guider et donc de faciliter la construction de la base de connaissances de pilotage de programmes et celle de gestion de données visuelles.

Concernant la plate forme de vision cognitive, les contributions n'ont pas été les même pour tous les modules de la plate forme.

1. **Module d'interprétation sémantique**
Pour construire le module d'interprétation sémantique nous nous sommes inspirés de travaux développés auparavant dans l'équipe Orion concernant la reconnaissance d'objets isolés. Nous avons modifiés les solutions existantes pour permettre en compte la reconnaissance de situations (prise en compte des relations spatiales entre les objets et de leur configuration spatiale). Nos travaux ont aussi contribué à l'intégration de l'ontologie de concepts visuels dans le nouveau modèle de représentation des connaissances proposé. Dans [Thonnat, 2002], le moteur proposé est un moteur de classification. Nous avons modifié ce moteur pour prendre en compte des requêtes de l'utilisateur et pour la construction de requêtes pour le module de gestion de données visuelles quand elles sont nécessaires.

2. **Module de gestion de données**
Ce module constitue le coeur de nos contributions. Ce module permet de traiter les problèmes d'ancrage de symboles et de raisonnement spatial comme des problèmes à part entière avec leur propre expertise et leur propres stratégies de raisonnements. Nous avons défini dans le cadre de cette thèse un ensemble de concepts génériques pour modéliser la connaissance liée à la gestion des données visuelles et un moteur générique permettant de mettre en oeuvre des stratégies dirigées par les modèles et dirigées part les données de gestion de données visuelles.

3. **Module de pilotage de programme de traitement d'images**
Pour ce module, notre contribution s'est limitée à l'intégration de solutions existantes, conçues dans l'équipe Orion [Clement and Thonnat, 1993b] et validées sur des applications diverses [Thonnat et al., 1998b].

8.4.1.2 Facilité d'utilisation de la plate forme de vision cognitive

La plate forme de vision cognitive permet de limiter la construction d'un système d'interprétation d'images pour une application particulière à la construction de trois bases de connaissances particulières. La construction de ces bases de connaissances est un travail de coopération entre différents experts qui dès lors n'interviennent qu'au niveau d'abstraction correspondant à leur expertise. Des langages de description dédiés (YAKL pour le pilotage de programme et SIKL++ pour l'interprétation sémantique) permettent la construction de ces bases de connaissances d'une manière naturelle.

8.4.2 Le système ROSESIM

La plate forme de vision cognitive a été utilisé pour construire un système d'interprétation sémantique pour une application concrète: le diagnostic précoce des pathologies foliaires du rosier de serre. La création de ce système a permis de valider les modèles de connaissances proposés, le fonctionnement des moteurs ainsi que leur coopération. Du point de vue applicatif, des développements à court terme sont nécessaires pour améliorer le système ROSESIM. Cependant, l'INRA souhaite utiliser ce système et la plate forme de vision cognitive pour compléter les bases de connaissances développées et pour les valider.

8.4.3 Vers un système minimal d'interprétation sémantique

Nous proposons d'ajouter une dimension supplémentaire de réutilisabilité aux solutions proposées. Il s'agit de fournir en plus de la plate forme de vision cognitive, une base de connaissances minimale de gestion de données visuelles et une base de connaissances minimale de pilotage de programmes de traitement d'images. La construction d'un système d'interprétation sémantique pour une application particulière se résume alors à la construction de la base de connaissances d'interprétation sémantique propre à l'application et à la construction des instances de concepts visuels et de relations spatiales utilisés pour décrire les concepts de l'application pour compléter la base de connaissances de gestion de données. Dans le cas ou cela est nécessaire les bases de connaissances minimales de pilotage de programmes de traitement d'images et de gestion de données visuelles peuvent être complétées de manière incrémentale pour répondre aux besoins spécifiques de l'application (ajout de nouveaux concepts visuels, de nouveaux types de relations spatiales, de nouvelles fonctionnalités de traitement d'images). Les applications cibles doivent bien sûr appartenir à une même classe de problème (scènes statiques 2D dans notre cas).

8.4.4 Perspectives à court terme

8.4.4.1 Apprentissage de la relation d'ancrage entre les concepts visuels et les descripteurs images

Une des faiblesses majeures de la plate forme de vision cognitive proposée est la construction manuelle du lien d'ancrage entre les concepts visuels utilisées pour décrire le domaine et des descripteurs image de l'ontologie de concepts de traitement d'images. Nous proposons d'utiliser les travaux de Nicolas Maillot développés dans l'équipe ORION sur l'apprentissage à l'aide d'exemples représentatifs de détecteurs de concepts visuels [Maillot et al., 2004a]. Nous proposons d'intégrer le module d'apprentissage proposé dans la plate forme pour faciliter la construction de la base de connaissances de gestion des données visuelles. Le processus d'acquisition de la connaissance serait donc composé de deux étapes:

- L'acquisition de la connaissance du domaine d'application à l'aide de l'ontologie de concepts visuels consistant en la description des différentes classes du domaine par des concepts visuels mais aussi en l'annotation et en la segmentation manuelle d'un ensemble d'images d'exemples.
- La création automatique des liens d'ancrage entre les instances des concepts visuels utilisées pour décrire les classes du domaine et les descripteurs image selon le procédé décrit dans l'image 8.4

8.4.4.2 Apprentissage pour la segmentation d'images

Dans [Martin, 2004], un apprentissage supervisé pour améliorer la segmentation d'images est proposé. En particulier, les techniques d'apprentissage proposées permettent le paramétrage automatique des algorithmes de segmentation. Afin d'améliorer la robustesse et

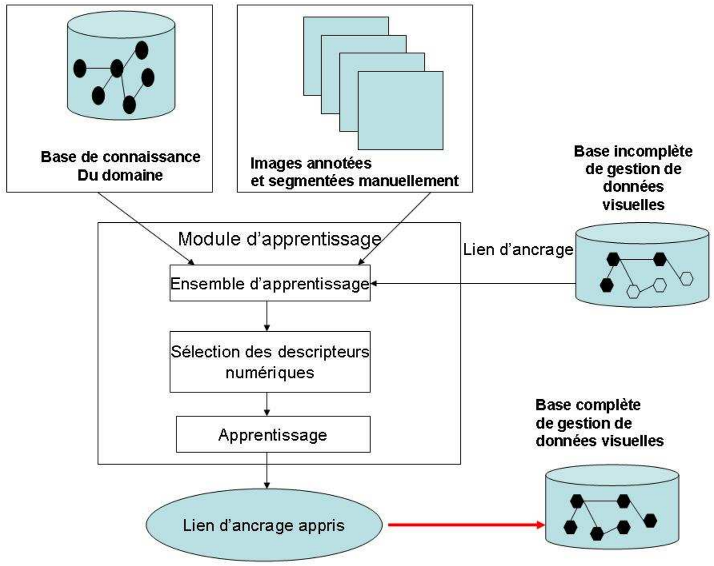


Figure 8.4: Apprentissage basé sur l'ontologie de concepts visuels pour la construction automatique des liens d'ancrage

l'autonomie de la plate forme de vision cognitive, nous envisageons d'intégrer ces techniques dans la plate forme. Elle permettrait de réduire le coût de construction de la base de connaissances dédiée au pilotage de programmes de traitement d'images.

8.4.5 Perspectives à long terme

8.4.5.1 Des bases de connaissances dynamiques

Actuellement, les solutions proposées par la plate forme se heurtent encore au problème du *monde fermé*, c'est à dire qu'elles se limitent à la connaissance a priori modélisée dans les bases de connaissances. Une solution pour résoudre ce problème pourrait être la construction de bases de connaissances dynamiques capables de s'actualiser automatiquement et d'être augmentées de manière incrémentale.

8.4.5.2 Intégration de la dimension temporelle

Une évolution à long terme possible de la plate forme de vision cognitive proposée est l'introduction de la dimension temporelle. Le but serait d'étendre les solutions proposées à l'interprétation sémantique de séquences d'images c'est à dire en terme d'événements ou de scénarios.

8.4.5.3 Un raisonnement opportuniste

La plate forme de vision cognitive pourrait être modifier par l'ajout de stratégies de recherches opportunistes, c'est à dire pas seulement par des requêtes mais en prenant en compte des situations ou des événements comme dans [Sandakly, 1995]. Cela permettrait de rendre la plate forme plus flexible.

Appendix A

The SIKL++ Grammar

The SIKL++ language has been developped from the SIKL language to enable domain experts to describe all the different types of knowledge involved in semantic interpretation, independently of any application domains. Moreover, the current SIKL++ language is also used to built the visual data management knowledge base. As SIKL and YAKL, the language offers two types of declarative descriptions: structured frame-based and rule-oriented. This appendix describes in detail the BNF(Bacchus Naur Form) grammar rule of SIKL++.

A.1 Semantic Interpretation Knowledge Base

A.1.1 Domain Knowledge Base

A semantic interpretation knowledge base in SIKL++ is composed of :

- either a description of the knowledge base as whole (**domainkb_desc**) in a separate file (with **.kb** extension),
- or a **.sikl++** file composed of:
  - a set of imported file names (**import_list**) (in particular the file describing visual concepts),
  - definitions of domain classes (**domainclass_def**),
  - definitions of domain and acquisition context types and instances (**context_def** and **context_instance_def**),
  - definitions of context criteria (**context_criteria_def**),
  - definitions of domain requests and domain request instances (**domainrequest_def**,**domainrequest_instance_def**),
  - definition of domain taxonomy (**domaintaxo_def**).

domainkb: domainkb-desc
 | import_list domainclass_def context_def context_instance_def
 context_criteria_def domainrequest_def domaintaxo_def

domainkb_desc: **DomainKB { name** IDENT
 Complete Name strings

path
Version FLOAT
Domain Taxonomy ident_list
List of Files ident_list **}**

path:
|**KB Path** strings

A.1.2 Imported files

They represent other KB files, needed for the description of the current files contents. In particular, the definition of visual concepts is needed for the description of domain classes.

import_list:
| import_list import_line
import_line: **Import** ident_list

A.1.3 Definition of Domain Classes

domainclass_def:
| domainclass_list
domainclass_list: domainclass
| domainclass_list domainclass
domainclass : **DomainClass { name** IDENT (FLOAT) comments superclass
subpart_description visualdescription spatial_relation_description **}**
| **DomainClass { name** IDENT (FLOAT) comments
subpart_description visualdescription spatial_relation_description **}**
comments
| **comment** STRING
superclass: **SuperClass** IDENT
subpart_description:
| **SubPart Description** property-list
visualdescription:
| **Visual Description** spatialdescription colordescription texturedescription
spatialdescription:
| **Spatial Description** property-list
colordescription:
| **Color Description** property-list
texturedescription:
| **Texture Description** property-list
spatial_relation_description:
| **Spatial Relation Description** property-list

Property Declaration

property-list: property
| property-list property
property: type **name** IDENT (FLOAT) comments property-info

type: IDENT
property-info: default range facets
default:
|**default** value
value: IDENT
|FLOAT
|INTEGER
|STRING
| |value-set|
nil
|**{**CODE**}**
range:
| **range** [interval]
| **range** [value-set]
| **range** [(IDENT, IDENT)] (*dans le cas particulier des structures spatiales*)
interval: FLOAT,FLOAT
| INTEGER,INTEGER
value-set: ident-list
| float-list
| int-list
| string-list
ident-list: IDENT
| ident-list IDENT
float-list: FLOAT
| float-list FLOAT
int-list: INTEGER
| int-list INTEGER
string-list: STRING
|string-list STRING
facets:
| facet-list
facet-list: facet
|facet-list facet
facet: **At least** INTEGER
| **At most** INTEGER
| **calculation** CODE
| **calculation** IDENT

A.1.4 Definition of Context

Experts can define two types of contexts: domain context and acquisition context.

context_def:
| context_list

context_list : context
| context_list context

context: domaincontext

| acquisitioncontext

domaincontext: **DomainContext {** **name** IDENT attributes }
acquisitioncontext: **AcquisitionContext {** **name** IDENT attributes }
attributes:
|**Attributes** attribute-list
atribute-list: attribute
| attribute-list attribute
attribute: IDENT **name** IDENT comments attribute-info
attribute-info: default range if-needed
if-needed:
|**calculation** CODE
|**calculation** IDENT
|**calculation** item-file

Once the types of context defined, experts or end-users can define instances of these contexts. These instances are used in domain request attribute assignment.

context_instance_def:
|context_instance_list

context_instance_list: context_instance
| context_instance_list context_instance

context_instance: domaincontext-instance
| acquisitioncontext-instance

domaincontext-instance: **DomainContextInstance{** IDENT **name**
IDENT comments
attribute-assignments **}**

acquisitioncontext-instance:**AcquisitionContextInstance{** IDENT **name**
IDENT comments
attribute-assignments **}**
attribute-assignments:
| **Attributes** attribute-assign-list
attribute-assign-list: attribute-assig
|attribute-assig-list attribute-assig
attribute-assig :IDENT := value
|IDENT :=**{** attribute-assig-list **}**

A.1.5 Definition of Domain Requests

Domain requests are defined by domain experts and their instances come from end users.

domainrequest_def:
|domainrequest_list
domainrequest_list: domainrequest

|domainrequest_list domainrequest
domainrequest: **DomainRequest** {
name : IDENT comments
Attributes attribute-list **}**

domainrequest_instance_def:
|domainrequest_instance_list
domainrequest_instance_list: domainrequest_instance
|domainrequest_instance_list domainrequest_instance
domainrequest_instance: **DomainRequest** {
name : IDENT comments
Attributes attribute_assignements **}**

A.1.6 Definition of Context Criteria

contextcriteria_def:
|contextcriteria_list

contextcriteria_list: contextcriteria
| contextcriteria_list contextcriteria

contextcriteria: domainrule

domainrule:**Rule{**rulebody **}**
rulebody:**name** IDENT comments owner
Let declslist
If preclist **Then** actlist
| **name** IDENT comments owner
If preclist **Then** actlist
owner:
| **LinkedDomainClass** IDENT
declslist: decl
|declslist, decl
decl: IDENT a IDENT
| IDENT an IDENT
| IDENT in IDENT
| IDENT COMPOSEIDENT

Premise

precslist:true
|prec
|precslist, prec
prec: (prec)
|**not** prec
|**{** CODE **}**
|rule-exp <> **nil**
|rule-exp == **nil**

|rule-exp compar rule-exp
|valid IDENT
rule-exp:(rule-exp)
|value
|**max**(rule-exp.rule-exp)
|**min**(rule-exp.rule-exp)
|rule-exp + rule-exp
|rule-exp - rule-exp
|rule-exp * rule-exp
|rule-exp / rule-exp
|rule-exp **quotient** rule-exp
|COMPOSIDENT

Actions

actlist: act
|actlist, act
act: common_actions
|postinterpretation_actions
|initialization_actions
common_actions: **CODE**
| IDENT:=exp
| COMPOSIDENT
| COMPOSIDENT:=exp
postinterpretation_actions: **ALERT** STRING
initialization_actions: **setImportance** IDENT FLOAT

A.1.7 Domain Taxonomy

domaintaxonomy:**DomainTaxonomy { name** IDENT
Root IDENT
Context_Criteria_List rule-list

A.1.8 Visual Concept

visualconcept : **VisualConcept { name** IDENT comments superconcept
grounding extraction **}**
| **VisualConcept { name** IDENT comments
grounding extraction **}**
superconcept: **SuperConcept** IDENT
grounding:
|**Grounding Link** fuzzy-descriptor-list
extraction:
|**Object Extraction Criteria** rule-list

fuzzy-descriptor-list: fuzzy-descriptor
|fuzzy-descriptor-list fuzzy-descriptor
fuzzy-descriptor: type **name** IDENT comments fuzzydescriptor-info

type: IDENT
fuzzydescriptor-info: linguistic-values domain fuzzy-sets unit
linguistic-values:
|**LinguisticValues** [lvalue-set]
lvalue-set: string-list
fuzzy-sets:
|**FuzzySet** fuzzyset-list
fuzzyset-list: fuzzyset
| fuzzyset-list fuzzyset
fuzzyset:{INTEGER, INTEGER, INTEGER, INTEGER}
|{FLOAT, FLOAT, FLOAT, FLOAT}
domain:**Domain**[interval]

Appendix B

The ROSESIM System

B.1 The ROSESIM Interpretation Knowledge Base

B.1.1 Domain KB definition

A file name domainkb-rosesym.kb contains the domain knowledge base description alone.

B.1.2 Domain Classes

A domainclasses.siklpp file gathers the descriptions of the ROSESIM application domain classes. It imports the visualconcepts.syklpp file, because visual concepts are necessary to describe domain classes. Visual concept instances can be defined by the domain expert knowledge to constrained image descriptors values. An example is given for the domain class HYPHAE. Some domain classes are empty. Some of them represents intermediate domain classes. Others are empty but included in the domain knowledge base because they have a bological meaning but we have not study them.

Import visualconcepts

DomainClass {
    name LEAF
    comment "A rose leaf scene"
    SubPart Description
        VEGETAL_SUPPORT **name** leaf_vegetal_support
        range [VERY_YOUNG_VS YOUNG_VS MATURE_VS]

DomainClass {
    name VEGETAL_SUPPORT
    comment "The description of the vegetal structure of the rose leaf"
    SubPart Description
        VEGETAL_TISSUE **name** leaf_vegetal_tissue
        VEINS **name** leaf_veins
    Visual Description
        Spatial Description
            Scene_Spatial_Occupation **name** VS_occupation
            range [*Partial Complete*] **}**

DomainClass {
    name VEGETAL_TISSUE

comment "The description of the apparence of the vegetal tissue of a rose leaf"
Visual Description
Spatial Description
Geometry **name** vtgeometry
range [*Point_Set*]
Color Description
Generic_Hue **name** vthue
range [*Red Green*]
Lightness **name** vtlightness
range [*Very_Dark Dark*]
}

DomainClass {
name VERY_YOUNG_VEGETAL_SUPPORT
comment "The leaf is a very young leaf"
SuperClass VEGETAL_SUPPORT
Visual Description
leaf_vegetal_tissue.vthue
range [Red]
leaf_veins.hue
range [Red] **}**

DomainClass {
name YOUNG_VEGETAL_SUPPORT
comment "The leaf is young"
SuperClass VEGETAL_SUPPORT
Visual Description
leaf_vegetal_tissue.vthue
range [Green]
leaf_veins.hue
range [Red] **}**

DomainClass {
name MATURE_VEGETAL_SUPPORT
comment "The leaf is mature"
SuperClass VEGETAL_SUPPORT
Visual Description
leaf_vegetal_tissue.vthue
range [Green]
leaf_veins.veins_hue
range [Green] **}**

DomainClass {
name VEINS
comment "Network of leaf veins"
Visual Description
Spatial Description
Network_Of **name** veins_network
range [(VEIN,CONNECTED)]
Color Description
Generic_Hue **name** veins_hue
range [*Red Green*]
}

DomainClass {
name VEIN

comment "A leaf vein"
Visual Description
Spatial Description
Geometry **name** vein_geometry
range [*Line Segment*]
Thickness **name** vein_thickness
range [*Thick*]
Color Description
Generic_Hue **name** vein_hue
range [*Red Green*]
}

DomainClass {
name HEALTHY_LEAF
comment "A rose leaf without disease"
SuperClass LEAF
Visual Description
Spatial Description
LEAF.VEGETAL_SUPPORT.VS_occupation
range [Complete] **}**

DomainClass {
name NON_HEALTHY_LEAF
comment "A rose leaf with some diseases"
SuperClass LEAF
Spatial Description
LEAF.VEGETAL_SUPPORT.VS_occupation
range [Partial] **}**

DomainClass {
name FUNGAL_INFECTION
comment "An infection by a fungal pathogen : presence of fungi symptoms"
SuperClass NON_HEALTHY_LEAF
SubPart Description
FUNGI_SYMPTOM **name** fungal_infection_symptom
range [MYCELIUM CONIDIA CONIDIOPHORES]
at least 1
}

DomainClass {
name FUNGI_SYMPTOM
comment "A sign or a symptom of the presence of fungi"**}**

DomainClass {
name MYCELIUM
comment "A group or mass of discrete hyphae : vegetative structure of fungi"
SuperClass FUNGI_SYMPTOM
Visual Description
SpatialDescription
Network_Of **name** mycelium_network
range [(HYPHAE, *Connected*)]
at least 1
Density **name** mycelium_density
range [*Partially_Spaced Spaced*]
Relational Description
ProperPart **name** mycelium_proper_part_of

range [LEAF] **}**

DomainClass {
name FREELY_DISPERSED_MYCELIUM
comment "Mycelium with low degree of branching"
SuperClass MYCELIUM
Visual Description
SpatialDescription
Mycelium.mycelium_density
range [*Spaced*] **}**

DomainClass {
name MYCELIAL_CLUMP
comment "Mycelium with high degree of branching"
SuperClass MYCELIUM
Visual Description
Spatial Description
Mycelium.mycelium_density
range [*Partially Spaced*]
Relational Description
Connected **name** mycelium_connected_to
range [CONIDIOPHORES]
al least 1 **}**

DomainClass {
name HYPHAE
comment "A threadlike, tubular filamentous fungal structure"
SuperClass FUNGI_SYMPTOM
Visual Description
Spatial Description
Geometry **name** hyphae_geometry
range [*Line Segment*]
Thickness **name** hyphae_thickness
range [*Thin Very_Thin*]
Length **name** hyphae_length
Straightness **name** hyphae_straightness
range [*Almost_Straight*]
Color Description
Neutral_Color **name** hyphae_color
range [*White Gray*]
Lightess **name** hyphae_lightness
range [*Very_Light Light*] **}**
//

VisualConceptInstance {
Thickness **name** HYPHAE_THICKNESS
Grounding Link
Float **name** width
domain [1 10]
unit [um] **}**

VisualConceptInstance {
Length **name** HYPHAE_LENGTH
Grounding Link
Float **name** lenght
domain [10 infinity]

unit [um] **}**
//

DomainClass {
name CONIDIA
comment “”
SuperClass FUNGI_SYMPTOM
Visual Description
Spatial Description
Geometry **name** conidia_geometry
range [*Elliptical_Surface*] **}**

DomainClass {
name CONIDIOPHORES
comment “”
SuperClass FUNGI_SYMPTOM **}**

DomainClass {
name GERMINATED_CONIDIA
comment “Germinating Conidia : the symptom appears as a germ tube”
SuperClass CONIDIA
Relational Description
Connected_to **name** conidia_connected_to
range [HYPHAE] **}**

DomainClass {
name CONIDIA_CLUSTER
comment “Cluster of Conidia ”
SuperClass CONIDIA
Relational Description
Connected_to **name** conidia_connected_to
range [HYPHAE] **}**

DomainClass {
name UNGERMINATED_CONIDIA_PRESENCE
comment “Presence of a fungi conidia ”
SuperClass FUNGAL_INFECTION
SubPart Description
FUNGAL_INFECTION.fungal_infection_symptom
range [CONIDIA CONIDIA_CLUSTER] **}**

DomainClass {
name HARMLESS_POWDERY_MILDEW
comment “” **}**

DomainClass {
name SPREADING_POWDERY_MILDEW
comment “Powdery Mildew”
SuperClass MYCELIUM_INSTALLATION **}**

DomainClass {
name PEST_INFECTION
comment “Presence of pests”
SuperClass NON_HEALTHY_LEAF
SubPart Description
PEST_SYMPTOM **name** pestinfection_symptom

at_least 1 **}**

DomainClass {
name PEST_SYMPTOM
comment "Symptom of the presence of pest"**}**

DomainClass {
name INSECT_BODY
comment "Presence of an insect,Description of its global body"
SuperClass PEST_SYMPTOM
Visual Description
Spatial Description
Geometry **name** body_geometry
range [*Geometric_surface Non_Geometric_Surface*]
Size **name** body_size
range [*Small Medium Important*]
Elongation **name** body_elongation
range [*Moderately_Elongated Moderately_Squat*]
Topology **name** body_topology
range [*Without_holes*]
Symmetry **name** WFabdomen_symmetry
range [*PrincipalAxisR_symmetrical*]
Color Description
Color_Homogeneity **name** body_color_homegeneity
range [*Homogeneous Moderately_Homegeneous*] **}**

DomainClass {
name WHITEFLY_BODY
comment "Presence of an insect,Description of its global body"
SuperClass INSECT_BODY
Visual Description
INSECTBODY.body_size
range [*Medium Important*]
INSECTBODY.body_elongation
range [*Moderately_Elongated*]
Periphery **name** whitefly_periphery
range [*Smooth Rough Hairy Highly_Hairy*]
Color Description
Hue **name** whitefly_body_hue
range [*White Gray Yellow*] **}**

DomainClass {
name APHID_BODY
comment "Presence of an insect,Description of its global body"
SuperClass INSECT_BODY
Visual Description
INSECTBODY.body_size
range [*Medium Important*]
INSECTBODY.body_elongation
range [*Moderately_Elongated Moderately_Squat*]
Periphery **name** aphid_periphery
range [*Smooth*]
Color Description
Hue **name** aphid_body_hue
range [*Green*]
Lightness **name** aphid_body_lightness

range [*Vey_Light*] **}**

DomainClass {
name SPIDERMITE_BODY
comment "Presence of an insect,Description of its global body"
SuperClass INSECT_BODY
Visual Description
Spatial Description
INSECTBODY.body_size
range [*Small*]
INSECTBODY.body_elongation
range [*Moderately_Elongated Moderately_Squat*]
Periphery **name** aphid_periphery
range [*Hairy*]
Color Description
INSECT_BODY.body_color_homegeneity
range [*Moderately_Homegeneous*]
Hue **name** spidermite_body_hue
range [*Brown Brownish_Orange Reddish_Orange Green Greenish_Yellow*]
Lightness **name** aphid_body_lightness
range [*Vey_Light*] **}**

DomainClass {
name INSECT_EGG
comment "Presence of egg insect, Description of egg appearance"
SuperClass PEST_SYMPTOM
Visual Description
Spatial Description
Geometry **name** egg_geometry
range [*Elliptic_surface*]
Size **name** egg_size
range [*Very_Small*]
}

DomainClass {
name WHITEFLY_EGG
SuperClass INSECT_EGG
Visual Description
Spatial Description
INSECT_EGG.egg_geometry
range [*Elliptical_Oblong_surface*]
Lenght **name** WFegg_size
Color Description
Hue **name** WFegg_hue
range [*White Yellow Gray*] **}**

DomainClass {
name SPIDERMITE_EGG
SuperClass INSECT_EGG
Visual Description
Spatial Description
INSECT_EGG.egg_geometry
range [*Circular_surface*]
Lenght **name** WFegg_size
Color Description
Translucidity **name** SPegg_translucidity

range [*Translucent*] **}**

DomainClass {
name WHITE_FLY_INFECTION
comment "Presence of White Fly on the leaf"
SuperClass PEST_INFECTION
SubPart Description
PEST_INFECTION.pestinfection_symptom
range [WHITEFLY_BODY] **}**

DomainClass {
name WHITE_FLY_ADULT
comment 'Presence of at least 1 white fly adult on the rose leaf"
SuperClass WHITE_FLY_INFECTION
SubPart Description
WHITE_FLY_ABDOMEN **name** abdomen
WHITE_FLY_ANTENNA **name** antenna1
WHITE_FLY_ANTENNA **name** antenna2 **}**

DomainClass {
name WHITE_FLY_PUPAE
comment 'Presence of at least 1 white fly adult on the rose leaf"
SuperClass WHITE_FLY_INFECTION
Visual Description
Spatial Description
Geometry **name** WFpupae_shape
range [*Elliptical_Surface*]
Lenght **name** WFpupae_lenght
Periphery **name** WFpupae_periphery
range [*Highly_Hairy*]
Color Description
Hue **name** pupae_hue
range [*Green*]
Lightness **name** pupae_lightness
range [*Very_Light*] **}**

DomainClass {
name WHITE_FLY_REPRODUCTION
comment "Presence of White Fly egg on the leaf"
SuperClass PEST_INFECTION
SubPart Description
PEST_INFECTION.pestinfection_symptom
range [WHITEFLY_EGG] **}**

DomainClass {
name WHITE_FLY_EGG_STRING
comment "Presence of White Fly egg string on the leaf"
SuperClass WHITE_FLY_REPRODUCTION
Visual Description
Spatial Description
Circle_Of **name** egg_circle
range [(WHITEFLY_EGG,NEAR_OF)] **}**

DomainClass {
name WHITE_FLY_ABDOMEN
comment "General description of a white fly abdomen"

SuperClass DomainClass
Visual Description
Spatial Description
Geometry **name** WFabdomen_shape
range [*Heart_Like_Shape*]
Convexity **name** WFabdomen_convexity
range [*Convex*]
Elongation **name** WFabdomen_elongation
range [*Strongly_Elongated Moderately_Elongated*]
Length **name** WFabdomen_lenght
range [*Long*]
Color Description
Neutral_Color **name** abdomen_color
range [*White Gray*]
Lightness **name** abdomen_lightness
range [*Light Very_Light*]
Relational Description
PP **name** abdomen_proper_part_of
range [White_Fly_Body] **}**

DomainClass {
name White_Fly_Antenna
comment "General description of a white fly antenna"
SuperClass DomainClass
Visual Description
Spatial Description
Geometry **name** WFantenna_shape
range [*Line Segment*]
Length **name** WFantenna_lenght
range [*Important*]
Relational Description
PP **name** antenna_proper_part_of
range [White_Fly_Body]
InFrontOf **name** antenna_infront_of
range [White_Fly_Abdomen] **}**

DomainClass {
name White_Fly_Adult_In_Insemination
comment 'Adult in phase of insemination"
SuperClass WWhite_Fly_Adult
Relational Description
Near_Of **name** WF_near_of
range [White_Fly_Egg White_Fly_Egg_String] **}**

DomainClass {
name Spider_Mite_Infection
comment "Presence of Aphid on the leaf"
SuperClass Pest_Infection
SubPart Description
Pest_Infection.pestinfection_symptom
range [Spider_Mite_Body] **}**

DomainClass {
name Spider_Mite_Reproduction
comment "Presence of Aphid on the leaf"
SuperClass Pest_Infection

SubPart Description
PEST_INFECTION.pestinfection_symptom
range [SPIDER_MITE_EGG] **}**

DomainClass {
name SPIDER_MITE
comment "Description of a Spider Mite"
SuperClass PEST_INFECTION
SPIDER_MITE_ABDOMEN **name** abdomen
SPIDER_MITE_DARK_SPOT **name** darkspot1
SPIDER_MITE_DARK_SPOT **name** darkspot2 **}**

DomainClass {
name SPIDER_MITE_DARK_SPOT
comment "The dark spot on the back of a spider mite"
Visual Description
Spatial Description
Geometric_Surface **name** spot_shape
range [*Elliptical Oblong*]
Size **name** spot_size
range [*Large*]
Color Description
Neutral_Color **name** spot_color
range [*Black*]
Lightness **name** spot_lightness
range [*Dark*]
Relational Description
TPP **name** spot_tangencial_proper_part_of
range [WHITE_FLY_ABDOMEN] **}**

DomainClass {
name SPIDER_MITE_ABDOMEN
comment "General description of a white fly abdomen"
SuperClass DOMAINCLASS
Visual Description
Spatial Description
Geometry **name** SMabdomen_shape
range [*Elliptical_Surface*]
Symmetry **name** SMabdomen_symmetry
range [*PrincipalAxisR_symmetrical OrthogonalAxisR_symmetrical*]
Color Description
Hue **name** SMabdomen_color
range [*Brown Brownish_Orange Reddish_Orange Green Greenish_Yellow*] **}**

DomainClass {
name APHID_INFECTION
comment "Presence of Aphid on the leaf"
SuperClass PEST_INFECTION
SubPart Description
PEST_INFECTION.pestinfection_symptom
range [APHID_BODY] **}**

DomainClass {
name APHID_COLONY
SuperClass APHID_INFECTION **}**

B.1.3 Domain Context and Acquisition Context Definition

DomainContext {
  name ROSESIM_Domain_Context
  comment *The Domain context of the rose disease diagnosis application*
  Attributes
    Float name Relative_humidity
      default no
      range [0 100]
    Symbol name Greenhouse_temperature
      default no
    Symbol name Season
      range [Winter Spring Summer Autumn]
    Symbol name Plant_Location
      range [Entrance Middle back]
    Symbol name Organ_Location
      range [Bottom Middle Top]
    Symbol name Rose_Variety
      range [Leonidas Texas FirstRed]
    Symbol name Leaf_Age
      range [Very_Young Young Mature Old] }

AcquisitionContext {
  name ROSESIM_Acquisition_Context
  comment *The Acquisition context of the rose disease diagnosis application*
  Attributes
    Symbol name Sensor_type
      default Binocular_microscope
      range [Binocular_microscope Video_microscope microscope]
    Float name Sensor_magnification
      default 64
      range [7.5 200]
    Float name Sensor_resolution
      range [0.25 0.35 0.5 1.0 4.0] 10 E-4 }

B.1.4 Context Criteria

B.1.4.1 Initialization Criteria

Rule {
  name *Context_Age_1*
  comment *“”*
  LinkedDomainClass LEAF
  Let *c a **Domain context** and O a **Visual object***
  If *C.Leaf_Age =Very_Young*
  Then *O.hue:=Red* }

Rule {

name *Context_Age_2*
comment *“”*
LinkedDomainClass LEAF
Let *c a* ***Domain context*** *and O a* ***Visual object***
If *C.Leaf_Age =Young*
Then *O.hue:=Green* }

Rule {
name *Context_Age_3*
comment *“”*
LinkedDomainClass LEAF
Let *c a* ***Domain context*** *and O a* ***Visual object***
If *C.Leaf_Age =Mature*
Then *O.hue:=Green* }

Rule {
name *Context_Variety_1*
comment *“”*
LinkedDomainClass FUNGAL_INFECTION
Let *c a* ***Domain context*** *and O a* ***Visual object***
If *c.Greenhouse_temperature< 25 and c.Relative_humidity > 80 and c.Rose_Variety= Leonidas*
Then ***setImportance*** *Fungal_Infection 1* }

Rule {
name *Context_humidity_1*
comment *“”*
LinkedDomainClass SPIDER_MITE_INFECTION
Let *c a* ***Domain context*** *and O a* ***Visual object***
If *c.Greenhouse_temperature > 23 and <30 and c.Relative_humidity < 50*
Then ***setImportance*** *Spider_Mite_Infection 1* }

Rule {
name *Context_season_1*
comment *“”*
LinkedDomainClass APHID_INFECTION
Let *c a* ***Domain context*** *and O a* ***Visual object***
If *c.Season = Summer*
Then ***setImportance*** *Aphid_Infection 1* }

B.1.4.2 Post Interpretation Criteria

Rule {
name *Powdery_Mildew_Diagnosis1*
comment *“”*
LinkedDomainClass SPREADING_POWDERY_MILDEW

Let *c a* ***Domain context***
If c.Greenhouse_temperature< 25 and c.Relative_humidity > 80
Then ALERT "A Powdery Mildew treatment is needed " }

Rule {
name *Powdery_Mildew_Diagnosis2*
comment *""*
LinkedDomainClass HARMLESS_POWDERY_MILDEW
Let *c a* ***Domain context***
If c.Greenhouse_temperature< 25 and c.Relative_humidity > 80
Then ALERT "A Powdery Mildew monitoring is needed "}

Rule {
name *Powdery_Mildew_Diagnosis3*
comment *""*
LinkedDomainClass SPREADING_POWDERY_MILDEW
Let *c a* ***Domain context***
If c.Greenhouse_temperature > 25 and c.Relative_humidity < 80
Then ALERT "Possible Powdery Mildew Infection "}

B.2 The ROSESIM Visual Data Management Knowledge Base

The ROSESIm visual data management knowledge base contains:

- The description of 107 visual concepts (shape visual concepts or color visual concepts) with their associated grounding link and their potential linked object extraction criteria. We don't give here the complete description of all the visual concepts.
- The description of the 22 spatial relations defined in the spatial relation ontology
- The definition of basic image data and descriptor set
- The definition of visual data management criteria

B.2.1 Shape Visual Concepts

VisualConcept {
name *ShapeConcept* **}**

B.2.1.1 Geometric Concepts

VisualConcept {
name *Geometry*
SuperConcept *ShapeConcept* **}**

VisualConcept {

name *Point*
SuperConcept *Geometry* **}**

VisualConcept {
name *Point_Set*
SuperConcept *Geometry*
ObjectExtractionCriteria
Rule {
name *PointSet_extraction*
comment *“”*
Let c a **visual content context** and O a visual Object
If O.geometry **in** [*Point_Set*]
Then c.Image_Entity_Type:= Class_Of_Pixels **} }**

VisualConcept {
name *Curve*
SuperConcept *Point_Set* **}**

VisualConcept {
name *Open_Curve*
SuperConcept *Curve*
ObjectExtractionCriteria
Rule {
name *OpenCurve_extraction*
Let c a **visual content context** and O a **visual object**
If O.geometry **in** [*Open_Curve*] and O.thickness **in** [*Thin Very_Thin*]
Then c.Image_Entity_Type:= Curvilinear_Structure **} }**

VisualConcept {
name *Close_Curve*
SuperConcept *Curve*
ObjectExtractionCriteria
Rule {
name *CloseCurve_extraction*
Let c a **visual content context** and O a visual Object
If O.geometry **in** [*Close_Curve*] and O.thickness **in** [*Thin Very_Thin*]
Then c.Image_Entity_Type:= Edge **} }**

VisualConcept {
name *Line*
SuperConcept *Open_Curve*
Grounding Link
Symbol **name** Elongatdness
Linguistic range [Important Medium]
Symbol **name** Width
Linguistic range [Small Medium] **}**

VisualConcept {
name *Segment*

SuperConcept *Open_Curve*
Grounding Link
... }

VisualConcept {
name *Geometric_Surface*
SuperConcept *Point_Set*
ObjectExtractionCriteria
Rule {
name *GeometricSurface_extraction*
Let c a **visual content context** and O a visual Object
If O.geometry **in** [*GeometricSurface*]]
Then c.Image_Entity_Type:= Region **} }**

VisualConcept {
name *Non_Geometric_Surface*
SuperConcept *Point_Set*
ObjectExtractionCriteria
Rule {
name *Non_GeometricSurface_extraction*
comment *“”*
Let c a **visual content context** and O a visual Object
If O.geometry **in** [*Nom_Geometric_Surface*]]
Then c.Image_Entity_Type:= Region **} }**

VisualConcept {
name *Elliptical_Surface*
SuperConcept *Geometric Surface*
Grounding Link
Symbol name eccentricity
comment *ratio of the lenght of the longest chord to the longest chord perpendicular to it*
Linguistic range: [very_low low medium high very_high]
FuzzySet
$F_{verylow} = \{0.0, 0.0, 0.19, 0.21\}$
$F_{low} = \{0.19, 0.21, 0.38, 0.42\}$
$F_{medium} = \{0.38, 0.42, 0.575, 0.625\}$
$F_{high} = \{0.575, 0.625, 0.76, 0.84\}$
$F_{veryhigh} = \{0.76, 0.84, 1, 1\}$
Domain: [0 1]
unit: none
Symbol name compactness
comment *Measure of how the shape is closely-packed*
...
Symbol name ellipticity
comment *Euclidian ellipticity : distance between fitting ellipse and region boundary*
... }

VisualConcept {
name *Circular_Surface*
SuperConcept *Elliptical_Surface*
Grounding Link
Symbol name eccentricity
comment *ratio of the lenght of the longest chord to the longest chord perpendicular to it*
Linguistic range: [high very_high]
FuzzySet
$F_{high} = \{0.575, 0.625, 0.76, 0.84\}$
$F_{veryhigh} = \{0.76, 0.84, 1, 1\}$
Domain: [0 1]
unit: none
Symbol name compactness
comment *Measure of how the shape is closely-packed*
Linguistic range: [high very_high]
...
Symbol name circularity
comment *Shape factor*
... }

B.2.1.2 Elongation Concept

The elongation of an object is a global metric property useful to describe the shape of an object. In [Clementini and Felice, 1997], a qualitative system about the elongation of an object is proposed. The qualitative system is organized along various levels of granularity: two levels (compact or elongated) or a level with four distinctions (styrongly compact, nearly compact, moderately elongated and strongly elongated). This property is defined by the ratio of the length to the width of the minimum bounding rectangle and with the operators: $=, \sim=, \sim<, <$, and $\ll$ as in [Clementini and Felice, 1997]. We instanciate these concepts as:

VisualConcept {
name *Elongation*
SuperConcept *ShapeConcept*
Grounding Link
Symbol **name** Elongatedness
comment "Ratio of the length to the width of the minimum bounding rectangle"
Linguistic range [very_small, small, high, very_high]
FuzzySet ...
Domain [0 1] **}**
Symbol **name** Axial_Elongatedness
comment "Ratio of the length to the width of the principal axis"
Linguistic range [very_small, small, high, very_high]
FuzzySet ...
Domain [0 1] **}**

VisualConcept {
name *Strongly_Elongated*
SuperConcept *Elongation*

Grounding Link
Elongation.Elongatedness
Linguistic range [very_small] **}**

VisualConcept {
name *Moderately_Elongated*
SuperConcept *Elongation*
Grounding Link
Elongation.Elongatedness
Linguistic range [small] **}**

VisualConcept {
name *Moderately_Squat*
SuperConcept *Elongation*
Grounding Link
Elongation.Elongatedness
Linguistic range [high] **}**

VisualConcept {
name *Strongly_Squat*
SuperConcept *Elongation*
Grounding Link
Elongation.Elongatedness
Linguistic range [very_high] **}**

B.2.1.3 Compactness concepts

Compactness expresses how tighly a shape is *packed*

VisualConcept {
name *Compactness*
SuperConcept *ShapeConcept*
Grounding Link
Symbol **name** Compactness
Linguistic range [very_small, small, high, very_high]
FuzzySet ...
Domain [1 infinity] **}**

VisualConcept {
name *Highly_Compact*
SuperConcept *Compactness*
...

VisualConcept {
name *Non_Compact*
SuperConcept *Compactness*
... }

B.2.1.4 Symmetry concepts

Symmetry is an important feature in the human visual system. Most of natural objects are strongly constrained by symmetry. It is a key feature to characterize natural objects. Some forms have no symmetry axis while other forms have one, two or even more symmetry axes.

VisualConcept {
name *Symmetry*
SuperConcept *ShapeConcept*
Grounding Link
Symbol **name** Asymetry_principal_axisreflectional_factor
comment “Using distance list see [Fischer et al., 2000]”
...
Symbol **name** Asymetry_orthogonal_axisreflectional_factor
comment “Using distance list see [Fischer et al., 2000]”
...
Symbol **name** Asymetry_rotational_factor
comment “Using distance list see [Fischer et al., 2000]”
... **}**

VisualConcept {
name *Asymmetrical*
SuperConcept *Symmetry*
... }

VisualConcept {
name *Reflectionally_symmetrical*
SuperConcept *Symmetry*
... }

VisualConcept {
name *PrincipalAxisR_symmetrical*
SuperConcept *Reflectionally_symmetrical*
... }

VisualConcept {
name *OrthogonalAxisR_symmetrical*
SuperConcept *Reflectionally_symmetrical*
... }

VisualConcept {
name *Rotationnaly_symmetrical*
SuperConcept *Symmetry*
... }

B.2.1.5 Size concepts

They refer to the quality of an object that determineshow much space it occupies: its dimensions or magnitudes.

VisualConcept {
name *Size Concept*
Grounding Link
Symbol **name** area
Linguistic range [very_small, small, medium, important, very_important]
Symbol **name** perimeter
Linguistic range [very_small, small, medium, important, very_important] **}**

VisualConcept {
name *Length*
SuperConcept *Size Concept*
Grounding Link
Symbol **name** Feretmaxdiameter
Linguistic range [very_small, small, medium, important, very_important]
Symbol **name** principal_axis_lenght
Linguistic range [very_small, small, medium, important, very_important]
Symbol **name** equivalent_diameter
Linguistic range [very_small, small, medium, important, very_important] **}**

VisualConcept {
name *Width*
SuperConcept *Size Concept*
Grounding Link
Symbol **name** Feretmindiameter
Linguistic range [very_small, small, medium, important, very_important]
Symbol **name** orthognal_axis_lenght
Linguistic range [very_small, small, medium, important, very_important] **}**

B.2.1.6 Periphery concepts

VisualConcept {
name *Periphery*
SuperConcept *Shape Concept*
Grounding Link
Symbol **name** axialboundarydistancecurveappearance
Symbol **name** roundness **}**

VisualConcept {
name *Smooth*
SuperConcept *Periphery*
Grounding Link
... }

VisualConcept {
name *Rough*
SuperConcept *Periphery*
Grounding Link
... }

VisualConcept {
name *Hairy*
SuperConcept *Periphery*
Grounding Link
... }

VisualConcept {
name *highly_Hairy*
SuperConcept *Periphery*
Grounding Link
... }

B.2.1.7 Topology

VisualConcept {
name *Topology*
SuperConcept *Shape Concept*
Grounding Link
Integer **name** euler_number **}**

VisualConcept {
name *Holed*
SuperConcept *Topology*
... }

VisualConcept {
name *Unholed*
SuperConcept *Topology*
... }

B.2.1.8 Spatial Structures

VisualConcept {
name *Spatial_Structure*
SuperConcept *ShapeConcept* **}**

VisualConcept {
name *Network_of*

SuperConcept *Spatial_Structure* **}**

VisualConcept {
name *Rows_of*
SuperConcept *Spatial_Structure* **}**

VisualConcept {
name *Circle_of*
SuperConcept *Spatial_Structure* **}**

B.2.2 Color Visual Concepts

VisualConcept {
name *Color Concept*

VisualConcept {
name *Hue*
SuperConcept *Color Concept*
Grounding Link
Integer **name** R_value
Domain [0 255] **}**
Integer **name** G_value
Domain [0 255] **}**
Integer **name** B_value
Domain [0 255] **}**
Float **name** H_value
Domain [0 1] **}**
Float **name** S_value
Domain [0 1] **}**
Float **name** L_value
Domain [0 1] **}**

VisualConcept {
name *Generic_Hue*
SuperConcept *Hue* **}**

VisualConcept {
name *Red*
SuperConcept *Generic_Hue*
Hue.H_value
Domain [0.9 1.0] **}**

VisualConcept {
name *Orange*
SuperConcept *Generic_Hue*
Hue.H_value
Domain [0.0 0.1]

Hue.L_value
Domain [0.5 1.0] **}**

VisualConcept {
name *Yellow*
SuperConcept *Generic_Hue*
Hue.H_value
Domain [0.1 0.2] **}**

VisualConcept {
name *Green*
SuperConcept *Generic_Hue*
Hue.H_value
Domain [0.2 0.3] **}**

VisualConcept {
name *Blue*
SuperConcept *Generic_Hue*
Hue.H_value
Domain [0.5 0.6] **}**

VisualConcept {
name *Violet*
SuperConcept *Generic_Hue*
Hue.H_value
Domain [0.6 0.7] **}**

VisualConcept {
name *Purple*
SuperConcept *Generic_Hue*
Hue.H_value
Domain [0.7 0.8] **}**

VisualConcept {
name *Pink*
SuperConcept *Generic_Hue* **}**

VisualConcept {
name *Brown*
SuperConcept *Generic_Hue*
Hue.H_value
Domain [0.0 0.1]
Hue.L_value
Domain [0.0 0.5] **}**

VisualConcept {
name *Composite_Hue*
SuperConcept *Hue*
Grounding Link

... }

VisualConcept {
name *Reddish Orange*
SuperConcept *Generic_Hue*
... }

VisualConcept {
name *Reddish Purple*
SuperConcept *Generic_Hue*
... }

VisualConcept {
name *Orange Yellow*
SuperConcept *Generic_Hue*
... }

VisualConcept {
name *Greenish Yellow*
SuperConcept *Generic_Hue*
... }

VisualConcept {
name *Yellowish Green*
SuperConcept *Generic_Hue*
... }

VisualConcept {
name *Greenish Yellow*
SuperConcept *Generic_Hue* **}**

VisualConcept {
name *White*
SuperConcept *Neutral_Color*
Neutral_Color.L_value
Domain [0.9 1.0] **}**

VisualConcept {
name *Black*
SuperConcept *Neutral_Color*
Neutral_Color.L_value
Domain [0.0 0.1] **}**

VisualConcept {
name *Grey*
SuperConcept *Neutral_Color*
Neutral_Color.L_value
Domain [0.1 0.9] **}**

VisualConcept {
name *Color_Homogeneity*
SuperConcept *Color Concept*
Grounding Link
Symbol **name** intensity_variance **}**

VisualConcept {
name *Translucidity*
SuperConcept *Color Concept*
Grounding Link
Symbol **name** intensity_variance **}**

VisualConcept {
name *Luminosity*
SuperConcept *Color Concept*
Grounding Link
Integer **name** R_value
Domain [0 255] **}**
Integer **name** G_value
Domain [0 255] **}**
Integer **name** B_value
Domain [0 255] **}**
Float **name** L_value
Domain [0 1] **}**

VisualConcept {
name *Bright*
SuperConcept *Luminosity*
Luminosity.L_value
Domain [0.8 0.9] **}**

VisualConcept {
name *Dark*
SuperConcept *Luminosity*
Luminosity.L_value
Domain [0.1 0.3] **}**

B.2.3 Spatial Relations

SpatialRelation {
name *Topological_Relation* }

SpatialRelation {
name *Proper_Part_Of*
SuperRelation *TopologicalRelation*
Inverse *Has_For_Proper_Part*
Complement *Discrete*

Symmetry *False*
Objtects_In_Relation
VisualObject **name** object1
VisualObject **name** object2
Conditions
Difference(object1.imagedata.Interior,object2.imagedata.Interior):= empty
Difference(object2.imagedata.Interior,object1.imagedata.Interior) != not empty
Intersection(object1.imagedata.Interior,object2.imagedata.Interior) != not empty }

SpatialRelation {
name *Has_Proper_Part_Of*
SuperRelation *TopologicalRelation*
Inverse *Proper_Part_Of*
Complement *Discrete*
Symmetry *False*
Objtects_In_Relation
VisualObject **name** object1
VisualObject **name** object2
Conditions
Difference(object1.imagedata.Interior,object2.imagedata.Interior):= empty
Difference(object2.imagedata.Interior,object1.imagedata.Interior) != not empty
Intersection(object1.imagedata.Interior,object2.imagedata.Interior) != not empty }

SpatialRelation {
name *Discrete*
SuperRelation *TopologicalRelation*
Inverse *Discrete*
Complement *Has_For_Proper_Part*
Symmetry *True*
Objtects_In_Relation
VisualObject **name** object1
VisualObject **name** object2
Conditions
Difference(object1.imagedata.Interior,object2.imagedata.Interior):= empty
Difference(object2.imagedata.Interior,object1.imagedata.Interior) != not empty
Intersection(object1.imagedata.Interior,object2.imagedata.Interior) != not empty }

SpatialRelation {
name *Equals*
SuperRelation *TopologicalRelation*
Inverse *Equals*
Complement *Discrete*
Symmetry *True*
Objtects_In_Relation
VisualObject **name** object1
VisualObject **name** object2
Conditions

Difference(object1.imagedata.Interior,object2.imagedata.Interior):= empty
Difference(object2.imagedata.Interior,object1.imagedata.Interior) != not empty
Intersection(object1.imagedata.Interior,object2.imagedata.Interior) != not empty}

SpatialRelation {
name *Partial_Overlaps*
SuperRelation *TopologicalRelation*
Inverse *Partial_Overlaps*
Complement *None*
Symmetry *False*
Objtects_In_Relation
VisualObject **name** object1
VisualObject **name** object2
Conditions
Difference(object1.imagedata.Interior,object2.imagedata.Interior):= empty
Difference(object2.imagedata.Interior,object1.imagedata.Interior) != not empty
Intersection(object1.imagedata.Interior,object2.imagedata.Interior) != not empty}

SpatialRelation {
name *Distance_Relation* }

SpatialRelation {
name *Near_Of*
SuperRelation *DistanceRelation*
Inverse *Near_Of*
Complement *Far_From*
Symmetry *True*
Objtects_In_Relation
VisualObject **name** object1
VisualObject **name** object2
Conditions
Distance(object1.imagedata.centerofgravity,object2.imagedata.centerofgravity) <= 2 * object2.imagedata.size}

SpatialRelation {
name *Orientation_Relation* }

SpatialRelation {
name *Left_Of*
SuperRelation *Orientation_Relation*
Inverse *Right_Of*
Complement *None*
Symmetry *False*
Objtects_In_Relation
VisualObject **name** object1
VisualObject **name** object2

Conditions
Angle(object1.imagedata.centerofgrac=vity,object2.imagedata.principalaxis) > -PI/2 and <PI/2}

B.2.3.1 Object Extraction Criteria related to Spatial Relation

Rule {
name *Proper_Part_Extraction1*
Spatial Relation *Proper_Part*
Let c a **visual content context** and O1 a visual object and O2 a visual object
If ProperPart(O1, O2)= true and O1.hue **in** [*Red*] and O2.hue **in** [*Green*]
Then c.Discriminative_Object_Color:= High **} }**
Rule {
name *Proper_Part_Constraint1*
Spatial Relation *Proper_Part*
Let c a **visual content context** and O1 a visual object and O2 a visual object
If ProperPart(O1, O2)= true and O1.hue **in** [*Red*] and O2.hue **in** [*Green*]
Then c.area_of_interest:= O2.ImageData.Interior **} }**

B.2.3.2 Spatial Deduction Criteria

Rule {
name *RightOf_transitivity*
Spatial Relation *Right_Of*
Let O1 a visual object, O2 a visual object and O3 a visual object
If Right_of(O1, O2)= true Right_of(O2, O3)= true
Then Right_of(O1, O3):=true **} }**
Rule {
name *NTPPRightOf_composition*
Spatial Relation *NTTP*
Let O1 a visual object, O2 a visual object and O3 a visual object
If NTTP(O1, O2)= true Right_of(O2, O3)= true
Then Right_of(O1, O3):=true **} }**

B.2.3.3 Evaluation Criteria

Rule {
name *imageprocessingevaluation*
If true
Then assess_data_by_user [correct incorrect under_segmentation over_segmentation]
} }

B.3 The ROSESIM Program Supervision Knowledge Base

```
Import types

Argument Type{
        name VisualContentContext
        Attributes
        File name contextFile
        Image name areaOfInterest
        String name primitiveType
                range ["Line" "Region" "Edge"]
        default "Region"
        String name discriminativeColor
                range ["yes" "no"]
                default "no"
        String name discriminativeTexture
                range ["yes" "no"]
                default "no"
        String name discriminativeIntensity
                range ["yes" "no"]
                default "no"
        String name objectLuminosity
                range ["bright" "medium" "dark"]
        Integer name minobjectSize
                default 1
        Integer name maxobjectSize
        Integer name numberOfObjects
                default 1
        String name BackgroundAppearance
                range ["Homogeneous" "Textured"]
        String name objectPosition
        range ["Top" "Centered" "Bottom" "Left" "Right"]
                default "Centered"
        String name objectRepartition
        range ["Dispersed" "Grouped"]

Methods void loadContext()

}




#========================================================
# Functionality : ObjectExtraction
#========================================================

Functionality{
        name ObjectExtraction
        comment "Top down extraction of an object from an image "
        Achieved by ObjectExtraction_operator
        Input Data
                Image name inputImage
                Image name maskImage
                VisualContentContext name inputContext
        Output Data
                Image name segmentedImage
}


#========================================================
```



```
# Composite Operator : ObjectExtraction_operator
#========================================================



Composite Operator {
         name  ObjectExtraction_operator

Functionality
        ObjectExtraction

Input Data
         Image name  inputImage
         comment  "Image d'entree"

        Image name  maskImage
         comment  "Image representant la zone d'interet si elle existe"

         VisualContentContext name inputContext


Output Data
        Image name segmentedImage



Preconditions
        valid inputImage

Postconditions
        valid segmentedImage



Initialization criteria
Rule {
        name  init_context
        comment  "Initialization of the context with the file"
                If      true
                Then
                inputContext.loadContext()
        }




Body
 Edge_segmentation_operator | Region_segmentation_operator | BackgroundSubtracti
on_operator | CurvilinearDetection_operator | Treshold_operator


Choice criteria
Rule{
        name ChoiceExtraction1
        If inputContext.primitiveType == "Line"
        Then
        use_operator CurvilinearDetection_operator
}

Rule{
        name ChoiceExtraction2
        If inoutContext.primitiveType == "Edge",
```

```
        Then
        use_operator Edge_segmentation_operator

}


Rule{
        name ChoiceExtraction3
        If inputContext.primitiveType == "Region", inoutContext.BackgroundAppear
ance == "Homogeneous", inputContext.objectPosition == "Centered"
        Then
        use_operator BackgroundSubtraction_operator

}


Rule{
        name ChoiceExtraction4
        If inputContext.primitiveType == "Region", inputContext.discriminativeCo
lor == "yes"
        Then
        use_operator Region_segmentation_operator

}


Rule{
        name ChoiceExtraction5
        If inputContext.primitiveType == "Region"
        Then
        use_operator Region_segmentation_operator

}



Rule{
        name ChoiceExtraction5
        If inputContext.primitiveType == "Region", inputContext.discriminativeIn
tensity == "yes"
        Then
        use_operator Threshold_operator

}


Rule{
        name ChoiceExtraction6
        If inputContext.primitiveType == "Edge", inputContext.discriminativeInte
nsity == "yes"
        Then
        use_operator Threshold_operator

}


Distribution
        ObjectExtraction_operator.inputImage/ Edge_segmentation_operator.inputIm
age
        ObjectExtraction_operator.maskImage/ Edge_segmentation_operator.maskImag
e
        ObjectExtraction_operator.segmentedImage/ Edge_segmentation_operator.seg
```



```
mentedImage

        ObjectExtraction_operator.inputImage/ Region_segmentation_operator.input
Image
        ObjectExtraction_operator.maskImage/ Region_segmentation_operator.maskIm
age

        ObjectExtraction_operator.inputImage/ BackgroundSubtraction_operator.inp
utImage
        ObjectExtraction_operator.maskImage/ BackgroundSubtraction_operator.mask
Image
        ObjectExtraction_operator.segmentedImage/ BackgroundSubtraction_operator
.segmentedImage

        ObjectExtraction_operator.inputImage/ CurvilinearDetection_operator.inpu
tImage
        ObjectExtraction_operator.maskImage/ CurvilinearDetection_operator.maskI
mage
        ObjectExtraction_operator.segmentedImage/ CurvilinearDetection_operator.
segmentedImage

        ObjectExtraction_operator.inputImage/ Threshold_operator.inputImage
        ObjectExtraction_operator.maskImage/ Threshold_operator.maskImage
        ObjectExtraction_operator.segmentedImage/ Threshold_operator.segmentedIm
age

}










#=========================================================
# Composite Operator :CurvilinearDetection_operator
#=========================================================

Composite Operator {
         name  CurvilinearDetection_operator
         comment  "Detection of curvilinear structures in images"


Input Data
        Image name  inputImage
        Image name maskImage
        VisualContentContext name inputContext

Input Parameters
        Integer name sigma
        comment  "Standard deviation of the gaussian filter, representative of t
he width of the ridge"
        default 1
```

```
        Symbol name curvilinear_type
        comment "Ridge or Valley"
        range ["Ridge" "Valley" "Both"]
        default "Both"

Output Data
        Image name  segmentedImage
        Image name  ridge_image_properties


Preconditions
        valid inputImage

Postconditions
        valid segmentedImage


Initialisation criteria
Rule {
        name  init_sigma
        comment  "sigma should be set approximately at expected object size / sq
rt(3)"
                If true
                Then
                sigma:= (inputContext.minobjectSize)/1.732
        }


Rule {
        name  init_curvilinearType1
        comment  "What curvilinear feature"
                If imputContext.objectLuminosity == "bright"
                Then
                curvilinear_type:= "Ridge"
        }


Rule {
        name  init_curvilinearType2
        comment  "What curvilinear feature"
                If imputContext.objectLuminosity == "dark"
                Then
                curvilinear_type:= "Valley"
        }






Body
 Recursive_Gaussian_Derivative_operator − Steger_Ridge_Point_Detector − Ridge_Fi
ltering_Operator

Distribution
  CurvilinearDetection_operator.inputImage/ Recursive_Gaussian_Derivative_operat
or.inputImage
  CurvilinearDetection_operator.sigma/ Recursive_Gaussian_Derivative_operator.si
gma
  CurvilinearDetection_operator.curvilinear_type/ Steger_Ridge_Point_Detector.cu
rvilinear_type
```



```
  CurvilinearDetectionoperator.maskImage / Ridge_Filtering_Operator.maskImage
  CurvilinearDetectionoperator.segmentedImage / Ridge_Filtering_Operator.segment
edImage
  CurvilinearDetectionoperator.ridge_image_properties / Steger_Ridge_Point_Detec
tor.ridge_image_properties


Flow
  Recursive_Gaussian_Derivative_operator.mfxx / Steger_Ridge_Point_Detector.mfxx
  Recursive_Gaussian_Derivative_operator.mfyy / Steger_Ridge_Point_Detector.mfyy
  Recursive_Gaussian_Derivative_operator.mfxy / Steger_Ridge_Point_Detector.mfxy
  Recursive_Gaussian_Derivative_operator.mfdx / Steger_Ridge_Point_Detector.mfdx
  Recursive_Gaussian_Derivative_operator.mfdy / Steger_Ridge_Point_Detector.mfdy
  Steger_Ridge_Point_Detector.ridgeImage /  Ridge_Filtering_Operator.ridgeImage
}














#================================================================
# Primitive Operator : RecursiveGaussianDerivativeComputing
#===============================================================


Primitive Operator{
        name   Recursive_Gaussian_Derivative_operator
        comment  "Computing for a given sigma of a set of recursive gaussian der
ivatives "

Input Data
        Image name inputImage

Input Parameters
        Integer name sigma
        comment  "Standard deviation of the gaussian filter, representative of t
he width of the ridge"
        default 1


Output Data
        Image name  mfxx
        comment  "second gaussian derivative in the direction X"

        I−O relations
     mfxx.path := inputImage.path,
         mfxx.basename := inputImage.basename + "mfxx",
         mfxx.extension := ".pan"

        Image name  mfyy
        comment  "second gaussian derivative in the direction Y"
```

```
        I−O relations
         mfyy.path := inputImage.path,
         mfyy.basename := inputImage.basename + "mfyy",
         mfyy.extension := ".pan"

        Image name  mfxy
        comment  "second gaussian derivative in the direction XY"
        I−O relations
         mfxy.path := inputImage.path,
         mfxy.basename := inputImage.basename + "mfxy",
         mfxy.extension := ".pan"

        Image name  mfdx
        comment "first gaussian derivative in the direction X"
        I−O relations
         mfdx.path := inputImage.path,
         mfdx.basename := inputImage.basename + "mfdx",
         mfdx.extension := ".pan"

        Image name  mfdy
        comment  "first gaussian derivative in the direction Y"
        I−O relations
     mfdy.path := inputImage.path,
         mfdy.basename := inputImage.basename + "mfdy",
         mfdy.extension := ".pan"




Preconditions
        valid inputImage


Postconditions
        valid mfxx,
        valid mfyy,
        valid mfxy,
        valid mfdx,
        valid mfdy



Call

        language  shell
        syntax /user/chudelot/home/pandore/pandore4/application/script/GaussianD
erivatives sigma inputImage.get_filename() mfxx.get_filename() mfyy.get_filename
() mfxy.get_filename() mfdx.get_filename() mfdy.get_filename()
        endsyntax
        program name GaussianDerivatives
        type  real




}
```



```
#===========================================
# Primitive Operator : Steger_Ridge_Point_Detector
#===========================================

Primitive Operator{
        name  Steger_Ridge_Point_Detector
        comment  " Differential geometry approach to detect ridge :Sub−pixel rid
ge extraction : Carsten Steger approach "


Input Data
        Image name  mfxx
        comment  "second gaussian derivative in the direction X"


        Image name  mfyy
        comment  "second gaussian derivative in the direction Y"

        Image name  mfxy
        comment  "second gaussian derivative in the direction XY"

        Image name  mfdx
        comment "first gaussian derivative in the direction X"


        Image name  mfdy
        comment  "first gaussian derivative in the direction Y"

Input Parameters

        Symbol name curvilinear_type
        comment "Ridge or Valley"
        range ["Ridge" "Valley" "Both"]
        default "Both"

        Float name  thres_first_der
        comment  "First derivative threshold above which a point is a possible r
idge point"
        default  0.1
        range  [0.0 ,1.0]

        Float name  thresh_second_der
        comment  "abs(2nde derivative) threshold above which a point is a possib
le ridge point"
        default  0.5
        range  [0.0, 1.0]

        Float name  thresh_first_second
        comment  "maximum allowed distance between vanishing first derivative an
d maximum second derivative"
        default  0.55
        range  [0.0 , 1.0]



Output Data
        MyImage name  ridgeImage
        comment  "      image of ridge point "
        I−O Relations
        ridgeImage.path := mfxx.path,
        ridgeImage.basename := "ridge",
        ridgeImage.extension := ".pan"
```

```
        MyImage name  ridge_image_properties
        comment  "      image of ridge point "
        I−O Relations
        ridge_image_properties.path := mfxx.path,
        ridge_image_properties.basename := "ridge_properties",
        ridge_image_properties.extension := ".pan"


Preconditions
        valid mfxx,
        valid mfyy,
        valid mfxy,
        valid mfdx,
        valid mfdy

Postconditions
        valid ridgeImage,
        valid ridge_image_properties



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/LineStegerDetection2  t
hres_first_der  thresh_second_der  thresh_first_second mfxx.get_filename() mfyy.
get_filename() mfxy.get_filename() mfdx.get_filename() mfdy.get_filename() ridge
Image.get_filename() ridge_image_properties.get_filename() −type curvilinear_typ
e
        endsyntax
        program name  LineStegerDetection2
        type  real


}



#============================================
# Primitive Operator : Ridge_Filtering_Operator
#============================================

Primitive Operator{
        name  Ridge_Filtering_Operator
        comment  " Operations to filter steger ridges "


Input Data
        Image name  ridgeImage
        Image name  maskImage

Output Data
        Image name  segmentedImage
        comment  "      image of line "
        I−O Relations
        segmentedImage.path := ridgeImage.path,
        segmentedImage.basename := "Lines",
        segmentedImage.extension := ".tif"
```



```
Preconditions
        valid ridgeImage


Postconditions
        valid segmentedImage

Assessment criteria
Rule{
        name eval1
        If true
        Then
        segmentedImage.displayImage(),
        assess_data_by_user segmentedImage "Segmentation is correct?" ["incorrec
t" "correct"]

}

Rule{
        name eval2
        If assess_data? segmentedImage incorrect
        Then
        assess_operator bad repair

}
Rule{
        name eval3
        If assess_data? segmentedImage correct
        Then
        assess_operator ok continue

}



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/LineFilt
ering ridgeImage.get_filename() segmentedImage.get_filename() −m maskImage
        endsyntax
        program name  LineFiltering
        type  real


}




#=========================================================
# Primitive Operator : BackgroundSubtraction_operator
#=========================================================




Primitive Operator{
        name BackgroundSubtraction_operator
        comment  "Detect the pixel belonging to the homogeneous background in im
age border"
```

```
Input Data
        Image name  inputImage
        Image name  maskImage
        VisualContentContext name inputContext


Input Parameters

        Symbol name borderParts
        comment  "Which parts of the border is considered"
        range ["TopLess" "BottomLess" "LeftLess" " RightLess" "All"]
        default "All"


        Symbol name filterType
        comment  "Smoothing filter used"
        range ["Nothing" "Median" "KNearest"]
        default "Median"

        Symbol name objectLabeling
        range ["yes" "no"]
        default "yes"


Output Data
        MyImage name  segmentedImage

        I−O Relations
        segmentedImage.path := inputImage.path,
        segmentedImage.basename := "regions",
        segmentedImage.extension := ".pan"


Preconditions
        valid inputImage

Postconditions
        valid segmentedImage




Initialisation criteria
Rule {
        name  init_border1
        comment  "if object is non center"
                If inputContext.objectPosition== "Top"
                Then
                borderPart:="TopLess"
        }

Rule {
        name  init_border2
        comment  "if object is non center"
                If inputContext.objectPosition== "Bottom"
                Then
                borderPart:="BottomLess"
        }

Rule {
```



```
        name  init_border3
        comment  "if object is non center"
                If inputContext.objectPosition== "Left"
                Then
                borderPart:="LeftLess"
        }

Rule {
        name  init_border4
        comment  "if object is non center"
                If inputContext.objectPosition== "Right"
                Then
                borderPart:="RightLess"
        }



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/Backgrou
ndSegmentation  inputImage.get_filename() segmentedImage.get_filename() −f filte
rType −b borderParts −l objectLabeling −m maskImage
        endsyntax
        program name csPresegmentation
        type  real

}









#==========================================================
# Composite Operator : Region_segmentation_operator
#==========================================================

Composite Operator {
         name  Region_segmentation_operator
         comment  "Region based segmentation in images"


Input Data
        Image name  inputImage
        Image name maskImage
        VisualContentContext name inputContext
        Symbol name segmentationType
        range ["Robust" "Approximative"]
        default "Robust"



Output Data
        Image name  segmentedImage



Preconditions
```

```
        valid inputImage

Postconditions
        valid segmentedImage



Body
 ImageColorSegmentation_operator − SegmentationEnhancement_operator − RegionSele
ction_Operator − BoundaryExtraction_Operator

Distribution
        Region_segmentation_operator.inputImage/ ImageColorSegmentation_operator
.inputImage


Flow
  ImageColorSegmentation_operator.segmentedImage / SegmentationEnhancement_opera
tor.segmentedImage
  SegmentationEnhancement_operator.enhancedImage /RegionSelection_Operator.enhan
cedImage
  RegionSelection_Operator.selectedImage/BoundaryExtraction_Operator.selectedIma
ge

}




#========================================================
# Composite Operator : ImageColorSegmentation_operator
#========================================================

Composite Operator {
         name  ImageColorSegmentation_operator


Input Data
        Image name  inputImage
        Image name maskImage
        VisualContentContext name inputContext
        Symbol name segmentationType
        range ["Robust" "Approximative"]
        default "Robust"


Output Data
        Image name  segmentedImage




Preconditions
        valid inputImage

Postconditions
        valid segmentedImage



Body
 KMeans_operator | MeanShift_operator | RegionGrowing_Operator
```



```

Choice criteria
Rule{
        name ChoiceMethod1
        If segmentationType=="Robust"
        Then
        use_operator MeanShift_operator
}

Rule{
        name ChoiceMethod2
        If segmentationType=="Approximative"
        Then
        use_operator KMeans_operator
}



Rule{
        name ChoiceMethod3
        Let c a Context
        If c.backgroundAppearance == "Homogeneous"
        Then
        use_operator RegionGrowing_Operator

}




Distribution
        ImageColorSegmentation_operator.inputImage/ KMeans_operator.inputImage
        ImageColorSegmentation_operator.maskImage/ KMeans_operator.maskImage
        ImageColorSegmentation_operator.segmentedImage/ KMeans_operator.segmente
dImage
        ImageColorSegmentation_operator.inputImage/ MeanShift_operator.inputImag
e
        ImageColorSegmentation_operator.maskImage/MeanShift_operator.maskImage
        ImageColorSegmentation_operator.segmentedImage/MeanShift_operator.segmen
tedImage
        ImageColorSegmentation_operator.inputImage/ RegionGrowing_operator.input
Image
        ImageColorSegmentation_operator.maskImage/ RegionGrowing_operator.maskIm
age
        ImageColorSegmentation_operator.segmentedImage/ RegionGrowing_operator.s
egmentedImage
}















#========================================================
```

```
# Primitive Operator : MeanShift_operator
#=========================================================



Primitive Operator{
        name MeanShift_operator


Input Data
        Image name  inputImage
        Image name  maskImage
        VisualContentContext name inputContext


Input Parameters

        Float name maxNeighbourColorDistance
        comment  "Color difference to joined pixels"
        default 3


        Symbol name option
        range ["Quantization" "OverSegmentation" "UnderSegmentation"]
        default "UnderSegmentation"

        Integer name minRegionSize
        default 15


Output Data
        Image name  segmentedImage

        I−O Relations
        segmentedImage.path := inputImage.path,
        segmentedImage.basename := "regions",
        segmentedImage.extension := ".pan"


Preconditions
        valid inputImage

Postconditions
        valid segmentedImage




Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/MeanShif
tSegmentation  inputImage.get_filename() segmentedImage.get_filename() −o option
 −color
 maxNeighbourColorDistance −s minRegionSize   −m maskImage
        endsyntax
        program name MeanShiftSegmentation
        type  real

}
```



```


#=========================================================
# Primitive Operator : KMeans_operator
#=========================================================



Primitive Operator{
        name MeanShift_operator


Input Data
        Image name  inputImage
        Image name  maskImage
        VisualContentContext name inputContext


Input Parameters

        Symbol name filterType
        range ["Nothing" "Median" "KNearest"]
        default "KNearest"



Output Data
        Image name  segmentedImage

        I−O Relations
        segmentedImage.path := inputImage.path,
        segmentedImage.basename := "regions",
        segmentedImage.extension := ".pan"


Preconditions
        valid inputImage

Postconditions
        valid segmentedImage




Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/KMeansSe
gmentation filterType inputImage.get_filename() segmentedImage.get_filename() −
m maskImage
        endsyntax
        program name KMeansSegmentation
        type  real

}




#=========================================================
```

```
# Primitive Operator : RegionGrowing_Operator
#=========================================================



Primitive Operator{
        name RegionGrowing_Operator


Input Data
        Image name  inputImage
        Image name  maskImage
        VisualContentContext name inputContext




Output Data
        Image name  segmentedImage

        I−O Relations
        segmentedImage.path := inputImage.path,
        segmentedImage.basename := "regions",
        segmentedImage.extension := ".pan"


Preconditions
        valid inputImage

Postconditions
        valid segmentedImage




Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/RegionGr
owingSegmentation inputImage.get_filename() segmentedImage.get_filename() − m ma
skImage
        endsyntax
        program name RegionGrowingSegmentation
        type  real

}






#=========================================================
# Primitive Operator :SegmentationEnhancement_operator
#=========================================================



Primitive Operator{
        name SegmentationEnhancement_operator
```



```
Input Data
        Image name  segmentedImage


Output Data
        Image name  enhancedImage

        I−O Relations
        enhancedImage.path := segmentedImage.path,
        segmentedImage.basename := "regionsenhanced",
        segmentedImage.extension := ".pan"


Preconditions
        valid segmentedImage

Postconditions
        valid enhancedImage




Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/Enhancem
ent segmentedImage.get_filename() enhancedImage.get_filename()
        endsyntax
        program name RegionSegmentationEnhancement
        type  real

}




#=========================================================
# Primitive Operator :RegionSelection_operator
#=========================================================



Primitive Operator{
        name RegionSelection_operator


Input Data
        Image name  enhancedImage


Input Parameters

        Symbol name selection_type
                range[ "size" "color"]
                default "size"
        Float name selection_value




Output Data
        Image name  selectedImage
```

```
        I−O Relations
        enhancedImage.path := segmentedImage.path,
        segmentedImage.basename := "regionsenhanced",
        segmentedImage.extension := ".pan"


Preconditions
        valid enhancedImage

Postconditions
        valid selectedImage

Initialisation Criteria
Rule {
        name  init_selectionvalue
        Let c a Visual Content Context
        comment  "What curvilinear feature"
                If selection_type == "size"
                Then
                selection_value:=  inputContext.minobjectSize
        }


Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/RegionSe
lection selection_type selection_value enhancedImage.get_filename() selectedImag
e.get_filename()
        endsyntax
        program name RegionSelection
        type  real

}





#=========================================================
# Primitive Operator :BoundaryExtraction_operator
#=========================================================



Primitive Operator{
        name BoundaryExtraction_operator


Input Data
        Image name  selectedImage


Output Data
        Image name  outputImage




Preconditions
```



```
        valid selectedImage

Postconditions
        valid outputImage




Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/Frontier
e 8 selectedImage.get_filename() outputImage.get_filename()
        endsyntax
        program name Frontiere
        type  real

}









#=========================================================
# Composite Operator : Threshold_operator
#=========================================================

Composite Operator {
         name  Threshold_operator
         comment  "Detection of objects in images by thresholding"


Input Data
        Image name  inputImage
        Image name maskImage
        VisualContentContext name inputContext


Output Data
        MyImage name  segmentedImage

Preconditions
        valid inputImage

Postconditions
        valid segmentedImage





Body
 BITonalThresholding_operator | AutomaticThresholding_operator

Distribution
        Threshold_operator.inputImage/ BITonalThresholding_operator.inputImage
        Threshold_operator.maskImage/ BITonalThresholding_operator.maskImage
        Threshold_operator.segmentedImage/ BITonalThresholding_operator.segmente
dImage
```

```
        Threshold_operator.inputImage/ AutomaticThresholding_operator.inputImage
        Threshold_operator.maskImage/ AutomaticThresholding_operator.maskImage
        Threshold_operator.segmentedImage/ AutomaticThresholding_operator.segmen
tedImage



}







#=========================================================
# Primitive Operator : BiTonalThresholding_operator
#=========================================================




Primitive Operator{
        name BiTonalThresholding_operator
        comment  "Seuillage d’une image entre deux valeurs d’intensite "



Input Data
        Image name  inputImage
        VisualContentContext name inputContext
        Image name maskImage

Input Parameters

        Integer name  seuilA
        comment  "Threshold 1"
        default  0


        Integer name  seuilB
        comment  "Threshold 1"
        default  0

Output Data
        MyImage name  segmentedImage

        I−O Relations
        segmentedImage.path := inputImage.path,
        segmentedImage.basename := "regions",
        segmentedImage.extension := ".pan"


Preconditions
        valid inputImage

Postconditions
        valid segmentedImage
```



```
Initialisation criteria
Rule {
        name  init_seuilA_bright
                Let c a Context
                If c.objectLuminosity=="bright"
                Then
                seuilA:=125, seuilsB:=255
        }

Rule {
        name  init_seuilA_dark
                Let c a Context
                If c.objectLuminosity=="dark"
                Then
                seuilA:=0, seuilsB:=125
        }

Rule {
        name  init_seuilA_medium
                Let c a Context
                If c.objectLuminosity=="dark"
                Then
                seuilA:=75, seuilsB:=200
        }


Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/Threshol
d seuilA seuilB  inputImage.get_filename() segmentedImage.get_filename() −m mask
Image
        endsyntax
        program name Threshold
        type  real

}





#=========================================================
# Primitive Operator : BiTonalThresholding_operator
#=========================================================




Primitive Operator{
        name AutomaticThresholding_operator
        comment  "Seuillage automatique d’une image entre deux valeurs d’intensi
te "




Input Data
        Image name  inputImage
        VisualContentContext name inputContext
        Image name maskImage

Output Data
        MyImage name  segmentedImage
```

```
        I−O Relations
        segmentedImage.path := inputImage.path,
        segmentedImage.basename := "regions",
        segmentedImage.extension := ".pan"


Preconditions
        valid inputImage

Postconditions
        valid segmentedImage




Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/Automati
cThreshold inputImage.get_filename() segmentedImage.get_filename() −m maskImage
        endsyntax
        program name AutomaticThresholding
        type  real

}
```



```
#=========================================================
# Primitive Operator : Edge_segmentation_operator
#=========================================================

Primitive Operator {
         name  Edge_segmentation_operator
         comment  "Detection of objects in images by edge detection : to complet
e in composite operator"


Input Data
        Image name  inputImage
        Image name maskImage
        VisualContentContext name inputContext


Output Data
        MyImage name  segmentedImage

Preconditions
        valid inputImage

Postconditions
        valid segmentedImage


Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/application/script/EdgeDete
ction sobel inputImage.get_filename() segmentedImage.get_filename() −m maskImage
        endsyntax
```



```
        program name EdgeDetection
        type  real




}
```

Printed by Celine Hudelot

Feb 24, 05 22:37 **t−FeatureExtraction.yakl** Page 1/13

```
#================================================================
# Celine Hudelot−2003−2004
#================================================================
#================================================================
#Base de Connaissances "Generic Image Processing"
#Sous partie FeatureExtraction
#Pilotage de programmes de Traitement d'Images
#Operateurs de Feature Extraction
#================================================================
#================================================================
# Terminal Operators
#================================================================


Argument Type{
        name ContextFeature
        Attributes
        File name contextFile
        String name primitiveType
                        range ["Line" "Region" "Edge"]
        String name featureType
                        range ["Size" "Position" "Shape"]


Methods void loadContextFeature()

}


Argument Instance {
        Image name my_image
        Attributes
        path := "/user/chudelot/home/NOSAVE/ImageExecution/"
        basename := "carteregion"
        extension := ".pan"




}



Argument Instance{
        File name context_file
        Attributes
        path := "/user/chudelot/home/NOSAVE/WorkDirectory/ContextFeatureFile/"
        basename := "ContextFeatureFile"
        extension := ".dat"

}







Argument Instance{
        ContextFeature name context
Attributes
```

Feb 24, 05 22:37 **t−FeatureExtraction.yakl** Page 2/13

```
        contextFile:= context_file



}









#======================================================
# Functionality : FeatureExtraction
#======================================================

Functionality{
        name FeatureExtraction
        comment "Extraction of image data features "
        Achieved by Feature_Extraction_operator
        Input Data
                Image name inputImage

                ContextFeature name inputContext

        Output Data
                File name featureFile
}



#============================================
# Primitive Operator : LineShapeFeatureExraction
#============================================

Primitive Operator{
        name  Line_Shape_Feature_Extraction


Input Data
        Image name inputImage

Output Data
        File  name  featureFile

I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "LineShape",
         featureFile.extension := ".pan"
Preconditions
        valid inputImage

Postconditions
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_lineShapeDescrip
tors Shape inputImage.get_filename() /user/chudelot/home/NOSAVE/WorkDirectory/In
```

```
putImage/ridge.pan featureFile.get_filename()
        endsyntax
        program name  calcul_lineShapeDescriptors
        type  real


}

#===========================================
# Primitive Operator : LinePositionFeatureExraction
#===========================================

Primitive Operator{
        name  Line_Position_Feature_Extraction


Input Data
        Image name inputImage

Output Data
        File  name  featureFile
I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "LinePosition",
         featureFile.extension := ".pan"

Preconditions
        valid inputImage

Postconditions
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_linePositionDesc
riptors Position inputImage.get_filename() featureFile.get_filename()
        endsyntax
        program name  calcul_linePositionDescriptors
        type  real


}
#===========================================
# Primitive Operator : LineSizeFeatureExraction
#===========================================

Primitive Operator{
        name  Line_Size_Feature_Extraction


Input Data
        Image name inputImage

Output Data
        File  name  featureFile
I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "LineSize",
         featureFile.extension := ".pan"
```



```


Preconditions
        valid inputImage

Postconditions
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_lineSizeDescript
ors Size inputImage.get_filename() featureFile.get_filename()
        endsyntax
        program name  calcul_lineSizeDescriptors
        type  real


}





#===========================================
# Primitive Operator : EdgeShapeFeatureExraction
#===========================================

Primitive Operator{
        name  Edge_Shape_Feature_Extraction


Input Data
        Image name inputImage

Output Data
        File  name  featureFile
I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "EdgeShape",
         featureFile.extension := ".pan"


Preconditions
        valid inputImage

Postconditions
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_lineShapeDescrip
tors Shape inputImage.get_filename()  featureFile.get_filename()
        endsyntax
        program name  calcul_lineShapeDescriptors
        type  real
```

```
}

#===========================================
# Primitive Operator : EdgePositionFeatureExraction
#===========================================

Primitive Operator{
        name  Edge_Position_Feature_Extraction


Input Data
        Image name inputImage

Output Data
        File  name  featureFile
I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "EdgePosition",
         featureFile.extension := ".pan"

Preconditions
        valid inputImage

Postconditions
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_linePositionDesc
riptors Position inputImage.get_filename() featureFile.get_filename()
        endsyntax
        program name  calcul_linePositionDescriptors
        type  real


}
#===========================================
# Primitive Operator : EdgeSizeFeatureExraction
#===========================================

Primitive Operator{
        name  Edge_Size_Feature_Extraction


Input Data
        Image name inputImage

Output Data
        File  name  featureFile
I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "EdgeSize",
         featureFile.extension := ".pan"

Preconditions
        valid inputImage

Postconditions
```



```
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_lineSizeDescript
ors Size inputImage.get_filename() featureFile.get_filename()
        endsyntax
        program name  calcul_lineSizeDescriptors
        type  real


}


#===========================================
# Primitive Operator : RegionShapeFeatureExraction
#===========================================

Primitive Operator{
        name  Region_Shape_Feature_Extraction


Input Data
        Image name inputImage

Output Data
        File  name  featureFile
I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "RegionShape",
         featureFile.extension := ".pan"

Preconditions
        valid inputImage

Postconditions
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_RegionShapeDescr
iptors Shape inputImage.get_filename() featureFile.get_filename()
        endsyntax
        program name  calcul_lineShapeDescriptors
        type  real


}

#===========================================
# Primitive Operator : RegionPositionFeatureExraction
#===========================================

Primitive Operator{
        name  Region_Position_Feature_Extraction
```

Printed by Celine Hudelot


```
Input Data
        Image name inputImage

Output Data
        File  name  featureFile
I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "RegionPosition",
         featureFile.extension := ".pan"

Preconditions
        valid inputImage

Postconditions
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_RegionPositionDe
scriptors Position inputImage.get_filename() featureFile.get_filename()
        endsyntax
        program name  calcul_linePositionDescriptors
        type  real


}
#============================================
# Primitive Operator : RegionSizeFeatureExraction
#============================================

Primitive Operator{
        name  Region_Size_Feature_Extraction


Input Data
        Image name inputImage

Output Data
        File  name  featureFile
I−O relations
     featureFile.path := "/user/chudelot/home/NOSAVE/fichiers/",
         featureFile.basename := inputImage.basename + "RegionSize",
         featureFile.extension := ".pan"

Preconditions
        valid inputImage

Postconditions
        valid featureFile



Call

        language  shell
        syntax  /user/chudelot/home/pandore/pandore4/bin/calcul_RegionSizeDescri
ptors Size inputImage.get_filename() featureFile.get_filename()
        endsyntax
```



```
        program name  calcul_lineSizeDescriptors
        type  real


}


Composite Operator {
         name  Line_Feature_Extraction_operator



Input Data
         Image name  inputImage
         ContextFeature name inputContext



Output Data
        File name featureFile




Preconditions
        valid inputImage

Postconditions
        valid featureFile






Body
Line_Size_Feature_Extraction | Line_Position_Feature_Extraction | Line_Shape_Fea
ture_Extraction
```

```
Choice criteria
Rule{
        name ChoiceLineFeature1
        If inputContext.featureType == "Size"
        Then
        use_operator Line_Size_Feature_Extraction
}


Rule{
        name ChoiceLineFeature2
        If inputContext.featureType == "Shape"
        Then
        use_operator Line_Shape_Feature_Extraction
}


Rule{
        name ChoiceLineFeature3
        If inputContext.featureType == "Position"
        Then
        use_operator Line_Position_Feature_Extraction
}

Distribution
Line_Feature_Extraction_operator.inputImage / Line_Size_Feature_Extraction.input
Image
Line_Feature_Extraction_operator.inputImage / Line_Shape_Feature_Extraction.inpu
tImage
Line_Feature_Extraction_operator.inputImage / Line_Position_Feature_Extraction.i
nputImage

}




Composite Operator {
         name  Edge_Feature_Extraction_operator



Input Data
         Image name  inputImage
         ContextFeature name inputContext



Output Data
        File name featureFile



Preconditions
        valid inputImage

Postconditions
        valid featureFile
```



```
Body
Edge_Size_Feature_Extraction | Edge_Position_Feature_Extraction | Edge_Shape_Fea
ture_Extraction

Choice criteria
Rule{
        name ChoiceEdgeFeature1
        If inputContext.featureType == "Size"
        Then
        use_operator Edge_Size_Feature_Extraction
}


Rule{
        name ChoiceEdgeFeature2
        If inputContext.featureType == "Shape"
        Then
        use_operator Edge_Shape_Feature_Extraction
}



Rule{
        name ChoiceEdgeFeature3
        If inputContext.featureType == "Position"
        Then
        use_operator Edge_Position_Feature_Extraction
}

Distribution
Edge_Feature_Extraction_operator.inputImage / Edge_Size_Feature_Extraction.input
Image
Edge_Feature_Extraction_operator.inputImage / Edge_Shape_Feature_Extraction.inpu
tImage
Edge_Feature_Extraction_operator.inputImage / Edge_Position_Feature_Extraction.i
nputImage

}




Composite Operator {
         name  Region_Feature_Extraction_operator



Input Data
         Image name  inputImage
         ContextFeature name inputContext



Output Data
        File name featureFile



Preconditions
        valid inputImage

Postconditions
```

Printed by Celine Hudelot


```
        valid featureFile




Body
Region_Size_Feature_Extraction | Region_Position_Feature_Extraction | Region_Sha
pe_Feature_Extraction


Choice criteria
Rule{
        name ChoiceRegionFeature1
        If inputContext.featureType == "Size"
        Then
        use_operator Region_Size_Feature_Extraction
}


Rule{
        name ChoiceRegionFeature2
        If inputContext.featureType == "Shape"
        Then
        use_operator Region_Shape_Feature_Extraction
}


Rule{
        name ChoiceRegionFeature3
        If inputContext.featureType == "Position"
        Then
        use_operator Region_Position_Feature_Extraction
}

Distribution
Region_Feature_Extraction_operator.inputImage / Region_Size_Feature_Extraction.i
nputImage
Region_Feature_Extraction_operator.inputImage / Region_Shape_Feature_Extraction.
inputImage
Region_Feature_Extraction_operator.inputImage / Region_Position_Feature_Extracti
on.inputImage



}




Composite Operator {
         name  Feature_Extraction_operator


Functionality
        FeatureExtraction

Input Data
         Image name  inputImage
         ContextFeature name inputContext
```



```

Output Data
        File name featureFile



Preconditions
        valid inputImage

Postconditions
        valid featureFile

Initialization criteria
Rule {
        name  init_context1
        comment  "Initialization of the context with the file"
                If      true
                Then
                inputContext.loadContextFeature()
        }

Body
Region_Feature_Extraction_operator | Line_Feature_Extraction_operator | Edge_Fea
ture_Extraction_operator


Choice criteria
Rule{
        name ChoiceFeature1
        If inputContext.primitiveType == "Line"
        Then
        use_operator Line_Feature_Extraction_operator
}

Rule{
        name ChoiceSegmentation2
        if inputContext.primitiveType == "Region"
        Then
        use_operator Region_Feature_Extraction_operator

}
Rule{
        name ChoiceSegmentation3
        if inputContext.primitiveType == "Edge"
        Then
        use_operator Edge_Feature_Extraction_operator

}


Distribution
  Feature_Extraction_operator.inputImage/ Line_Feature_Extraction_operator.input
Image
  Feature_Extraction_operator.inputContext/ Line_Feature_Extraction_operator.inp
utContext
  Feature_Extraction_operator.inputImage/ Region_Feature_Extraction_operator.inp
utImage
  Feature_Extraction_operator.inputContext/ Region_Feature_Extraction_operator.i
nputContext
  Feature_Extraction_operator.inputImage/ Edge_Feature_Extraction_operator.input
Image
```

Bibliography

[cvg, 1994] (1994). *CVGIP: Image Understanding*, 60(1):65–118.

[Aamodt and Plaza, 1994] Aamodt, A. and Plaza, E. (1994). Case-based reasoning: Foundational issues, methodological variations, and system approaches. *AI Communications*, 7:39–59.

[Aloimonos, 1990] Aloimonos, J. (1990). Purposive and qualitative active vision. *IEEE International Conference on Pattern Recognition*, 1:346–360.

[Aloimonos et al., 1987] Aloimonos, J., Weiss, I., and Bandopadhay, A. (1987). Active vision. *International Journal On Computer Vision*, 1(4):333–356.

[Asher and Vieu, 1995] Asher, N. and Vieu, L. (1995). Toward a Geometry of Common Sense: a Semantics and a Complete Axiomatization of Mereotopology. In *IJCAI*, pages 846–852, San Mateo, Canada.

[Bachimont, 2000] Bachimont, B. (2000). *Engagement semantique et engagement ontologique : conception et realisation d'ontologie en ingenierie des connaissances dans* ***Ingeniere des connaissances, evolutions recentes et nouveaux defis***. Eyrolles.

[Bajcsy, 1988] Bajcsy, R. (1988). Active perception. *Proceedings of IEEE*, 76(8):996–1005.

[Ballard, 1992] Ballard, D. (1992). Principles of animate vision. *CVGIP: Image Understanding*, 56(1):3–21.

[Benjamins et al., 1999] Benjamins, V., Chandrasekaran, B., Perez, A. G., and Guarino, N. (1999). Ontologies and problem-solving methods: Lessons learned and future trends. In *Proceedings of the IJCAI-99 Workshop*.

[Benyon et al., 1999] Benyon, F., Jones, A., Tovey, E., and Stone, G. (1999). Differenciation of allergenic fungal spores by image analysis, with application to aerobiological counts. *Aerobiologica*, 15:211–223.

[Bernier and Landry, 2000] Bernier, T. and Landry, J. A. (2000). Algorithmic recognition of biological objects. *Canadian Agricultural Engineering*, 42(2):101–109.

[Bhushan et al., 1997] Bhushan, N., Rao, A. R., and Lohse, G. L. (1997). The texture lexicon: Understanding the categorization of visual texture terms and their relationship to texture images. *Cognitive Science*, 21(2):219–246.

[Biederman, 1987] Biederman, I. (1987). Recognition-by-components : A theory of human image understanding. *Psychological Review*, 94(2):115–147.

[Bloch et al., 2003] Bloch, I., Géraud, T., and Maître, H. (2003). Representation and fusion of heterogeneous fuzzy information in the 3d space for model-based structural recognition; application to 3d brain imaging. *Artificial Intelligence*, 148:141–175.

[Bloch and Ralescu, 2003] Bloch, I. and Ralescu, A. (2003). Directional relative position between objects in image processing : a comparaison between fuzzy approaches. *Pattern Recognition*, 36:1563–1582.

[Bloch and Saffiotti, 2004] Bloch, I. and Saffiotti, A. (2004). Some similarities between anchoring and pattern recognition concepts. In *AAAI Fall Symposium on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*.

[Borst, 1997] Borst, W. (1997). *Construction of Engineering Ontologies for Knowledge Sharing and Reuse*. Dutch graduate school for information and knowledge systems, University of Twente, Enschede.

[Bredeche, 2002] Bredeche, N. (2002). *Ancrage de lexique et perceptions : changements de représentation et apprentissage dans le cadre d'un agent situé et mobile*. Informatique, Université Paris XI.

[Brolio et al., 1989] Brolio, J., Draper, B., Beveridge, J., and Hanson, A. (1989). Isr: A database for symbolic processing in computer vision. *Computer*, 22(12):22–30.

[Buxton and Gong, 1995] Buxton, H. and Gong, S. (1995). Visual surveillance in a dynamic and uncertain world. *Artificial Intelligence*, 78:431–459.

[Camara et al., 2001] Camara, G., Egenhofer, M., Fonseca, F., and Montero, A. (2001). What's in an image? In Montello, D., editor, *Spatial information Theory: Fundations of geographic Information Science, International Conference, COSIT*, Lecture Notes in Computer Science, 2205, Morro bay, CA. Springer-Verlag.

[Chella et al., 1997] Chella, A., Frixione, M., and Gaglio, S. (1997). A cognitive architecture for artificial vision. *Artificial Intelligence*, 89:73–111.

[Chella et al., 2002] Chella, A., Frixione, M., and Gaglio, S. (2002). Anchoring symbols to conceptual spaces: the case of dynamic scenarios. *Robotics and Autonomous Systems, special issue on Perceptual Anchoring*, 89.

[Clarke, 1981] Clarke, B. L. (1981). A calculus of individuals based on "connection". *Notre Dame Journal of Formal Logic*, 22(3):204–218.

[Clement and Thonnat, 1993a] Clement, V. and Thonnat, M. (1993a). A knowledge-based approach to integration of image procedures processing. *CVGIP: Image Understanding*, 57(2):166–184.

[Clement and Thonnat, 1993b] Clement, V. and Thonnat, M. (1993b). A knowledge-based approach to integration of image procedures processing. *CVGIP: Image Understanding*, 57(2):166–184.

[Clementini and Felice, 1997] Clementini, E. and Felice, P. D. (1997). A global framework for qualitative shape description. *Geoinformatica*, 1(1):11–27.

[Clementini and Felice, 1998] Clementini, E. and Felice, P. D. (1998). Topological invariants for lines. *IEEE Transactions on Knowledge and Data Engineering*, 10(1):38–54.

[Clementini et al., 1997] Clementini, E., Felice, P. D., and Hernandez, D. (1997). Qualitative representation of positional information. *Artif. Intell.*, 95(2):317–356.

[Clouard et al., 1999] Clouard, R., Elmoataz, A., Porquet, C., and Revenu, M. (1999). Borg: A knowledge-based system for automatic generation of image processing programs. *IEEE Trans. Pattern Anal. Mach. Intell.*, 21:128–144.

[Cohn and Hazarika, 2001] Cohn, A. G. and Hazarika, S. M. (2001). Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29.

[Cohn et al., 1994] Cohn, A. G., Randell, D. A., and Cui, Z. (1994). Taxonomies of logically defined qualitative spatial relations. *Journal of Human-Computer Studies*.

[Cohn A.G, 2002] Cohn A.G, H. S. (2002). Qualitative spatial representation and reasoning: An overview. *Fundamenta Informaticae*, 46(1-2):1–29.

[Colliot et al., 2004] Colliot, O., Camara, O., Dewynter, R., and Bloch, I. (2004). Description of brain internal structures by means of spatial relations for mr image segmentation. In *Proceeding of the International Society of Optical Engineering*, volume 5370 of *SPIE 2004 Medical Imaging*, pages 444–455, San Diego. USA.

[Coradeschi, 1999] Coradeschi, S. (1999). *Anchoring symbols to sensory data.* Linköping studies in science and technology, Linköping, Sweden.

[Coradeschi et al., 2001] Coradeschi, S., Driankov, D., Karlsson, L., and Saffiotti, A. (2001). Fuzzy anchoring. In *Proc. of the IEEE Intl. Conf. on Fuzzy System*, pages 111–114.

[Coradeschi and Saffiotti, 2003] Coradeschi, S. and Saffiotti, A. (2003). An introduction to the anchoring problem. *Preprint submitted to Elsevier Science.*

[Crevier and Lepage, 1997] Crevier, D. and Lepage, R. (1997). Knowledge-based image understanding systems: a survey. *Comput. Vis. Image Underst.*, 67(2):160–185.

[Crubézy et al., 1997] Crubézy, M., Aubry, F., Moisan, S., Chameroy, V., Thonnat, M., and Di Paola, R. (1997). Managing Complex Processing of Medical Image Sequences by Program Supervision Techniques. In *SPIE Medical Imaging'97*, volume 3035, pages 614–625.

[Devedzić, 2002] Devedzić, V. (2002). Understanding ontological engineering. *Commun. ACM*, 45(4):136–144.

[Draper, 2003] Draper, B. (2003). From knowledge bases to markov models to pca. In *Invited talk for the Workshop on Computer Vision System Control Architectures.*

[Draper et al., 2000] Draper, B., Bins, J., and Baek, K. (2000). Adore: Adaptive object recognition. *Videre*, 1(4):86–99.

[Draper et al., 1989] Draper, B., Collins, R., Brolio, J., Hanson, A., and Riseman, E. (1989). The schema system. *IJCV*, 2(3):209–250.

[Draper and Hanson, 1991] Draper, B. and Hanson, A. (1991). An example of learning in knowledge-directed vision. In *SCIA91*, pages 189–201.

[Draper B., 1996] Draper B., Hanson A.R, R. E. (1996). Knowledge-directed vision: control, learning and integration. *Proceedings of IEEE*, 84(11):1625–1681.

[E. Clementini and Hernandez, 1997] E. Clementini, P. F. and Hernandez, D. (1997). Qualitative representation of positional information. *Artificial Intelligence*, 85:317–356.

[Egenhofer, 1993] Egenhofer, M. (1993). A model for detailed binary topological relationships. *Geomatica*, 47(3-4):261–273.

[Egenhofer and Sharma, 1993] Egenhofer, M. and Sharma, J. (1993). Relations between regions in r2 and z2. In Abel, D. and Ooi, B., editors, *Advances in Spatial Databases*, Singapore. Springer Verlag.

[Ehret et al., 2001] Ehret, D. L., Lau, A., Bittman, S., Lin, W., and Shelford, T. (2001). Automating monitoring of greenhouse crops. *Agronomie*, 21:403–414.

[El-Azhary et al., 2000] El-Azhary, E., Hassan, H., and Rafea, A. (2000). Pest control xpert system for tomato (pcest). *Knowledge and Information Systems*, 2:242–257.

[El-Faki et al., 2000] El-Faki, M., Zhang, N., and Peterson, D. E. (2000). Weed detection using color machine vision. *Transactions on the ASAE*, 43(6):1969–1978.

[El-Helly et al., 2004] El-Helly, M., El-Beltagy, S., and Rafea, A. (2004). Image analysis based interface for diagnostic expert systems. In *WISICT '04: Proceedings of the winter international synposium on Information and communication technologies*, pages 1–6. Trinity College Dublin.

[Engbers and Smeulders, 2003] Engbers, E. A. and Smeulders, A. W. (2003). Design considerations for generic grouping in vision. *IEEE Transactions On Pattern Analysis And Machine Intelligence*, 25(4):445–457.

[Ficet-Cauchard et al., 1999] Ficet-Cauchard, V., Porquet, C., and Revenu, M. (1999). Cbr for the management and reuse of image-processing expertise: a conversational system. *Engineering Applications of Artificial Intelligence*, 12(6):733–747.

[Fischer et al., 2000] Fischer, S., Binkert, M., and Bunke, H. (2000). Symmetry based indexing of diatoms in an image database. In *ICPR00*, pages Vol II: 895–898.

[Forsyth and Ponce, 2002] Forsyth, D. A. and Ponce, J. (2002). *Computer Vision: A Modern Approach*. Prentice Hall.

[Freeman, 1975] Freeman, J. (1975). The modelling of spatial relations. *Computer Graphics and Image Processing*, 4(2):156–171.

[Freksa et al., 2004] Freksa, C.and Knauff, M., Krieg-Bruckner, B., Nebel, B., and Barkowsky, T. (2004). *Spatial Cognition IV - Reasoning, Action, Interaction*. Springer Verlag.

[Furst et al., 2003] Furst, F., Leclere, M., and Trichet, F. (2003). Ontological engineering and mathematical knowledge management : a formalization of projective geometry. *Annals of Mathematics and Artificial Intelligence*.

[Gandon, 2002] Gandon, F. (2002). *Distributed Artificial Intelligence And Knowledge Management: Ontologies And Multi-Agent Systems For A Corporate Semantic Web*. Doctoral school of sciences and technologies of information and communication, University of Nice Sophia Antipolis.

[Gärdenfors, 2000] Gärdenfors, P. (2000). *Conceptual Spaces*. MIT Press.

[Gibson, 1979] Gibson, G. (1979). *The ecological approach to visual perception*. Houghton-Mifflin Co, Boston.

[Granger, 1985] Granger, C. (1985). *Reconnaissance d'objets par mise en correspondance en vision par ordinateur*. Robotique et vision, Université de Nice Sophia Antipolis.

[Grimnes and Aamodt, 1996] Grimnes, M. and Aamodt, A. (1996). A two layer case-based reasoning architecture for medical image understanding. In *Proceedings of 3rd European Workshop on Case-Based Reasoning EWCBR*.

[Gruber, 1993] Gruber, T. R. (1993). Towards Principles for the Design of Ontologies Used for Knowledge Sharing. In Guarino, N. and Poli, R., editors, *Formal Ontology in Conceptual Analysis and Knowledge Representation*, Deventer, The Netherlands. Kluwer Academic Publishers.

[Guyer et al., 1996] Guyer, D., Uthaisombut, P., and Stockman, G. (1996). Tissue reflectance and machine vision for automated sweet cherry sorting. In *Proc. SPIE Vol. 2907, p. 152-165, Optics in Agriculture, Forestry, and Biological Processing II, George E. Meyer; James A. DeShazer; Eds.*, pages 152–165.

[Hanson and Riseman, 78] Hanson, A. and Riseman, E. (78). Visions : A computer system for interpreting scenes. *Computer Vision Systems*.

[Hudelot, 2002] Hudelot, C. (2002). Etude de faisabilité d'un système d'acquisition d'images de végétaux pour la détection precoce de pathologies. Technical Report, INRA, INRIA.

[J. T. J. Srzednicki and Czelakowski, 1984] J. T. J. Srzednicki, V. F. R. and Czelakowski, J. (1984). *Lesniewski's Systems: Ontology and Mereology*. Ossolineum and Kluwer Academic Publishers.

[Jahns et al., 2001] Jahns, G., Nielsen, H. M., and Paul, W. (2001). Measuring image analysis attributes and modelling fuzzy consumer aspects for tomato quality grading. *Computers and Electronics in Agriculture*, 31:17–29.

[JunWei et al., 2000] JunWei, L., Guillemin, J. P., Xiaohui, Z., Sarrazin, P., ChenLin, M., and Coquille, J. C. (2000). A study for measuring the sunlit areas of onions and weeds in the fied by image processing. In *Proceedings of ICETS*, Beijing, China.

[Knauff et al., 1997] Knauff, M., Rauh, R., and Renz, J. (1997). A cognitive assessment of topological spatial relations : Results from an empirical investigation. In *Proceedings of the Third Conference on Spatial Information Theory*.

[Kossen, 1989] Kossen, N. (1989). The morphology. *IEEE Transactions on Knowledge and Data Engineering*, 1(1):89–100.

[Kuipers, 1996] Kuipers, B. (1996). A hierarchy of qualitative representations for space. In *Working Papers of the Tenth International Workshop on Qualitative Reasoning*, Fallen Leaf Lake, California.

[Lammens, 1994] Lammens, J. M. (1994). A computational model of color perception and color naming. Technical Report 94-26.

[Le Ber and Napoli, 2002] Le Ber, F. and Napoli, A. (2002). The design of an object-based system for representing and classifying spatial structures and relations. *Journal of Universal Computer Science*, 8(8):751–773.

[Leibe and Schiele, 2003] Leibe, B. and Schiele, B. (2003). Analyzing appearance and contour based methods for object categorization. In *Conference on Computer Vision and Pattern Recognition*.

[Levner et al., 2003] Levner, I., Bulitko, V., Lee, G., Li, L., and Greine, R. (2003). Automated creation of image interpretation systems. In *Proceedings of the 16th Australian Joint Conference on Artificial Intelligence*.

[Liedtke et al., 1997] Liedtke, C., Buckner, J., Grau, O., Growe, S., and Tonjes, R. (1997). Aida: A system for the knowledge based interpretation of remote sensing data. In *3rd International Airborne Remote Sensing Conference*.

[Mahaman et al., 2003] Mahaman, B., Passam, H., Sideridis, A., and Yialouris, C. (2003). Diares-ipm: a diagnostic advisory rule-based expert system for integrated pest management in solanaceous crop systems. *Agricultural Systems*, 76(3):1119–1135.

[Maillot et al., 2003a] Maillot, N., Thonnat, M., and Boucher, A. (2003a). Towards ontology based cognitive vision. In *Third International Conference on Computer Vision Systems*, Graz, Austria.

[Maillot et al., 2003b] Maillot, N., Thonnat, M., and Boucher, A. (2003b). A visual concept ontology for automatic image understanding. In *Poster Session of the Second International Semantic Web Conference*.

[Maillot et al., 2004a] Maillot, N., Thonnat, M., and Boucher, A. (2004a). Towards ontology based cognitive vision. *Machine Vision and Applications (MVA)*, 16(1):33–40.

[Maillot et al., 2004b] Maillot, N., Thonnat, M., and Hudelot, C. (2004b). Ontology based object learning and recognition : Application to image retrieval. In *Proceedings of 16th IEEE International Conference on Tools For Artificial Intelligence*, Boca Raton, USA. IEEE Computer Society Press.

[Mangelinck, 1998] Mangelinck, L. (1998). *Representation et classification de structures spatials. Application a la reconnaissance de paysages agricoles*. Loria, Nancy.

[Manh et al., 2001] Manh, A. G., Rabatel, G., and Aldon, M. J. (2001). Weed leaf image segmentation by deformable templates. *Journal of Agricultural Engineering Research*, 80(2):139–146.

[Mao and Bell, 1998] Mao, W. Z. and Bell, D. A. (1998). Integrating visual ontologies and wavelets for image content retrieval. In *DEXA Workshop*, pages 379–384.

[Marr, 1982] Marr, D. (1982). *VISION*. W. H. Freeman and Company, New York.

[Martin, 2004] Martin, V. (2004). Apprentissage supervise' pour la segmentation d'images. Rapport de DEA, Universite' de Nice Sophia Antipolis.

[Mathieu-Marni et al., 1995] Mathieu-Marni, S., Moisan, S., and Vincent, R. (1995). A knowledge-based system for the computation of land use mixing and the classification of a multi-spectral satellite scene. *International Journal of Remote Sensing*.

[Matsakis et al., 2001] Matsakis, P., Keller, J. M., Wendling, L., Marjamaa, J., and Sjahputera, O. (2001). Linguistic description of relative positions in images. *IEEE Transactions on Systems, Man, and Cybernetics - Part B : Cynernetics*, 31(4):573–588.

[Matsuyama and Hwang, 1990] Matsuyama, T. and Hwang, V. (1990). *SIGMA: A Knowledge-Based Aerial Image Understanding System.* Perseus Publishing.

[Meathrel, 2001] Meathrel, R. C. (2001). *A General theory of Boundary-Based Qualitative Representation of Two-Dimensional Shape.* Doctor of philosophy in computer science, University of Exeter.

[Mezaris et al., 2004] Mezaris, V., Kompatsiaris, I., and Strintzis, M. G. (2004). Region-based image retrieval using an object ontology and relevance feedback. *Eurasip Journal on Applied Signal Processing*, 2004(6):886–901.

[Mezzo et al., 2003] Mezzo, B. D., Rabatel, G., and Fioro, C. (2003). Weed leaf recognition in complex natural scenes by model-guided edge pairing. In *Proceedings of the European Conference on Precision Agriculture.*

[Minsky, 1974] Minsky, M. (1974). A framework for representing knowledge. Technical report.

[Moisan, 1998] Moisan, S. (1998). *Une plate-forme pour une programmation par composants de systèmes à base de connaissances.* Habilitation à diriger les recherches, Université de Nice.

[Moisan et al., 2001] Moisan, S., Ressouche, A., and Rigault, J.-P. (2001). BLOCKS, a Component Framework with Checking Facilities for Knowledge-Based Systems. *Informatica, Special Issue on Component Based Software Development*, 25(4):501–507.

[Moisan and Thonnat, 1995] Moisan, S. and Thonnat, M. (1995). Knowledge-based systems for program supervision. In *1st International Workshop on Knowledge Based systems for the (re)Use of Program Libraries*, pages 4–8.

[Moisan and Thonnat, 2000] Moisan, S. and Thonnat, M. (2000). What can program supervision do for program reuse. *IEE Proceedings - Software*, 147(5):179–185.

[Mojsilovic et al., 2002] Mojsilovic, A., Hu, J., and Soljanin, E. (2002). Extraction of perceptually important colors and similarity measurement for image matching retrieval and analysis. *IEEE Trans. on Image Processing*, 11(11):1238–1248.

[Munsell, 1923] Munsell, A. H. (1923). *A Color Notation.*

[Neumann and Weiss, 2003] Neumann, B. and Weiss, T. (2003). Navigating through logic-based scene models for high-level scene interpretations. In *3rd International Conference on Computer Vision Systems - ICVS*, pages 212–222.

[Niemann et al., 1990] Niemann, H., Sagerer, G. F., Schroder, S., and Kummert, F. (1990). Ernest: A semantic network system for pattern understanding. *IEEE Trans. Pattern Anal. Mach. Intell.*, 12(9):883–905.

[Nouvel and Dalle, 2001] Nouvel, A. and Dalle, P. (2001). Description des entités de traitement d'images pour la conception interactive d'applications. In *ORASIS.*

[Nunez et al., 1996] Nunez, F., Rodriguez, M., Cordoba, J., Bermudez, M. E., Asensio, M., Yialouris, C., and Sideridis, A. (1996). An expert system for tomato diseases. *Computers and Electronics in Agriculture*, 14(1):61–76.

[Ossola et al., 1996] Ossola, J., Brémond, F., and Thonnat, M. (1996). A communication level in a distributed architecture for object recognition. In *8th International Conference on Systems Research Informatics and Cybernetics*.

[Ossola and Thonnat, 1995a] Ossola, J. and Thonnat, M. (1995a). Cooperation of knowledge based systems for galaxy classification. In *8th International Conference on Image Analysis and Processing*.

[Ossola and Thonnat, 1995b] Ossola, J. and Thonnat, M. (1995b). A distributed architecture for automatic object recognition. In *Advances in Intelligent Data Analysis*.

[Ossola, 1996] Ossola, J.-C. (1996). *Coopération de systèmes à base de connaissances pour l'analyse et la reconnaissance d'objets naturels complexes: application au classement de galaxies ou de zooplanctons*. These d'informatique de l'école doctorale sciences de l'ingénieur, Université de Nice Sophia Antipolis.

[Papadias and Egenhofer, 1997] Papadias, D. and Egenhofer, M. (1997). Hierarchical reasoning about direction relations. *GeoInformatica*, 1(3):251–273.

[Paul and Thomas, 1998] Paul, G. C. and Thomas, C. R. (1998). Characterisation of mycelial morphology using image analysis. *Advances in Biochemical Engineering / Biotechnology*, 60.

[Peng and Bhanu, 1998] Peng, J. and Bhanu, B. (1998). Closed-loop object recognition using reinforcement learning. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20:139–154.

[Perez et al., 1997] Perez, A., Lopez, F., Benlloch, J., and Christensen, S. (1997). Colour and shape analysis techniques for weed detection in cereal fields. In *Proceedings of the First European Conference for Information Technology in Agriculture*, Copenhagen, Denmark.

[Perner, 2001] Perner, P. (2001). Why case-based reasoning is attractive for image interpretation. In *ICCBR*, pages 27–43.

[Perner et al., 2003a] Perner, P., Guenther, T., and Perner, H. (2003a). Airborne fungi identification by case-based reasoning. In *5th Intern. Conference on Case-Based Reasoning, Workshop Proceeding*, pages 73–79.

[Perner et al., 2003b] Perner, P., Gunther, T., and Perner, H. (2003b). Airbone fungi identification by case based reasoning. In *Worshop on CBR in the eath Science, The Fifth International Conference on Case Based Reasoning*.

[Pons and Vivier, 1999] Pons, M. N. and Vivier, H. (1999). Biomass quantification by image analysis. *Advances in Biochemical Engineering / Biotechnology*, 66.

[Renz and Nebel, 1999] Renz, J. and Nebel, B. (1999). On the complexity of qualitative spatial reasoning: A maximal tractable fragment of the region connection calculus. *Artificial Intelligence*, 108(1-2):69–123.

[Renz et al., 2000] Renz, J., Rauh, R., and Knauff, M. (2000). Towards cognitive adequacy of topological spatial relations. In *Spatial Cognition II, Integrating Abstract Theories, Empirical Studies, Formal Methods, and Practical Applications*, pages 184–197, London, UK. Springer-Verlag.

[Rimey and Brown, 1994] Rimey, R. D. and Brown, C. M. (1994). Control of selective perception using bayes nets and decision theory. *International Journal of Computer Vision*, 12(2):173–207.

[Robertson, 2000] Robertson, P. (2000). An architecture for self-adaptation and its application to aerial image understanding. In *IWSAS*, pages 199–223.

[Robertson and Laddaga, 2002] Robertson, P. and Laddaga, R. (2002). A self-adaptive architecture and its application to robust face identification. In *PRICAI*, pages 542–551.

[S. Christensen and Walter, 1996] S. Christensen, T. H. and Walter, M. (1996). Patch spraying in cereals. In *Proceedings of the Second International Weed Control Congress*, Copenhagen, Denmark.

[Sandakly, 1995] Sandakly, F. (1995). *Contribution à la mise en oeuvre d'une architecture à base de connaissances pour l'interprétation de scènes 2D et 3D*. These d'informatique. robotique et vision, Université de Nice Sophia Antipolis.

[Sandakly and Giraudon, 1994] Sandakly, F. and Giraudon, G. (1994). Scene analysis system. In *Proceedings of the International Conference of Image Processing*, volume 3, pages 806–810.

[Sandakly and Giraudon, 1995] Sandakly, F. and Giraudon, G. (1995). Reasoning strategies for 3d object detection. In *IEEE International Symposium on Computer Vision*.

[Schiele, 1997] Schiele, B. (1997). *ObjectRecognition using Mutidimensional Receptive Field Histograms*. PhD thesis, Institut National Polytechnique de Grenoble.

[Shahin and Symons, 2001] Shahin, M. and Symons, S. (2001). A machine vision system for grading lentils. *Canadian Biosystems Engineering*, 43.

[Shariff et al., 1998] Shariff, A. R., Egenhofer, M., and Mark, D. (1998). Natural-language spatial relations between linear and areal objects: The topology and metric of english-language terms. *International Journal of Geographical Information Science*, 12(3):215–246.

[Sistla and Yu, 2000] Sistla, A. P. and Yu, C. (2000). Reasoning about qualitative spatial relationships. *J. Autom. Reason.*, 25(4):291–328.

[Socher, 1997] Socher, G. (1997). *Qualitative Scene Descriptions from Images for Integrated Speech and Image Understanding*. University of bielefeld, Sankt Augustin.

[Steger, 1998] Steger, C. (1998). An unbiased detector of curvilinear structures. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(2).

[Steinmetz et al., 1994] Steinmetz, V., Delwiche, M. J., Giles, D. K., and Evans, R. (1994). Sorting cut roses with machine vision. *Transactions of the ASAE*, 37(4):1347–1353.

[Stocaj, 2003] Stocaj, D. (2003). *Robust subspace approaches to visual learning and recognition.* University of ljubljana, Faculty of Computer and Information Science,Ljubljana.

[Studer et al., 1998] Studer, R., Benjamins, V. R., and Fensel, D. (1998). Knowledge engineering: Principles and methods. *Data Knowledge Engineering*, 25(1-2):161–197.

[Studer et al., 1996] Studer, R., Eriksson, H., Gennari, J., Tu, S., Fensel, D., and Musen, M. (1996). Ontologies and the configuration of problem-solving methods. In *10th Banff Knowledge Acquisition for Knowledge-Based Systems Worksho.*

[T. Dorge and Frisvad, 2001] T. Dorge, J. C. and Frisvad, J. (2001). Direct identification of pure penicillium species using image analysis. *Journal of Microbiological Methods*, 41:121–133.

[Thonnat, 2002] Thonnat, M. (2002). Knowledge-based techniques for image processing and for image understanding. *J. Phys. IV France EDP Science, Les Ulis*, 12:189–236.

[Thonnat and Moisan, 1995] Thonnat, M. and Moisan, S. (1995). Knowlege-based systems for program supervision. In *Proceedings of the first international workshop on Knowledge-Based systems for the (re)Use of Program librairies KBUP*, Sophia Antipolis, France. INRIA.

[Thonnat et al., 1998a] Thonnat, M., Moisan, S., and Crubézy, M. (1998a). Experience in integrating image processing programs. In Henrik Christensen, editor, *Proceeding of the 1st International Conference on Vision Systems*, Lecture Notes in Computer Science, Las Palmas, Gran Canaria, Spain. Springer-Verlag.

[Thonnat et al., 1998b] Thonnat, M., Moisan, S., and Crubézy, M. (1998b). Experience in integrating image processing programs. In Henrik Christensen, editor, *Proceeding of the 1st International Conference on Vision Systems*, Lecture Notes in Computer Science, Las Palmas, Gran Canaria, Spain. Springer-Verlag.

[Tian et al., 2000] Tian, L., Slaughter, D. C., and Norris, R. (2000). Machine vision identification of tomato seedlings for automated weed control. *Unknown.*

[Town and Sinclair, 2004] Town, C. and Sinclair, D. (2004). Language-based querying of image collections on the basis of an extensible ontology. *Image Vision Comput.*, 22(3):251–267.

[Trivedi and Rosenfeld, 1989] Trivedi, M. and Rosenfeld, A. (1989). On making computers see. *SMC*, 19(6):1333–1335.

[Tsotsos, 1994] Tsotsos, J. (1994). There is no one way to look at vision. *CVGIP: Image Understanding*, 60(1):95–97.

[Unay and Gosselin, 2004] Unay, D. and Gosselin, B. (2004). A quality sorting method for jonalgold apple. In *Proc. of the Agricultural Engineering Conference.*

[Urena et al., 2001] Urena, R., Rodriguez, F., and Berenguel, M. (2001). A machine vision system for seed germination quality evaluation using fuzzy logic. *Computers and Electronics in Agriculture*, 32:1–20.

[Uschold and Grüninger, 1996] Uschold, M. and Grüninger, M. (1996). Ontologies: principles, methods, and applications. *Knowledge Engineering Review*, 11(2):93–155.

[Vernon, 2004] Vernon, D. (2004). Ecvision research roadmap v4.0. In *http://www.ecvision.org/*.

[Von-Wun et al., 2002] Von-Wun, S., Chen-Yu, L., Jium, Y. J., and Ching-Chich, C. (2002). Using sharable ontology to retrieve historical images. In *Proceedings of the second ACM/IEEE-CS Joint Conference on Digital Libraries*.

[Zhang and Lu, 2004] Zhang, D. and Lu, G. (2004). Review of shape representation and description techniques. *Pattern Recognition*, 37:1–19.

[Zhang and chaisattapagon, 1995] Zhang, N. and chaisattapagon, C. (1995). Effective criteria for weed identification in wheat fields using machine vision. *Transactions on the ASAE*, 38(3):965–974.

[Zlatoff, 2004] Zlatoff, N. (2004). Image understanding and scene models: a generic framework integrating domain knowledge and gestalt theory. In *Proceedings of International Conference on Image Processing*.