A communication level in a distributed architecture for object recognition

Jean-Christophe Ossola, François Brémond and Monique Thonnat
INRIA,
Sophia Antipolis, France

Keywords: Automatic recognition, image processing, data interpretation, distributed artificial intelligence, knowledge-based system.

Abstract

We present a communication level for a distributed architecture devoted to object recognition applications. The recognition process consists of two main stages: data processing for object description, and data interpretation for object recognition. Our architecture is based on the cooperation of two knowledge-based systems called Image-processor and Data-interpreter. This paper describes the main characteristics of a communication corpus used to make them communicate. The corpus allows us to define a generic communication level that can be adapted to a large class of object recognition applications. At the end of the paper we give a methodology to develop a new application by taking advantage of this level.

1 Introduction

We address the problem of the automatic recognition of natural and complex objects described by an image (Loonis, Zahzah and Bonnefoy, 1995). In a previous paper (Ossola and Thonnat, 1995) we have shown that the object recognition process can generally be decomposed into two main stages described by figure 1 :

- **image processing** which main goals are first to improve the quality of an image and then to extract numerical characteristics that describe one (or more) object(s) contained in the image. This process works on the pixels of input images. That corresponds to the **low level** part of the object recognition process.
- **data interpretation** takes as input the numerical characteristics extracted in the previous stage in order to assign a predefined class to the object(s). It infers symbolic data from numerical data and use them to find the class the object belongs to. That corresponds to the **high level** part of the object recognition process.

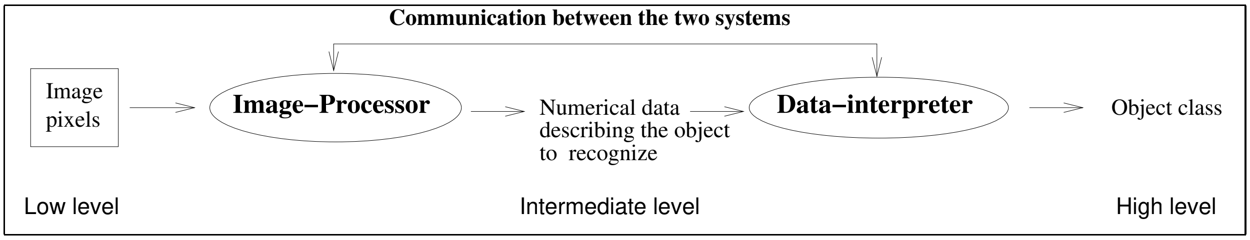

Figure 1: The two stages of an image recognition process

The **communication level** corresponds to the numerical data that are exchanged between the low and high level of abstraction (Meisels, 1991) (Dellepiane, Venturi and Vernazza, 1992). It defines the communication part of the object recognition process. This paper studies the different communication problems that appear when two systems belonging to separate knowledge domains need to exchange messages. These problems may be simple in the case of reactive systems like Brooks's robot society (Brooks, 1989). Indeed, they only react to stimuli they feel and then let their trace in the environment. In our case the global architecture is based on knowledge-based systems that solve problems in the same way experts proceed. The communication between two intelligent systems is much more complex due to the fact that they have their own knowledge domain. For object recognition, both image processing and interpretation domains need specific knowledge. Thus the goal of this paper is to present a communication level allowing such communications.

In section 2 we describe the two communicating systems and we explain the need for a communication level. Section 3 and 4 describe the communication corpus itself and show how it is used in an object recognition application. Before concluding, we show in section 5 how the communication level can be used in an application that has to recognize a moving object described by an image sequence.

2 The object recognition architecture : its need for a communication level

Our object recognition architecture is based on two knowledge-based systems called **image-processor** and **data-interpreter**. As indicated by their name they are dedicated respectively, to image processing and to data interpretation. Their development is facilitated by the use of two specialized KBS shells : OCAPI (Clément and Thonnat, 1993) and CLASSIC (Granger, 1985). Instead of chaining the global process, each system is able to send messages to the other at any step of its reasoning. For example this communication possibility allows the data-interpreter to improve its classification process by sending specific requests to the image-processor (e.g. to focus the image-processor attention). We have defined three types of messages :

1. at the beginning of the recognition process, the data-interpreter asks the image-processor to compute numerical data describing the object
2. during the reasoning, the data-interpreter needs new information on an image in order to compute specific object characteristics. It asks the image-processor for it.
3. during its reasoning, the image-processor may need an information. For example if the data-interpreter sends a request and forgets to provide contextual information, then the image-processor sends back a message asking for the missing information.

There are several reasons to use a communication level. The first one is due to the goal-directed nature of our image-processor architecture. It uses two structures *goal* and *operator* to represent its knowledge. A goal is an image processing functionality and an operator can either describe sequences of image processing treatments (complex operator) or programs of libraries (elementary operator). The utilization of the image-processor requires the name of one goal and also some additional parameters. In order to send messages of the first and second type to the image-processor, the data-interpreter needs to know the description of the goal to reach and the description of the required parameters.

In a symmetric way, for messages of the third type, the image-processor needs to access the base of facts of the data-interpreter (where all data are ranked with their values). If no common corpus is available between the two systems, it is impossible to access the good data. This is another point where the communication between these two systems can not be done without the definition of an adapted corpus.

3 Communication level

As suggested in the previous section, we define a communication corpus that is composed by a set of **basic terms common to both systems**. As for the image-processor, just a part of the system takes part in the communication process. In fact, the image-processor is be divided into an image processing library and a program-supervisor which has to dynamically build the plan of the treatment. The program-supervisor is a sub-system that schedules, executes and controls the results of the programs and exchanges messages with the data-interpreter. Thus the communication level is composed by the program-supervisor, the communication protocol and the corpus that embraces the common terms of the program-supervisor and the data-interpreter (see figure 2).

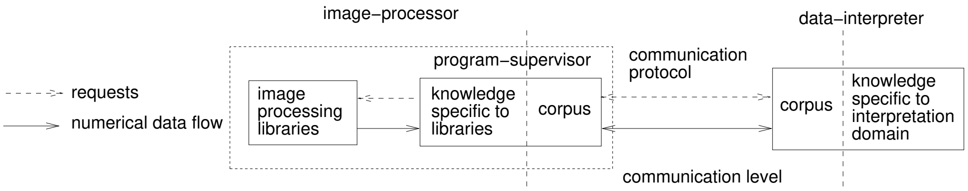

Figure 2: The scope of the communication level.

The corpus is used to formalize the relationships between the two systems. The meaning of the corpus terms belongs to two separate domains. First, the terms represent all the terminal symbols used to describe the knowledge on natural object classes of the data-interpreter. Here a terminal symbol means a word that describes a class and that is linked to an image property (e.g. a *contour*, a *barycenter* of a *contour*...). We can notice that the data-interpreter does not belong directly to the communication level because it only uses the corpus to send messages and not to perform its reasoning. Second, the terms express image processing functionalities (e.g. to *compute* the *barycenter* of the *contour*). The communication corpus is defined on one hand by abstracting the image processing functionalities and on a second hand by gathering the basic and generic terms used in several recognition systems we have developed (for galaxy (Thonnat, Clément and Ossola, 1995), foraminifera (Liu, Thonnat and Berthod, 1994) and zooplanktonic applications (Thonnat and Gandelin, 1988)). Thus the problem of making the two knowledge-based systems communicate comes down to link the image processing functionalities to the generic terms. For this reason we have defined a program-supervisor system which functionalities correspond explicitly to the generic interpretation terms.

The **communication protocol** explains the corpus utilization by the data-interpreter and the program-supervisor. It contains two main rules. First the expert of the interpretation domain has to use the corpus to express a piece of knowledge understandable by the program-supervisor (e.g. a curve, a contour...). Second, the messages between the data-interpreter and the program-supervisor must be based on the corpus terms.

4 The implementation of the communication level

To describe the knowledge of the interpretation domain we use an object oriented formalism. The knowledge is represented by **prototypes** (i.e. object classes) which features are described by slots. At the present time the communication corpus is composed of terms that represent eight prototypes described by 59 slots:
• **input-image** (described by 8 slots like minimal, maximal and average luminosity) represents global image properties.
• **isolated-object** (11 slots like barycenter and surface) describes the natural object image without the background image.
• **contour** (8 slots like perimeter and maximal length) represents the image containing a contour.
• **axis** (4 slots like orientation and position).
• **curve** (8 slots like maximal hight and length) represents either a luminosity curve or a distance curve.
• **peak** (9 slots like maximal hight and position) is a peak in a curve.
• **trough** (9 slots: the same as the peak's ones) is a trough in a curve.
• **approximation** (2 slots : type and error) indicates the method used to approximate contours and curves and gives the error value.

The corpus terms are organized through the composition relation as showed on figure 3. For example an isolated object is composed of two contours and of three curves.

The message structure between the data-interpreter and the program-supervisor comprises five semantic components: the **message type**, an **input datum**, a list of **input parameters**, a list of **output data** and the **process context**. There are three types of messages *compute-prototype*, *compute-slot* and *provide-info*, explaining the message objective. The two first types are used for data-interpreter requests. The type *provide-info* is used by the program-supervisor to ask the data-interpreter for a datum. In this case the contents of the message are reduced to the message type and to one output datum.

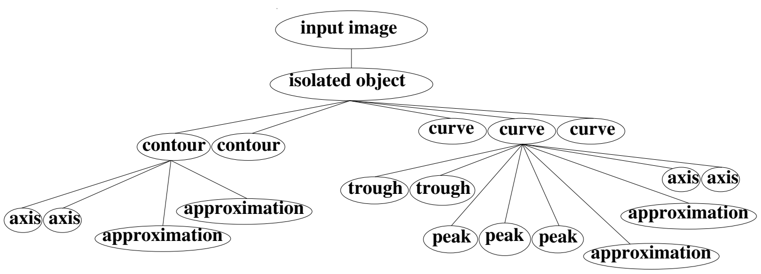

Figure 3: The corpus is organized through the composition relation.

A datum is the collection of a **label**, a **meaning** and a **value**. The label is used as a reference to the datum (e.g. a file name) and the meaning is a corpus term describing the datum type. The input datum is the current prototype on which the data-interpreter wants to obtain some results (e.g. a contour). The input parameters are data used by the program-supervisor. The output data give the result type : if the message type is *compute-prototype* then the output datum is a prototype (e.g. contour, curve...) ; if it is *compute-slot* the output data are a set of prototype slots (e.g. perimeter, surface...). The process context gives the additional information needed by the program-supervisor. This information is often dependent on the application. One typical example of context is the image quality, or the type of the sensor used.

message type	input datum	list of input parameters	list of output data	context
compute−slot	contour		barycenter, length−max, surface...	object centered...

Figure 4: The figure shows an example of a message and its contents.

So all the knowledge contained in the message is expressed thanks to the communication corpus what makes possible the communication between the two systems. First the data-interpreter can send directly its own data through messages. Second the program-supervisor can match a message with one of its image processing functionalities (e.g. the image processing functionality *compute-slot-contour*). The name of the functionality is obtained through the concatenation of the message type and the input datum as shown on figure 4.

Therefore the communication corpus gives a common vocabulary to the program-supervisor and the data-interpreter. It defines a communication level between them.

5 Benefits of the communication level

The communication level has two main qualities. First it provides help for the development stage of the data-interpreter that is planned to use the predefined program-supervisor. Thanks to the corpus the expert of the interpretation domain is able to structure its knowledge with adapted image properties.

The second quality of the communication level is its genericness. First it can easily use new image processing libraries. For this, it is only required to add new operators to the program-supervisor. As the main image processing functionalities are not modified, the communication corpus does not need to be changed. Thus old data-interpreter systems can use the new libraries.

Moreover the communication system is independent of the interpretation domain. As soon as the data-interpreter knowledge is represented through the predefined corpus, this system can use the predefined program-supervisor. The corpus has already been used for two applications (Ossola, 1996), to recognize zooplanktonic organisms and to recognize galaxies. Sometimes to develop a new application, additional terms may be needed. However, we hope that after using the corpus for a few applications, no more additional terms will need to be added. Our aim is to develop a data-interpreter to recognize a mobile object described by an image sequence. For this a new prototype *moving-object* with its slots in the corpus will be added. The corresponding image processing functionality is realized by a program that computes the difference between a current image describing the mobile object and a reference image describing the background. So we also need to add this functionality to the program-supervisor. This example shows the benefits of using a communication corpus. Even for a quite different application, the predefined communication level can be re-used.

6 Conclusion

The global object recognition system built with the proposed communication level has several advantages. It gives a common corpus to the program-supervisor and to the data-interpreter. It allows us to structure the knowledge of the data-interpreter. A new image processing functionality can be added without changing the data-interpreter. In a symmetric way, a new data-interpreter system can use a program-supervisor developed for another application as soon as they use the same corpus.

The main drawback of this approach is the learning phase of the communication corpus by the expert of the interpretation domain. In a practical point of view he or she still needs to have experience in the predefined program-supervisor to use it.

We have already developed a corpus for the recognition of zooplanktonic organisms and galaxies described by one input image. We plan to re-use it for the recognition of mobile objects described by an image sequence. We hope that the extension of the corpus will prove that the proposed communication level can be extended to a large class of recognition applications.

References

Brooks, R. A. (1989). A robot that walks: Emergent behaviour from a carefully evolved network., *Neural computation, MIT Press* .

Clément, V. and Thonnat, M. (1993). Integration of image processing procedures, OCAPI: a knowledge-based approach., *Computer Vision Graphics and Image Processing : Image understanding* **57**(2).

Dellepiane, S., Venturi, G. and Vernazza, G. (1992). Model generation and model matching of real images by a fuzzy approach, *Pattern recognition* **25**(2): 115–137.

Granger, C. (1985). *Reconnaissance d'objets par mise en correspondance en vision par ordinateur*, PhD thesis, Université de NICE.

Liu, S., Thonnat, M. and Berthod, M. (1994). Automatic classification of planktonic foraminifera by a knowledge-based system, *IEEE - 10th Conference on artificial intelligence for applications* pp. 358–364.

Loonis, P., Zahzah, E. and Bonnefoy, J.-P. (1995). Cooperation of Various Classification Techniques : Application to Complex Objects Recognition, *Proceedings of the International Conference on Systems Research, Informatics and Cybernetics : Advances in Intelligent Data Analysis* **I**: 109–113.

Meisels, A. (1991). Levels of knowledge for object extraction, *Machine vision and application* **4**: 183–192.

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.*, PhD thesis, Université de Nice Sophia Antipolis.

Ossola, J. C. and Thonnat, M. (1995). A distributed architecture for automatic object recognition, *Proceedings of the 7th International Conference on Systems Research Informatics and Cybernetics - Advances in Intelligent Data Analysis* **I**.

Thonnat, M., Clément, V. and Ossola, J. C. (1995). Automatic galaxy description, *Astrophysical Letters and Communication* **31**(1-6).

Thonnat, M. and Gandelin, M.-H. (1988). An expert system for the automatic classification of zooplanktons from monocular images, *9 th International Conference on Pattren Recognition ICPR9*, pp. 114–118.