

Generic Shape Learning and Recognition¹

Alexandre R.J. François² and Gérard Medioni

Institute for Robotics and Intelligent Systems
University of Southern California
Los Angeles, California 90089-0273
{afrancoi,medioni}@iris.usc.edu

Abstract. We address the problem of generic shape recognition, in which exact models are not available. We propose an original approach, in which learning and recognition are intimately linked, as recognition is based on previous observation.

The input to our system is in the form of segmented descriptions of objects in terms of parts. In 2-D, the shape is incrementally decomposed into parts suggested by curvature sign changes, and for each part an axial description is derived from both local and global information. The parts are organized into a connection hierarchy. For 3-D objects, we intend to use segmented tridimensional descriptions, the parts being modeled by generalized cylinders. In this case, the connection graph is not necessarily a hierarchy, but can still be used with our algorithms.

The part description obtained at this point is still too detailed and fine grained in order to easily categorize and compare shapes. Hence, we use a simplified description of parts, capturing part local geometry and connection with the superpart information. The local geometry parameters are qualitative and symbolic, and are quasi-invariants under projection and viewpoint change. Both types of parameters take discrete values derived from the available fine description. The connection parameters are normalized to be scale-independent. These simplified part descriptions are organized into a connection hierarchy as provided by the original decomposition. The parameters are chosen to ensure that the information carried in these descriptions is sufficient to perform shape recognition.

Actual shape descriptions are stored in a data-base, from which they must be efficiently and specifically retrieved when a new shape is proposed for recognition. We define a hierarchical indexing system based on the structure of the descriptions and the local description of parts. This mechanism allows for dynamic updating of the data-base with a minimum computing cost.

When a shape is submitted for recognition, the data-base is searched for the closest known shapes. A partial match, based on the connection structure and the aggregation of dissimilarities between parts, is computed incrementally level by level between the shape and the possible candidates. The combination of the incremental process with the hierarchical indexing makes the number of shapes processed at each step decrease rapidly, therefore dramatically reducing the average complexity of the retrieval. The selected retrieved shape(s) are used to give a classification for the submitted shape.

This approach to recognition is influenced by the Case-Based Reasoning (CBR) paradigm, which embeds all the characteristics required to meet our goals, such as the ability to process noisy, incomplete and new data. It also provides an interesting framework for higher-level intelligent processing (e.g. justified interpretation, automatic learning).

We describe our implementation for 2-D shapes recognition and present results. The current implementation should also work on 3-D descriptions as described above, with minor changes. We also intend to use this system as the core of a higher-level vision-based reasoning system.

1. This research was supported in part by the Advanced Research Projects Agency of the Department of Defense and was monitored by the Air Force Office of Scientific Research under Contract No. F49620-90-C-0078 and/or Grant No F49620-93-1-0620. The United States Government is authorized to reproduce and distribute reprints for governmental purposes notwithstanding any copyright notation hereon.

2. Sponsored for this research by NOESIS S.A., Immeuble Ariane, Domaine Technologique de Saclay, 4 rue René Razel - Saclay, F-91892 Orsay Cedex, FRANCE.

1 Introduction

Generic shape recognition is a major problem in image understanding. In order to perform shape recognition without *a priori* exact geometric or semantic knowledge about the observed objects, we need to produce context independent high-level shape descriptions. Many researchers have discussed this problem (see [16] for an overview). The conclusion is that a “good” shape description should produce segmented parts organized in a hierarchy. Recent progress in the quality of resulting descriptions in 2-D and 3-D makes it relevant to consider such recognition (see e.g. [17][20]).

Our design of a recognition engine is influenced by the Case-Based Reasoning (CBR) paradigm successfully applied in Artificial Intelligence [11][18]. In this context, the system uses a visual memory, which is an organized set of previously described and identified shapes, to propose a documented classification of a given shape. Therefore shape descriptions are dynamically and incrementally stored in a database (learning process), from which they must be efficiently and specifically retrieved when a new shape is proposed for recognition. This implies an efficient *indexing*, combined with an appropriate *retrieval* algorithm. The originality of this approach is that recognition is based on previous observation, which means that the problem addressed is really shape *recognition* as opposed to shape identification (model matching, pattern recognition, etc.).

Here, we describe a recognition system featuring symbolic hierarchical description of shapes, hierarchical indexing of the database and incremental partial-matching retrieval. The principles presented are illustrated by experimental results obtained with an implementation based on the hierarchical 2-D shape description method described in [17]. We start with a brief overview of the work in shape recognition, and an overview of the CBR paradigm. In section 3, we present the symbolic hierarchical description model we are using. Section 4 describes the dynamic database organization process, and section 5 the retrieval process with a complexity study. Tests performed on real and simulated databases are presented in section 6. Finally, we summarize our contribution and discuss possible extensions for the system.

We do not address the actual processing of the images to obtain shape descriptions (figure-ground problem, segmentation, feature extraction, etc.), and assume that the input to the system consists of adequate descriptions of shapes, segmented into parts, as produced in [17] or [20].

2 Previous Work

2.1 Recognition in Image Understanding

Most of the studies in object recognition are concerned with recognizing one object, for which an exact geometric model already exists, by finding its position and orientation (see [6]). In a sense, this should be referred to as pose estimation and calibration. We are interested in the issues associated with a very large number of entries, and the absence of exact models (genericity).

A number of systems developed use low-level primitives for recognition. These primitives are chosen to exhibit invariance properties to viewpoint and occlusion. Representative examples of such work are Bolles and Cain’s local-feature-focus method [3], and Grimson and Lozano-Pérez’s interpretation tree [7][8]. The major drawback of these methods is that recognition relies on low-level features, which usually occur in large numbers

in a scene and in relatively low number in objects, and have a low discrimination power, because of the large search space involved.

These methods usually ignore the complexity associated with the number of models. In order to be able to handle large model data-bases, indexing techniques were introduced, and even became the recognition engine foundation. A few examples of “indexing-based” recognition systems are [10],[12],[4],[19] and [9]. However, we do not believe that low-level primitives allow to perform high-level, human-like recognition.

Recently, Murase and Nayar proposed in [13] an original approach using actual object images instead of models. As a corollary, learning is an inherent feature of their system. Also, they do not explicitly address shape, which simplifies the process of recognition. While this system is useful for a number of application scenarios, several aspects do not meet the goals we stated: it does not allow occlusion, and higher-level interpretation cannot be conducted from the recognition (and pose estimation) result. We also note that the system’s visual memory can only be updated off-line, involving the recomputation of the universal eigenspace, which is extremely costly in terms of image storage since all the learning images have to be available at the time of an update.

Three main bodies of work present ideas that give partial answers to the problems identified above:

- Nevatia and Binford [14] developed a system that can recognize curved objects from range images (3-D from $2^{1/2}$ -D). They present a technique for generating structured, symbolic descriptions of objects by segmenting them into simpler subparts. The description is based on generalized cones. An object is represented by a connection graph in which the parts are described with a small number of symbolic parameters. The recognition is performed at a symbolic level, but lacks an efficient indexing method taking advantage of the description model and therefore cannot handle a large number of models. Another important aspect of this work is that the “models” used for recognition are object descriptions generated from actual object observations.

- Biederman’s recognition-by-components (RBC) theory [2] shows that this part-oriented, symbolic approach of object recognition is a mechanism used in human image understanding. According to this work, only a relatively small set of part primitives is enough to describe all possible complex objects. The information describing the object is a description of the parts, and the description of their relations. Moreover, the recognition of a complex object can be performed considering its few main parts, and eventually refined if necessary.

- Ettinger uses in the recognition paradigm he defines in [5] several notions that complement some aspects of the ideas found in Nevatia and Binford [14]:

- Decomposition of object models into subparts hierarchies.
- Non-exact matching recognition.
- Hierarchical indexing of the data-base.

However, the model object representation uses low-level features and therefore the recognition cannot be performed (and explained) at a symbolic level.

Given these works, and keeping in mind that we want to perform *re-cognition*, explicitly based on previous observation, we derive four principles to achieve our goal:

- *Hierarchical graph description*: the shapes should be described in terms of elementary parts and their connections, and should be organized hierarchically in function of the scale or size (at least a coarseness order has to be defined on the parts to orient the description graph).

- *Symbolic part description*: the description of parts used for recognition should be *symbolic*, and clearly separated from the low-level feature extraction process. It should also be context independent (the context parameters are separated from the generic description): scale normalization, orientation independence, etc.

- *Hierarchical database organization*: in order to handle large databases, the visual memory has to be organized, and a hierarchical indexing is expected to be an efficient solution in the RBC model context.
- *Partial matching*: The recognition paradigm is built on a partial-match hypotheses generation, followed by interpretation and validation.

Recently, Provan *et al.* [15] described learning of 3-D object models using Bayesian networks. They use volumetric segmented object description, the parts being modeled by generalized cylinders. Although their approach is motivated by similar goals, the general framework in which they work is different from ours, as they use a *probabilistic* method to perform learning of object *models*.

2.2 The Case-Based Reasoning paradigm

Given the principles above, we have found that they can be accommodated within the general framework of CBR defined in Artificial Intelligence. Clearly, CBR is not expected to be a turnkey solution to the difficult problem enunciated above; rather, it provides guidelines which need to be significantly refined to be turned into a working system.

CBR is a general paradigm for reasoning from experience [11][18]. Its developments in Artificial Intelligence include the representation of episodic knowledge, memory organization, indexing, case modification and learning. Since it is also a psychological theory of human cognition, it may be a natural reasoning model for performing intelligent tasks.

The underlying principle in CBR is that the reasoning is based on actual previous experiences (or *cases*) rather than on artificially theorized knowledge [1], such as for instance Rule-Based Reasoning or Neural Networks. This mainly avoids the difficult to control loss of information that occurs in the production of a set of rules or training of a neural network. The low-level processes of the recognition, i.e. case description and retrieval of the closest known cases, are only based on case data and meta-knowledge. The synthesis, without which there is no reasoning, occurs at a higher level, using more meta-knowledge, and specific knowledge if needed. In any case, this integration is clearly separated from the early steps of the reasoning process (retrieval), and the introduction of knowledge is precisely confined and controlled. Furthermore, interpreting the retrieval result means analyzing the similarities and inferring a relations between the problem to be solved or the situation to be interpreted, and the closest known problems or situations. Therefore a solution is always documented.

All these principles allow us to seriously consider *generic* shape recognition. Since the system relies on previous experience, it is *adaptive* and *expandable*: learning is an inherent feature of the system.

Since it is not our point to present CBR in this paper, we will not develop in detail its specific aspects. We will just point out that in the context of shape recognition (which is an interpretation problem), a case contains two sets of data. The *shape description* includes the symbolic description (as described below) which will be used for the retrieval of similar shapes, and information about the context in which this description has been produced (e.g. normalization parameters, original image, low-level data, reference to the original image(s), etc...). The *identification* data can simply be the class to which the object belongs (as in our current implementation, in which the qualitative reasoning following the retrieval is mainly left to the operator), or it can be a more complex and organized object which will be used for the interpretation of retrieved cases, and for the interpretation of complex scenes for example. In the remainder of this paper, the word *case* will sometimes be used to refer to the symbolic shape description used for recognition (which is the part of the case which is considered for the recognition steps

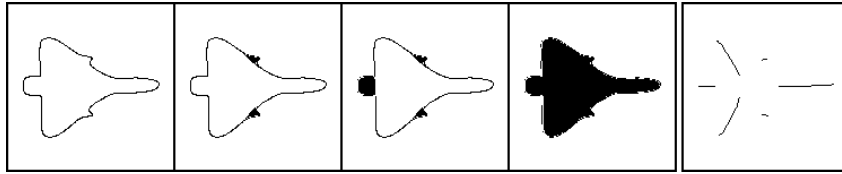


Fig. 1. Decomposition of an “f106” shape (from [17])

studied) and consequently, the term *case-base* will refer to the shape database. We want to emphasize that any shape description used in the system is supposed to be the description of an actual observed shape, and that the system never relies on some artificial object model.

3 Symbolic Hierarchical Generic Shape Description

In this section, we describe a general representation model for generic objects, and discuss how to build a case for the recognition system. We illustrate all the principles described with 2-D shapes originally processed as described in [17]. The extension of these principles to 3-D objects description is discussed in section 3.3.

Shape recognition is an interpretation problem. The solution is based on the interpretation of similarities and differences between a proposed shape and a set of known shapes. Therefore, the description model used to manipulate the shapes has a fundamental influence on the quality of the recognition. The first step to a good representation is a good low-level processing of the original data. We assume that such a description is available (see example in Figure 1).

3.1 Hierarchical Shape Decomposition

Principle

We characterize complex shapes using a small number of elementary parts, consistently with Biederman’s geons theory [2]. We apply this process to 2-D shapes and it can easily be adapted for 3-D objects. We briefly outline below the process to generate these descriptions from real images, more details can be found in [17].

Planar shapes are first described by a B-spline approximation of their contour. The contour is initially segmented at curvature sign changes into potential local parts, that are described by their Smooth Local Symmetry axis. The local description is complemented by the computation of parallel symmetries. Given this local information and global relationships, the shape is hierarchically decomposed into parts, by first removing the small and well defined parts and then by analyzing the remaining shape. This results in a natural axial description of the shape together with a hierarchical decomposition into its parts (see Figure 1).

Once the parts are identified and described, they are organized into a directed connection graph (see Figure 2). The edges are oriented from the larger to the smaller part, according to the size parameter chosen. For the 2-D shape description used, the size of the part is defined by the length of the axis. The shape description used for recognition is composed of the description of its parts translated into a symbolic description model (presented below), organized into a connection graph. This allows qualitative reasoning by finding and analyzing correspondences between the graph describing an input shape and a graph describing a possible model.

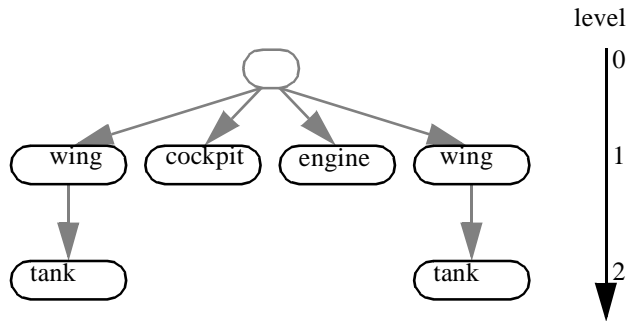


Fig. 2. Directed connection graph obtained from the decomposition of the “f106” shape shown in Figure 1 (hierarchy of depth 3).

Properties of the Description Graph.

The description graph in the case of 2-D shapes is a tree. In our implementation, the root contains no information used for the recognition. The root is the only node of level 0. For a description hierarchy of depth d , the levels are numbered from 0 (root) to $d-1$ (see Figure 2).

General properties of objects and the construction process for the connection graph allow us to infer structural and semantic information of such a description graph:

- A complex object is made of a limited number of main parts which, given our orientation of the description graph, will appear on the first level of the description hierarchy. Hence the first level is a coarse description of the shape (only the main parts), which is enough to infer an initial classification for the shape [2]. Each additional level refines the description.
- The depth of the description is limited to very few levels by the resolution of observation. As a corollary, the total number of parts is also limited by the resolution of the image.

3.2 Symbolic Description of Parts

We first generate symbolic parameter values derived from the low-level description of the shape and of the segmented parts.

Intrinsic Part Description: Local Geometry

Parts are characterized by intrinsic geometrical properties similar to the geons in [2]. For our 2-D parts description, we consider the following set of parameters (see Figure 3): the type of boundaries symmetry (parallel or non-parallel boundaries), the type of curvature (positive, negative or null) and the type of termination (mono-angular or pluri-angular). These symbolic parameters are obtained from the low-level description. Their qualitative nature makes them relatively insensitive to noise (the larger the part, the less sensitive the parameter values). The combinations of the possible values for these parameters define a limited set of elementary 2-D patterns that can describe any segmented part. A small number of special types of parts may also be defined to describe particular cases (e.g. blobs). It is important to emphasize the fact that these local characteristics of the part are determined by a process taking into account both local and global features of the shape [17].

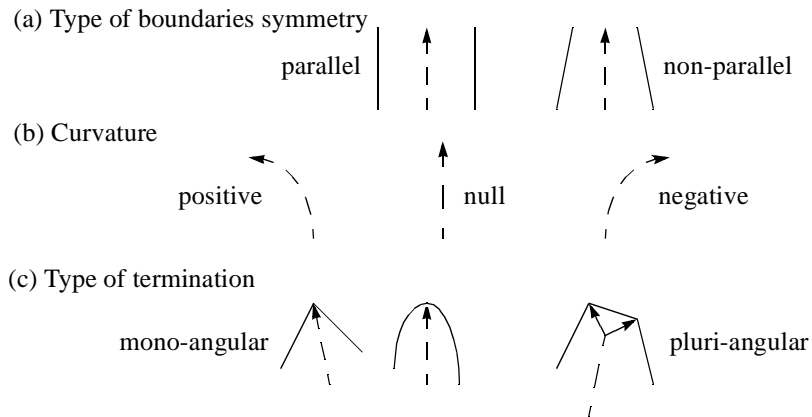


Fig. 3. Local geometrical characterization of parts: (a) symmetry; (b) curvature; (c) termination

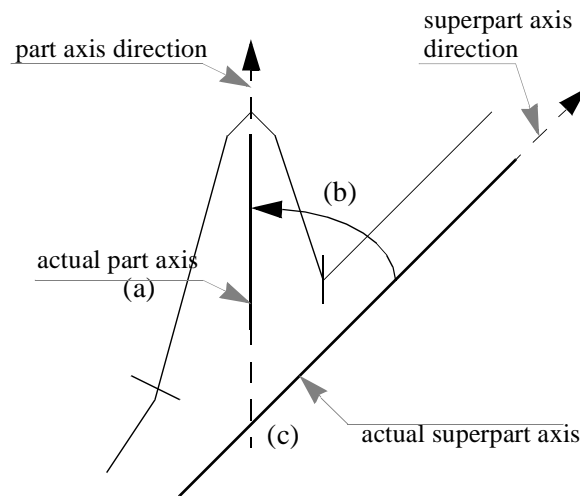


Fig. 4. Relative parameters for part description: (a) size of the part (normalized length of axis); (b) relative angle between the part axis and its superpart axis; (c) type of connection.

Geometrical Organization of Parts

A shape is also described by relations between connected parts. The description process used produces a connection hierarchy, which means that a part is only related to its superpart. The parameters we consider to describe planar shape parts relations are (see Figure 4): the size of the part (normalized length of the axis), the type of the connection with the superpart and the angle between the superpart axis and the part axis (if the axis are curved, we consider the angle between the tangents at the projected intersection point). The type of connection parameter takes symbolic values and is rather noise tolerant. The other two parameters are more subject to noise alteration, so their quantization for the recognition is delayed until the part comparison process (see section 5.2). Their values are directly computed from the low level description. The length of the axis is normalized by the length of the longest part axis in the shape in order to produce a scale independent description. There exist many other

possibilities that could for instance make this normalization insensitive to occlusion on the largest part, and more experimentation is needed to fine tune them. The normalization parameters are stored as part of the shape description, so that they can be used during the interpretation of the retrieved shapes.

The description obtained contains the information necessary to rebuild a scale-independent symbolic version of the original shape in which the geometrical relations are kept. An example of such a symbolic shape description is shown in Figure 2.

3.3 Extension to 3-D Objects Description

Very few modifications would be necessary to adapt the system for 3-D object recognition. The low-level description process used is the one described in Zerroug and Nevatia [20].

The object decomposition and the part description is directly adaptable from Biederman's geons [2] and Zerroug and Nevatia's [20] 3-D object segmented description. The result is a symbolic description of the 3-D object as a set of 3-D part symbolic descriptions organized in a connection graph. Compared to a 2-D shape part description, a 3-D object part description requires several additional parameters to describe spatial relations. Moreover, the importance of some parameters may differ. For instance, the type of connection between parts is a fundamental information for 3-D objects. The description graph can be oriented using the size of the parts (as we orient the 2-D shape description graph). In general it is not a tree, but a mono-rooted oriented acyclic graph (parts can have more than one connected part in a 3-D object). This structure is compatible with all the processes described later in the paper.

4 Dynamic Database Organization

If a good description of cases is the first requirement for shape recognition, the retrieval of known shapes from the visual memory is the core of the recognition engine. Once a shape is translated into a case, it has to be stored in a data-base from which it can later be efficiently retrieved when a described shape is proposed to the system for recognition.

Since we want the system to be able to handle very large databases, an efficient retrieval relies on two complementary aspects: an adequate organization of the case-base (identified as the *indexing problem*) and an adapted retrieval algorithm. In this section, we present the indexing mechanism we have developed and implemented. The complementary retrieval algorithm is described in the next section. In these two sections, the word "shape" refers to the symbolic description used for recognition.

4.1 Hierarchical Indexing of Shape Descriptions

In the recognition process, the search for similar cases is data-driven, which implies an efficient indexing based on shape descriptions for retrieval. The most natural and efficient data structure for indexing objects for retrieval is a hierarchy. Organizing the shapes in a hierarchy requires the definition of a partial order on the shapes, which we will infer from the description graph properties outlined in 3.1. Since the retrieval is data-driven, the hierar-

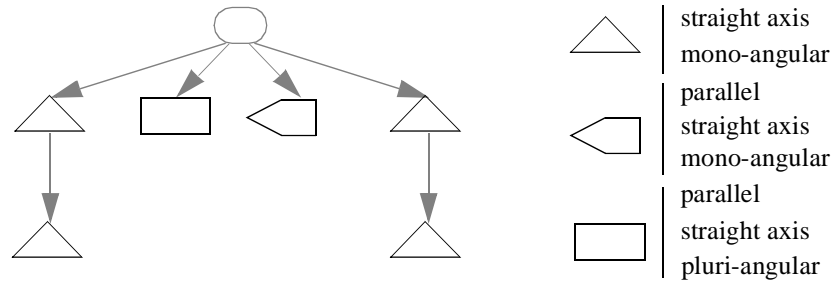


Fig. 5. I-Structure derived from the “f106” shape symbolic description

chical indexing should be based on the structure of the description hierarchies. This structural “indexing key” must be complemented by a simplified description of the parts (*i.e.* a subset of the description parameters).

Definition: We call *I-Structure* (for Indexing Structure) the simplified description graph obtained from the original symbolic description graph by reducing the number of parameters for the description of parts. The I-Structure is isomorphic to the original description graph and since the parameters used to describe the parts are a subset of the parameters used in the symbolic description used for recognition, the I-Structure information is strictly contained in the description of the shape.

Choice of Part Description Parameters for Indexing

Consistently with our part-oriented approach of recognition, the description of parts used for the I-Structure includes the local geometrical information. This choice is supported by the fact that the local geometry of the parts is less subject to variations due to noise, and is described with symbolic parameters. Since we would like the recognition system to be noise tolerant, it seems logical not to consider the parameters describing the relative position of parts. This also allows not to assume the objects to be rigid, but rather to be made of articulated rigid parts.

Finally, we use as I-Structures for our index the partial descriptions of shapes that is obtained by considering for the description of parts only the local geometry description parameters (see Figure 5).

Hierarchical Indexing of Hierarchical Structures

We propose a hierarchical indexing method for hierarchical structures. To each node of the index is associated one (and one only) I-Structure. The cases referenced by the node are those which description matches *exactly* the node’s I-Structure. We organize the nodes into a *specialization hierarchy*, by defining a partial order on the I-Structures:

An I-Structure S' is a direct specialization of an I-Structure S iff

- $depth(S') = depth(S) + 1$
- S and S' are identical down to depth $depth(S)$

We present in Figure 6 an example of hierarchical indexing of hierarchical structures built from three types of parts, based on the structure itself and on the type of the nodes at a given level.

Given the partial order defined above, the hierarchical index inherits properties from the description hierarchies:

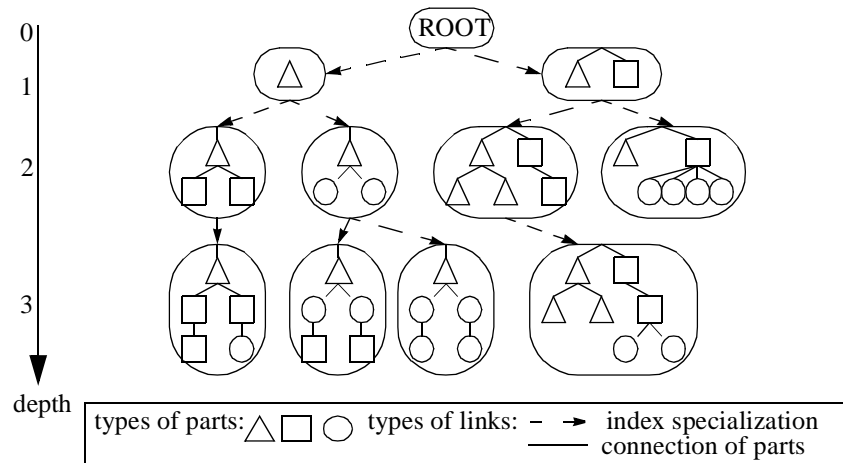


Fig. 6. Example of hierarchical index, discriminating shapes on the structure of their description and on local characterization of parts.

- Since the first level of a description hierarchy is a coarse description of the shape, the first level of the index represents a coarse filter (it allows initial discrimination hypotheses based on the few main parts of the shapes).

- The deeper levels allow to focus and refine the early recognition hypotheses.

Therefore, the branching factor of the root is expected to be high because of the diversity expected among observed shapes, even in terms of their main parts. On the contrary, the branching factor at deeper levels should be much lower due to the reduction of possible details of the shapes once they are coarsely characterized (and to the reduction of observable types as the parts become smaller in the image).

We want to point out that this indexing technique easily applies to acyclic oriented mono-rooted graphs which would occur for description of 3-D shapes (see section 3.3). Given the importance of the type of connection between two parts in the 3-D object description, this parameter should also be used in the I-Structure for 3-D object indexing.

4.2 Dynamic Evolution of the Database

One important feature of the indexing mechanism presented above is that a shape can be added to the data-base at anytime without any recomputation of the existing structure. If the corresponding nodes already exists, the new shape is added to the list of shapes pointed by the node. If the deepest node with a compatible structure does not have exactly the same structure (which means that the description hierarchy of the current node is deeper than the I-Structure attached to this node), then the needed nodes are created.

This dynamic updating principle allows to start either with an important list of shapes in the data-base or with a minimum data-base which is incrementally updated with new shapes (involving supervised learning).

4.3 Discussion

Since no specific knowledge about the described shapes is used in the description and indexing processes, the shapes in the index are not grouped semantically but *geometrically*. For example, a front and side views of a car will not be related directly through the index hierarchy. From the recognition point of view, this is coherent with a symbolic generic shape processing. Practically, this also means that the indexing is a pre-processing for the retrieval, consistent with the data-driven approach, which is a good point for retrieval efficiency, as we shall prove in section 5.4.

5 Shape Retrieval

In this section, we present a retrieval mechanism based on a partial matching of shape descriptions which takes advantage of the index. After justifying our preliminary choices, we define a method for the evaluation of both digital and symbolic similarities between parts, then describe the core of the retrieval process.

5.1 Exact Matching vs. Partial Matching

In most of the previously considered approaches, the recognition is based on an exact matching search of the model memory.

The indexing method described above allows to conduct such a search efficiently. Indeed, if one uses shape models instead of actual shape descriptions, each model is expected to have its own I-Structure which correspond to *one* node in the index. Since exact matching by itself doesn't meet our goals, we don't want to discuss here in detail the complexity of this search process, but one can see that the worst case cost, which is linear in the number of shapes in the database without using the index, is independent of the total number of shapes and becomes linear against the branching factor and the depth of the index hierarchy which are determined by the properties of the set of models. The hierarchical indexing of hierarchical structures principle presented in the previous section can be used very efficiently in a traditional model-based approach, without however making it more relevant for generic shape recognition. An exact matching search presents many draw-backs incompatible with generic recognition, especially in our "recognition from previous actual observations" approach. It ends in failure if the same exact shape is not found in the data-base. If the description process is not stable enough or if the data is incomplete or noisy, recognition cannot be performed, and no interpretation of the input shape is proposed. This is the main reason why we consider a partial matching retrieval approach.

The partial matching retrieval produces hypotheses for the recognition. It is based on a metric to evaluate a similarity (or a dissimilarity) between shape descriptions, complemented by qualitative information about the comparison to allow "high-level" (symbolic) solution generation and explanation.

5.2 Dissimilarity Between Shapes

Our evaluation of the dissimilarity between two shapes is based on the definition of "transition costs" that represent the cost of the assumption that two different "symbolic" objects (parameter values, parts, shape descriptions) have been obtained from the same real object, because of noise in the original data, variation of the observation conditions, etc. The processing of such costs between parts and their aggregation in costs between

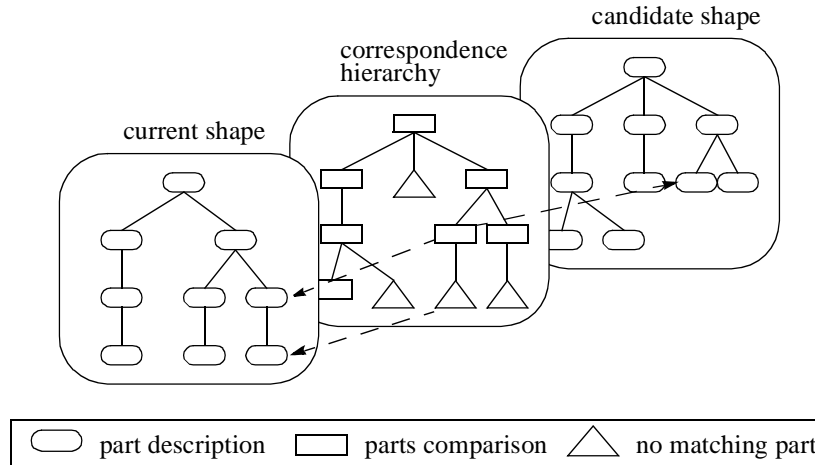


Fig. 7. Example of comparison structure instantiated between two hierarchical shape descriptions. Only a few links are shown to preserve readability.

shapes is described in the following subsections, along with the definition of a symbolic structure that facilitates the digital computation and allows to perform as well a symbolic, qualitative comparison of shapes.

Transition Cost Between Two Parts

At the lower level, transition costs are defined for each parameter. For symbolic parameters, a cost is attributed to each possible couple of its range of values. This transition cost is therefore a discrete function expressing the cost of the assumption that two different symbolic values for a parameter have been determined from compatible original data (as a result of noise in the image for example). For numerical parameters, we define discrete cost functions expressing the cost of the assumption that the difference observed between two values was caused by noise or context variations. In our implementation, we made sure the costs we defined were distances.

The aggregation function used to compute the dissimilarity (transition cost) between two parts is a weighted sum of the parameters transition costs. Weights can be used to give more importance to some of the parameters. In our current implementation, all the parameters are given the same weight.

Pairing the Parts: Correspondence Hierarchy

In order to easily compute the transition costs, we introduce a comparison structure which is a *correspondence hierarchy* instantiated between two shape descriptions (see Figure 7). Each node of this graph points to two matching parts of the considered shapes, and its position in the correspondence hierarchy is similar to the position of the parts it links in their respective description hierarchies. A special type of node is defined for non paired parts, and the transition cost associated to such a node is maximum (i.e. automatically set to 1).

The strategy we used in our implementation is the following:

- First try to find a pairing part in the known shape subparts list for each of the parts of the proposed shape subparts list, by selecting the closest (according to the defined transition cost) among the remaining subparts. If the cost between the two paired parts is equal to 1 (which can occur if parts are available for pairing but are completely different), then a missing part node is generated for the proposed shape part.

- As soon as one of the subparts list is completely paired, missing part nodes are generated for the remaining subparts in the other list.

In this approach, the matching search is based on the proposed shape data, which is consistent with our data-directed retrieval paradigm. The algorithm only ensures that associated parts have the smallest transition cost. A “security” threshold prevents the system from pairing two completely different parts.

The correspondence hierarchy is an adequate structure for the comparison process. It allows to store both digital and symbolic comparison data for later interpretation and reasoning about the retrieved shapes. It is also a key component for the retrieval algorithm. For efficiency, in our implementation the index nodes actually point to pre-instantiated comparison structures for each shape description in the database.

Transition Cost Between Two Shapes

Let d be the depth of the correspondence structure, and k_l the number of parts comparison nodes on level l ($0 < l < d$). Each node in the correspondence hierarchy (characterized by its number i ($1 < i < k_l$ on level l) is attributed the cost $c_{l,i}$ between the two parts it pairs. The cost between the two shape descriptions associated to this comparison structure is defined as:

$$C = \frac{\sum_{l=1}^{d-1} \left(\frac{k_l}{i=1} A(l) \sum_{i=1} c_{l,i} \right)}{d-1} \quad (5.1)$$

$$\sum_{l=1} k_l$$

The factor $A(l)$ is a decreasing function of the level which expresses that the presence of a detail is a strong argument in favor of the match, whereas the absence of a detail is not a reliable information. We propose to use the following function:

$$A(l) = \left(\frac{d-l}{d-1} \right)^q$$

defined for $0 < l < d$, and where q is an integer (in our implementation, $q=2$).

5.3 An Incremental Partial-Matching Retrieval Algorithm

Principle

The most computationally expensive task in the retrieval process is to find the correspondence between two shape descriptions. This is a subgraph isomorphism problem, which is NP-complete. We have developed an algorithm which uses the index and the correspondence structures to avoid computing this expensive correspondence for *all* shapes.

The principle of our algorithm is to build the comparison structures level by level, keeping at each stage only the structures which point to a *compatible* case for the current level. This is made possible by the hierarchical indexing of cases. At the beginning of the retrieval process, all *compatible* nodes of the first level of the index are selected, and the first level of the comparison structure is built for all the cases pointed by these nodes and the nodes in their sub-trees. The order defined for the descriptions and the indexing ensures that this first level correspondence is a coarse comparison. Hence an early diagnosis, highly discriminative, can be based on the

closest shapes for this level of detail. To build the next level (which makes sense only for cases which actually have parts of a deeper level, *i.e.* indexed by deeper index nodes), the *partially compatible* subnodes of the previously selected nodes are selected and the process is reiterated, increasing the precision of the comparison.

In the average case, the algorithm ends after the last level of the proposed shape has been processed, or after no candidate is left in the database for further investigation. In most cases, at least one shape will be retrieved for interpretation. The only case in which no shape can be retrieved is when no compatible first level node can be found, which means that the system has to learn more shapes.

Selection of the (Partially) Compatible Nodes

A critical notion in this algorithm is the (partial) *compatibility* between a shape description (or rather the sub-description called I-Structure) and the I-Structure attached to an index node. This determines the nodes that are selected on each level and therefore have both semantic and efficiency implications.

As suggested by the previously outlined semantic properties of the index (see section 4.1), this compatibility has to be considered separately for the first level (level 1) of the index and for the deeper levels:

- The selection of the first-level nodes is a key point of the use of the index. Since the large parts local geometry is less subject to noise alteration, the only factors we have to consider are the occlusion problem, and the ability to find close shapes even if the proposed shape is unusual (in the sense of different from the shapes in the database). Occlusion can result either in the absence of a part in the shape or in its replacement by a “degenerate” part. We have to keep in mind that the chosen strategy only affects the first level nodes selection, which means that the occlusion considered here occurs on main parts of the shape. Even a human cannot recognize a complex object whose main parts are occluded: consider for example a plane shape with occluded wings, it is easy to see a pencil, a rocket, etc. Therefore strong assumptions about the quality of this level of description will not degrade dramatically the expected performance of the system in terms of quality of the recognition. Hence a first possible selection strategy consists of keeping on the first level the one and only node exactly compatible with the input shape, assuming that no occlusion occurs on the main parts. It should not allow the system to handle extreme cases, but should make the retrieval very efficient since the number of selected nodes on the first level is equal to 1 (see complexity analysis below). Occlusion of main parts can be handled by considering a node partially compatible with the current shape if the node’s I-Structure first level contains a percentage of matching parts (from the local geometry point of view) with the shape first level. This percentage has to be adjusted to allow the system to process shapes with occlusion and to find at least one compatible node for new shapes. A 100% requirement, which means that a node is compatible if all the parts on level 1 of its I-Structure match one part in the shape I-Structure level 1, does not allow to process bad cases of occlusion (the occluded part is replaced with a degenerate part geometrically different) but allows to deal with total occlusion of parts and with nice partial occlusion cases since the node’s I-Structure is allowed to have more parts than the input shape on the considered level. A low percentage results in the selection of many nodes and compromises the efficiency of the retrieval. In general, this parameter depends on the properties of the shapes in the database and on the variability expected for the input shapes. For example, this approach would not be efficient (in term of quality of the recognition) with a database containing occluded shape descriptions. We present tests of these strategies in section 6.2.

- For the selection of deeper nodes, we consider that the next comparison level between the current shape and a shape in the data-base is worth computing if they present similarities down to this level of the description. In our current implementation, we require that at least one entire branch (down to the current depth) can be

matched between the two I-Structures (parts matching is based on local geometry). This may be considered as a rather loose constraint, but gives satisfactory results (see section 6.2). It is also easy to implement recursively and is not computationally expensive in the average case compared to a distance computation (see complexity analysis below).

5.4 Complexity Study

In this section, we perform a theoretical study of the complexity of the retrieval algorithm, as described above. Experimental validation is presented in section 6.2.

For this complexity study, we consider a case-base in which all the shapes descriptions have the same depth, so that the all cases are indexed by the leaves of the index tree, on the same level. The cases are supposed to be uniformly distributed in the leaves of the index tree. Given these assumptions, the parameters of a retrieval process are:

- n : number of shapes in the database.
- d : depth of the shapes in the database, and under our assumption of depth uniformity of the shapes, depth of the index tree (d levels from 0 to $d-1$).
- k : number of parts on each level of a description hierarchy. This number is assumed to be independent of the level (see section 3). It is also independent of the number of shapes in the database.

The parameters d and k are characteristic of the shape descriptions, and are independent of the number of shapes in the database. Therefore they will be treated as constants in the remainder of this section.

Since we are mostly interested in the performance of the retrieval system with large data-bases (several hundreds to several thousands shapes), the complexity is evaluated as the number of operations computed between parts (distances and compatibility tests) as a function of n .

Searching Without Using the Index

As a baseline to evaluate the improvement in complexity provided by using the index, we first compute the number of distances processed to build the correspondence hierarchies for all shapes. The number of distances computed between the parts of two descriptions to build a level l in the correspondence hierarchy is a function $K(k)$, independent of l , and with a complexity of $O(k^2)$.

With these assumptions, the average number of distances processed is:

$$n \sum_{l=1}^{d-1} K(k) = n(d-1)K(k) \quad (5.2)$$

and the retrieval is in $O(n)$.

Taking Advantage of the Hierarchical Indexing

Distances Computed: As explained above (see section 4), semantics of the hierarchical index make us separate the processing on the first level from the processing on deeper levels. Let b_l be the branching factor on level l of the index hierarchy (i.e. the average number of subnodes of a given node from level $l-1$) and c_l the number of subnodes kept at level l (compatibility factor: $c_l < b_l$).

- *First level:* The number of nodes on the first level (b_1) is not independent of the number of shapes in the database. We note $b_1 = f(n)$. Obviously, f is an increasing function of n and is always smaller than or equal to

n . For small values of n , $f(n)$ will be close to n . The asymptotic limit of f is imposed by the descriptive power of the I-Structure nodes parameters (number of parts observable at first level times the number of possible combinations of the local geometry description parameters). If these description parameters are adequate for a good coarse discrimination of the shapes, $f(n)$ is expected to have a linear domain for values of n above the efficiency threshold:

$$f(n) = B \cdot n, \text{ where } B \text{ is a constant and } B \ll 1 \quad (5.3)$$

We will consider c_1 constant, and provide experimental validation in section 6.2.

- *Deeper levels:* In accordance with the expected (and observed) structure of the index, we assume that the branching factor and the compatibility factor are constant for levels deeper than level 1: $\forall l \geq 2, b_l = \beta, \forall l \geq 2, c_l = \gamma$.

The average number of shapes considered for level 1 is nc_1/b_1 . The average number of shapes considered at level $l > 1$ is $n(c_1/b_1)(\gamma/\beta)^{l-1}$ and the average total number of distances processed for the retrieval is:

$$n \frac{c_1}{b_1} K(k) \left[1 + \sum_{l=2}^{d-1} \left(\frac{\gamma}{\beta} \right)^{l-1} \right] \quad (5.4)$$

This formula shows that the gain in terms of distances computed provided by the indexing is mainly determined by the compatibility factor over branching factor ratio on the first level of the index hierarchy. With $c_1 = 1$ and $f(n) = B \cdot n$, (5.4) becomes:

$$B^{-1} K(k) \left[1 + \sum_{l=2}^{d-1} \left(\frac{\gamma}{\beta} \right)^{l-1} \right] \quad (5.5)$$

Therefore, the number of distances between parts computed when the index is used tends to be *independent of the number of shapes in the database*.

In the cases where c_1 is not equal to 1, it is bounded by a constant. The result is that when B is not strictly constant, its deviation from a constant value is amplified in the factor nc_1/b_1 (see simulations in section 6.2). In this case, the number of distances computed remains linear against n , but as shown in the simulations, the slope of the linear relation is very small and the improvement in complexity remains important.

Influence of Compatibility Tests: Formula (5.4) shows that the cost in terms of distance computations is optimum for a high branching factor on the first level, and a comparatively low and decreasing branching factor for the deeper levels, but using the index introduces another complexity parameter which has to be taken into account: the number of compatibility tests processed between parts. A high branching factor presents indeed the drawback of requiring more compatibility tests between the parts of the current shape description and the parts of the I-Structure nodes. This number does not depend on the number of objects in the database, but on the parameters of the index tree instead. We expect the cost added by compatibility test computations to be negligible against the overall gain in number of costly operations (distance computations) that have to be computed for the retrieval.

In order to validate our assumptions and computations, we performed simulations using different algorithms. The simulation protocol and results are presented in section 6.2.

5.5 Summary

The combination of the indexing method and the retrieval algorithm presented here, used with an adequate shape description model, provide several major properties that make the system described meet our original goals of generic shape recognition:

- *Efficient learning*: the dynamic organization of the database gives the system a very powerful and flexible way of manipulating knowledge, especially with the open-world and adaptability assumptions, which are required for genericity.
- *Efficient recognition*: the memory organization allows the retrieval to be nearly independent of the number of shapes in the database.

6 Experimental Results

In this section, we present examples of symbolic hierarchical shape description, database dynamic creation and updating, and recognition performed on both actual and simulated shapes with different algorithm implementations. We implemented the system in CLOS, which prevented us to consider computing accurate time statistics (garbage collecting introduces random and not negligible noise in the statistics...).

6.1 2-D Shape Hierarchical Descriptions

We illustrate the description process by following the different steps leading to the description of an “F106” shape, obtained from real data. The shape decomposition is shown in Figure 1.

Description Parameters

We describe here the details of the implementation of the description model and of the transition costs processing.

We present in Table 1. the different possible values for the symbolic parameters together with the corresponding transition cost matrix. The transition costs are not used for indexing, and therefore the definition of these functions does not influence the index efficiency. Of course they are critical for the quality of the retrieval.

The cost for the length of axes from two parts is a function of the ratio r equals length of the smallest axis over length of the longest axis; the cost for the angle parameter is a function of the axis angle difference α . Table 2. shows the costs associated with the intervals of the possible values for these variables. Note that the definition of the transition costs for angles determines the degree of variation in the part angles allowed in the recognition process.

Resulting Shape Description

A scheme of the description and the corresponding I-Structure for an “f106” shape have already been presented in Figure 2 and Figure 5 respectively. We have developed two graphical tools to display partial information of shape descriptions. One draws a schematic skeleton, i.e. the part axes and respecting the geometrical structure of the shape, and the other draws the I-Structure hierarchy, showing the connection graph and the local geometry of

parameter	possible values	transition cost matrix
boundaries symmetry	0: parallel 1: non parallel	$\begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix}$
curvature	0: positive 1: straight axis 2: negative	$\begin{bmatrix} 0 & 1 & 2 \\ 1 & 0 & 1 \\ 2 & 1 & 0 \end{bmatrix}$
termination	0: mono-angular 1: pluri-angular	$\begin{bmatrix} 0 & 2 \\ 2 & 0 \end{bmatrix}$
part-superpart connection	0: end to close-end 1: end to close-body 2: end to mid-body 3: end to far-body 4: end to far-end	$\begin{bmatrix} 0 & 1 & 2 & 3 & 4 \\ 1 & 0 & 1 & 2 & 3 \\ 2 & 1 & 0 & 1 & 2 \\ 3 & 2 & 1 & 0 & 1 \\ 4 & 3 & 2 & 1 & 0 \end{bmatrix}$

Table 1. Values and corresponding transition cost matrix for the symbolic parameters.

parameter	associated variable	variable intervals	associated cost
size of part (length of axis)	$r = \frac{\min(l_1, l_2)}{\max(l_1, l_2)}$	$r < 0.1$ $0.1 \leq r < 0.2$ $0.2 \leq r < 0.4$ $0.4 \leq r < 0.6$ $0.6 \leq r$	0 1 2 3 4
angle between part axes	$\alpha = (\alpha_1 - \alpha_2) \equiv \pi$	$\alpha < 0.2$ $0.2 \leq \alpha < 0.6$ $0.6 \leq \alpha < 1.3$ $1.3 \leq \alpha < 1.7$ $1.7 \leq \alpha$	0 1 2 3 4

Table 2. Discrete cost function for numerical parameters.

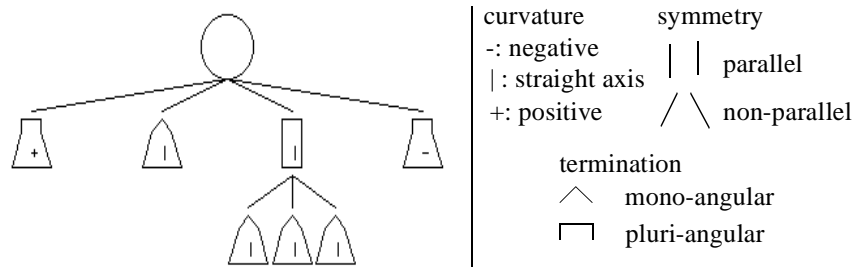


Fig. 10. I-Structure associated with the “F16” shape shown in Figure 8.

the parts. Screen hard copies of these displays for an “F16” shape (Figure 8) are presented in Figure 9 and Figure 10 respectively. It should be clear that the “skeleton” display is not an exact skeleton of the original shape, but a symbolic representation of the shape geometry.

6.2 Shape Recognition

Two issues have to be considered when evaluating a recognition system: the quality of the recognition, and its computational efficiency. In order to evaluate the incremental retrieval method proposed in this paper, we implemented four recognition engines and compared the results obtained with each of them:

- **Engine (1)**’s retrieval algorithm does not use the index. It builds all correspondence hierarchies between the submitted shape and the shapes in the data-bases, and sorts them by decreasing cost. This engine is expected to retrieve the closest shape in the data-base, according to the distance only which means that it may not be the best candidate, in any situation (noise, occlusion, etc.), with the worst computational efficiency.

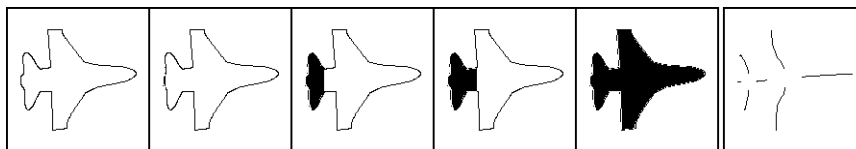


Fig. 8. Decomposition of an “f16” shape (from [17])

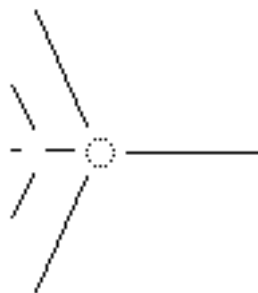


Fig. 9. Symbolic skeleton for the “F16” shape shown in Figure 8

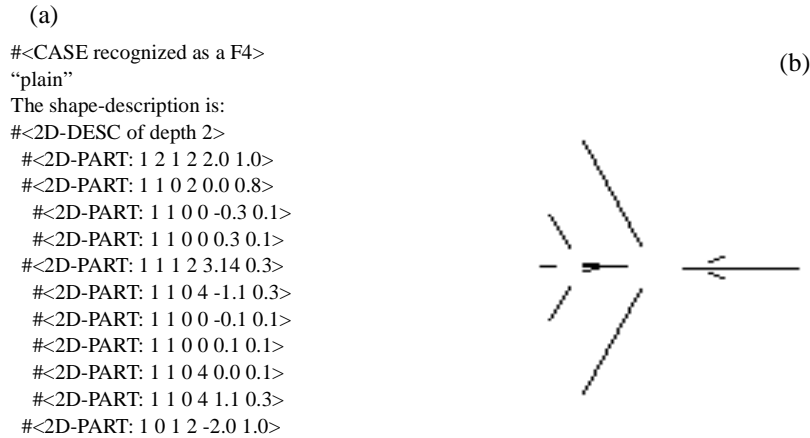


Fig. 11. Case built for an “F4” shape (a) and corresponding symbolic skeleton (b).

- **Engine (2)**’s algorithm uses the index on an exact compatibility selection basis: the deepest node exactly compatible with the submitted shape is found and the correspondence hierarchies are built (down to this node level) between the submitted shape and the shapes indexed by this node and its subnodes. This engine is expected to retrieve an input shape already stored in the database with the best computational cost. If the same exact shape is not in the database, the retrieved shape is not necessarily the closest, and the average complexity depends on the index statistics.

- **Engine (3)** implements the variant of the incremental retrieval algorithm described in section 5.3 in which the one and only exactly compatible node on the first level of the index is selected. This engine is expected to retrieve the best candidate shape with a very good efficiency, but should have problems in the retrieval of badly occluded shapes (*i.e.* when the geometry of the main parts is degenerated).

- **Engine (4)** implements the incremental algorithm described in section 5.3, with a selection ratio for first level index nodes of 75%. This engine is expected to retrieve the best candidate from the case-base with a good computational efficiency.

Quality of the Retrieval

We use a database of planes (such as the ones described above) and several other shapes processed in [17] (beans, human, seven-shape, etc.) to evaluate the quality of the recognition. The database contains a total of 14 shapes. There are 25 nodes in the index (8 on the first level). The evaluation of the quality is mainly performed with engines (3) and (4), to show the importance of the partial-match aspect of the retrieval.

We first propose to the system an “F4” shape already described and included in the data base (see Figure 11). Engines (3) and (4) retrieve the stored version of the shape as the closest shape. In this case we process the retrieval with engines (1) and (2) as well, and obtain the same result. The only difference between the different algorithms for a shape already stored is the efficiency of the retrieval (see efficiency study below).

Then we perform the recognition for two altered descriptions of the same stored “F4” shape to simulate different types of occlusion: in the first case, the geometry of the occluded part is kept, in the other case, the part is completely occluded. The symbolic skeletons of these altered shapes are presented in Figure 12 (a) and (b) respectively. Of course, a strictly exact matching retrieval would end in failure for these cases. Engine (4) retrieves

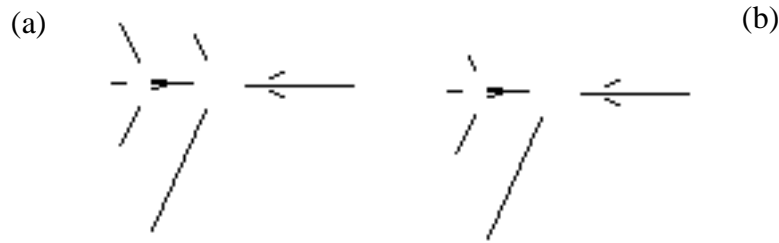


Fig. 12. Ocluded shapes: (a) “good” conditions (the local geometry of the occluded part is not changed); (b) “bad” conditions (the part is totally occluded).

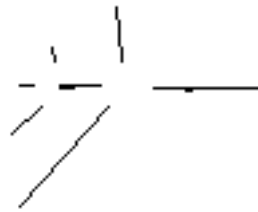


Fig. 13. View-point variation simulation: symbolic skeleton of the altered shape.

the original “F4” stored shape in both cases, pointing out the differences in the correspondence hierarchies. Engine (3) is able to retrieve the original “F4” shape only for the “good” occlusion case (the correspondence hierarchy is of course the same as the one built with engine (4)). No shape can be retrieved for the bad case. Note that if the database was much larger, one or several closest cases could be found and the analysis of the correspondence hierarchy would determine if they are serious candidates. Engine (2) gives also the same result, but the database is not large enough to show the limitations of this algorithm in the case of a shape not exactly identical to a stored shape.

We also process the retrieval with a shape altered to simulate a different view-point. The resulting symbolic skeleton is presented in Figure 13 a. Small parts are missing from self-occlusion, and the axis angles and part sizes are altered. However the main parts (first level of the description hierarchy) are kept, with the same local geometry. Therefore, engines (3) and (4) are able to retrieve the original “F4” shape description and the differences are pointed out in the correspondence hierarchy.

The retrieved closest shape(s) in the case of a completely unknown shape depends on the quality of the database, and the quality of the resulting recognition highly depends on the interpretation of these retrieved cases. Since we did not develop this higher-level aspect of the system, we do not find useful to present such tests. It is however obvious that when an algorithm cannot be used for the retrieval of shapes that are variations of stored shapes, it cannot be used for the retrieval of shapes that have no such direct relation with the stored shapes.

The results presented here show the interest of performing partial matching retrieval (engines (3) and (4)), which allows to deal with noisy and occluded shapes. Engine (4) is able to retrieve a good candidate in any case. Engine (3) can do the same in most cases. The next natural step in our tests is the comparison of their computing performances.

Computing Performances

The Efficiency of the recognition system in terms of computing performance highly depends on the use of the index. To perform tests on large data-bases in the short time we had, we build random shapes as follows:

- *Part generation:* Since the goal of these tests is the evaluation of the indexing performances, the only parameters used are those describing the local geometry. They all take symbolic values. For each parameter, a value is randomly chosen among the possible values (a probability is associated with each value). The other parameters are given arbitrary values.

- Shape generation

- The depth of the description is randomly chosen among 3, 4 and 5 with same probability.
- The number of parts for level 1 is randomly chosen among 2, 3, 4 and 5 with equal probabilities.

Then the corresponding number of parts is generated with a probability of 0.8 for a parallel symmetry (0.2 for non-parallel), 0.2 for a positive curvature, 0.6 for a straight axis (0.2 for negative curvature) and 0.3 for a mono-angular termination (0.7 for pluri-angular).

- For the next level, the number of parts is chosen among 2, 3 and 4 with equal probabilities, and the probabilities for the part generation are 0.2 for a parallel symmetry (0.8 for non-parallel), 0.2 for a positive curvature, 0.6 for a straight axis (0.2 for negative curvature) and 0.8 for a mono-angular termination (0.2 for pluri-angular).

- The distribution of the part connections between two adjacent levels (part and subpart levels) is done randomly: for each part on the part level, a number of subparts is drawn among 1, 2 and 3 with probabilities of 0.6, 0.3 and 0.1 respectively. The corresponding number of parts is taken from the subpart level and the process is reiterated while available subparts remain.

Then cases are built from the random shape descriptions, and collected into case-bases. This generation process allows to build databases which properties are similar to the properties expected for databases of real objects.

Since we are mostly interested in the influence of the number of shapes in the database (n), we build five databases for each value of n in 10, 20, 30, 40, 50, 60, 70, 80, 90, 100, 125, 150, 175, 200, 250, 300, 400, 500, 600, 700, 800, 900, 1000, 1200, 1400 and 1600 shapes. We also build one database for each value of n in 1800, 2400, 4000 and 5000 shapes, being aware that the random generation algorithm is such that the quality of the databases produced deteriorates for large values of n (the random generation cannot provide the variety expected in natural shapes), going against the algorithms using the index. We verify that the average number of nodes on the first level (b_1) is an increasing function n of with a large linear domain, which supports the assumptions made in the theoretical complexity study (see section 5.4).

For our recognition tests, we ran engines (2), (3) and (4) on each shape in each database (*i.e.* for $n=10$, 10 recognition instances were processed with each database, for $n=1000$, 1000 recognition instances were processed with each database). Engine (1) was used the same way on databases whose number of shapes does not exceed 300 because of the time complexity of the process. The retrieval statistics were collected, and the data presented are average values for each value of n .

In order to produce numerical data to evaluate the theoretical complexity computed in section 5.4, we are first interested in the relative behavior of the variable terms in formula (5.2) and in formula (5.4). We therefore have to compare n and nc_1/b_1 . A plot of the average values of c_1 against n for engines (2), (3) and (4) (engine (1) does not use the index) is presented in Figure 14. Of course for engines (2) and (3), it is a constant equal to 1. For engine (4) it is an increasing function of n that is bounded by a constant. We have also studied the term nc_1/b_1

Fig. 14. Average values of c_1 against n for engines (2), (3), (4).

Fig. 16. Average total number of operations computed between parts (distances+compatibility tests) with engines (1), (2), (3) and (4).

7 Conclusion and Perspectives

We have presented:

- A symbolic and structured shape description model which can be used for efficient indexing and retrieval of 2-D or 3-D generic object shapes.
- A dynamic hierarchical indexing method for hierarchical structures, that allows efficient update (used for learning in our case) and is data-driven retrieval oriented.
- A partial matching incremental retrieval method from which efficient algorithms allowing to process noisy, incomplete or totally new shapes can be implemented.

The combination of these features in a recognition system allows performances both in terms of quality and efficiency that are very promising for the development of higher level vision-based reasoning systems.

As previously mentioned, the algorithms we have developed represent the core of a fully integrated high-level recognition system, which leaves interesting work to do. On the input side, the system has to be directly connected with the low-level description process in order to create a single data flow from the image to the recognition hypothesis produced by the retrieval. After a straightforward adaptation of the description model, it can also be connected with a 3-D objects description engine.

Once the closest shape(s) in the database are retrieved, they are used to infer a classification (or identification information) for the new shape. The interpretation of the similarities and differences between the proposed shape and the retrieved closest known shapes (which in our case is an analysis of the correspondence hierarchy) allows to build an answer that can be justified (which is not possible with Neural Networks). This intelligent integration of the data retrieved requires knowledge. In shape recognition, there are two domains of knowledge: the knowledge related to the objects whose shapes are observed, and the meta-knowledge related to the observation process (image processing, description model, etc.). The former type is not to be considered in a generic shape recognition system (such knowledge could be added to customize the system for a specific application), but the latter can be used, for instance as a set of inference rules, without losing genericity (Rule-Based reasoning appears here at a meta-level). In our current implementation, the system simply gives the classe(s) of the closest shape(s) and the decision is left to the human operator (the number of cases proposed for the interpretation is a parameter of the retrieval process).

The CBR paradigm includes validation of the solution proposed, rectification of the solution and of the interpretation process in case of failure, and a possible update of the database (automatic learning). We have not addressed them in the context of shape recognition yet, but a few research directions are currently being investigated. For instance, more elaborate context information, such as functional cross-indexing, could be added to the system and used for validation.

Learning in the sense of storing shapes in the memory is at the core of our system. If we consider automatic learning, the strategy used for database update will affect the parameters of the index (like the branching factor and the load of the index nodes) and therefore should be determined according to the optimum efficiency of the index. For example, storing occluded shapes when a complete instance is already in the case-base is redundant and degrades the performances of the system (more index nodes in deeper levels but no useful discrimination on the first level). On the contrary, replacing several partial descriptions by one complete description, actually observed, would enhance the performances of the system. This however requires a (high) step towards higher-level intelligence in the management of the database (what makes the system discard a set of stored shapes and replace them by a more general occurrence of the same shape? Then why not allow the system to infer 3-D models from

the known shapes and use the models for recognition? This path leads to intelligent learning). Once the recognition itself is complete, the engine can be integrated into a more complex reasoning system to handle complex scenes, for instance using a higher level cross-index describing the relations between the objects in scenes. An identification module could also use the recognition hypothesis for efficient selection of pertinent low-level information.

Finally, we want to point out that the problems addressed in this work and the solutions proposed to solve them are not specific to *our* recognition paradigm. For instance, the hierarchical indexing method together with the hierarchical description method allow to build an exact matching recognition system with the best efficiency that can possibly be expected (constant).

References

1. R. Bareiss. *Exemplar-based knowledge acquisition: a unified approach to concept representation, classification and learning*. Academic Press, 1989.
2. I. Biederman. Human image understanding: recent research and a theory. In *Computer vision, graphics and image understanding*, vol. 32, no. 1, pp. 29-73, October 1985.
3. R.C. Bolles and R.A. Cain. Recognizing and locating partially visible objects: the local-feature-focus method. In *International Journal of Robotics Research*, vol. 1, no. 3, pp. 57-82, 1982.
4. A. Califano and R. Mohan. Multidimensional indexing for recognizing visual shapes. In *Proc. IEEE Computer Vision and Pattern Recognition*, pp. 28-34, Maui, Hawaii, June 1991.
5. G. J. Ettinger. Large hierarchical object recognition using libraries of parametrized model subparts. In *Proc. IEEE Computer Vision and Pattern Recognition*, pp. 32-41, Ann Arbor, Michigan, June 1988.
6. W. E. L. Grimson. *Object recognition by computer - The role of geometric constraints*. MIT Press, Cambridge, Massachusetts, 1990.
7. W. E. L. Grimson and T. Lozano-Perez. Model-based recognition and localization from sparse range or tactile data. In *International Journal of Robotics Research*, vol. 3, no. 3, pp. 3-35, 1984.
8. W. E. L. Grimson and T. Lozano-Perez. Localizing overlapping parts by searching the interpretation tree. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, no. 4, pp. 469-482, 1987.
9. P. Havaldar, G. Medioni and F. Stein. Extraction of groups for recognition. In *Proc. European Conference on Computer Vision*, vol. 1, pp. 251-261, Stockholm, Sweden, May 1994.
10. A. Kalvin, E. Schonberg, J.T. Schwartz and M. Sharir. Two-dimensional, model-based, boundary matching using footprints. In *International Journal of Robotics Research*, vol. 5, no. 4, pp. 38-55, 1986.
11. J. Kolodner. An introduction to Case-Based Reasoning. In *Artificial Intelligence review*, vol. 6, pp. 3-34, 1992.
12. Y. Lamdan and H. J. Wolfson. Geometric hashing: a general and efficient model-based recognition scheme. In *Proceedings of IEEE International Conference on Computer Vision*, pp. 218-249, Tampa, Florida, december 1988.
13. H. Murase and S. K. Nayar. Visual learning and recognition of 3-D objects from appearance. In *International Journal of Computer Vision*, vol. 14, no. 1, pp. 5-24, January 1995.

14. R. Nevatia and Th. O. Binford. Description and recognition of curved objects. In *Artificial Intelligence*, vol. 8, no. 1, pp. 77-98, February 1977.
15. G. Provan, P. Langley and Th. O. Binford. Probabilistic learning of three-dimensional object models. In *Proc. Image Understanding Workshop*, pp. 1403-1413, Palm Springs, California, February 1996.
16. K. Rao. *Shape description from sparse and imperfect data*. PhD Thesis. University of Southern California, December 1988. IRIS Technical Report 250.
17. H. Rom and G. Medioni. Hierarchical decomposition and axial shape description. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*. vol. 15, no. 10, pp. 973-981, October 1993.
18. S. Slade. Case-Based Reasoning: a research paradigm. In *AI Mag.*, pp. 42-55, Spring 1991.
19. F. Stein and G. Medioni. Structural indexing: efficient two dimensional object recognition. In *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pp. 1198-1204, February 1992.
20. M. Zerroug and R. Nevatia. From an intensity image to 3-D segmented descriptions. In *Proc. IEEE International Conference on Pattern Recognition*, vol. 1, pp. 108-113, Jerusalem, Israel, October 1994.