



0031-3203(95)00048-8

## POLYHEDRAL OBJECT RECOGNITION BY INDEXING\*

RADU HORAUD† and HUMBERTO SOSSA  
 LIFIA-IMAG, 46 avenue Félix Viallet, 38031 Grenoble, France

(Received 10 May 1994; in revised form 13 February 1995; accepted for publication 28 March 1995)

**Abstract**—In computer vision, the indexing problem is the problem of recognizing a few objects in a large database of objects while avoiding the help of the classical image-feature-to-object-feature matching paradigm. In this paper we address the problem of recognizing three-dimensional (3-D) polyhedral objects from 2-D images by indexing. Both the objects to be recognized and the images are represented by weighted graphs. The indexing problem is therefore the problem of determining whether a graph extracted from the image is present or absent in a database of model graphs. We introduce a novel method for performing this graph indexing process which is based both on polynomial characterization of binary and weighted graphs and on hashing. We describe in detail this polynomial characterization and then we show how it can be used in the context of polyhedral object recognition. Next we describe a practical recognition-by-indexing system that includes the organization of the database, the representation of polyhedral objects in terms of 2-D characteristic views, the representation of this views in terms of weighted graphs and the associated image processing. Finally, some experimental results allow the evaluation of the system performance.

Object recognition	Polyhedral object representation	Polynomial graph characterization
Indexing	Hashing	Feature extraction.

## 1. INTRODUCTION

The problem of object recognition in computer vision is the problem of matching object features with image features. Nevertheless, since an object has many features associated with it and since an image contains features that do not necessarily belong to that object, the matching process is a complex one because of the large size of the set of image-feature-to-object-feature assignments. Therefore, in the past, the rationale has been to use constraints—such as the rigidity constraint—in order to maintain the number of assignments as reduced as possible: Therefore, any search process, including exhaustive search, is likely to be fast enough because it has to visit a few thousands of nodes rather than millions.

Various implementations of matching-based recognition of rigid objects take the form of either a search-graph or a search-tree. Examples of search graphs are maximal-clique finding algorithms introduced in computer vision by Ambler *et al.*<sup>(1)</sup> popularized by Ballard and Brown,<sup>(2)</sup> and applied to 2-D object recognition by Bolles and Chain.<sup>(3)</sup> Subsequently, the advantage of using trees rather than graphs was stressed by a large number of authors such as Bolles and Horaud,<sup>(4)</sup>

Faugeras and Hebert,<sup>(5)</sup> Ayache and Faugeras,<sup>(6)</sup> Grimson and Lozano-Perez,<sup>(7)</sup> Goad,<sup>(8)</sup> Flynn and Jain,<sup>(9)</sup> and many others.

However, most of the object recognition methods just mentioned restrict the recognition to an object whose exact geometry is known in advance. A more general approach consists of representing both the image and the object by graphs and of casting the recognition problem into the graph matching problem. Graphs are a convenient way of representing features and relationships between these features. Various graph representations have been used by Kim and Kak,<sup>(10)</sup> Flynn and Jain,<sup>(11)</sup> Dickinson *et al.*<sup>(12)</sup> and Bergevin and Levine.<sup>(13)</sup> However, graph matching is a difficult problem in itself. Whenever the two graphs to be matched have the same number of nodes, graph matching is equivalent to searching for graph isomorphism and polynomial time solutions exist in this case.<sup>(14,15,16)</sup> It is, however, rarely the case that the image graph have the same size as the object graph: The problem is therefore equivalent to maximum subgraph matching—find the largest isomorphic subgraphs of the two graphs. So far, solutions proposed for solving the maximum subgraph matching problem involve some form of combinatorial optimization.<sup>(17,18)</sup>

If many objects rather than a single one (as it has often been the case) are present in a database of objects to be recognized, then the matching-based recognition becomes intractable because the complexity grows substantially with the number of features. An indexing process is crucial whenever the recognition process has to select among many possible objects. *Recognition by*

\* This work has been supported by the Esprit programme through the SECOND project (Esprit-BRA No. 6769) and by the ORASIS project (PRC Communication homme/ machine).

† All correspondence should be addressed to Radu. Horaud@image.fr; fax: +33 76 57 46 02.

*indexing* is therefore the process that attempts to rapidly extract from a large list of objects those few objects that fit a group of image features, while avoiding to establish image-feature-to-object-feature assignments.

Nevatia and Binford<sup>(19)</sup> were among the first to describe indexing as a part of an object recognition system. Ettinger<sup>(20)</sup> described a hierarchically organized object library well suited for indexing. Each object is decomposed into a list of subparts. The rationale is that many objects share a common set of subparts and what distinguishes one object from another is subpart relationships—the overall list of subparts grows *sub-linearly* with the number of objects in the library. This idea is applied to flat objects that are described by their outlines.

The idea of using hashing in conjunction with object recognition was introduced by Kalvin *et al.*<sup>(21)</sup> Outlines of flat objects are described in terms of *footprints*. The best way to think of a footprint is of an intrinsic curve such as a curvature as a function of curvilinear abscissa. The footprint of an object is further decomposed into intervals. Each such interval is described by a set of numbers (the sine and cosine Fourier coefficients, for example) and these numbers are hashed in hash-tables. The indexing itself takes the form of a vote: Each footprint interval detected in the image votes for those objects in the database containing this footprint interval. Finally, the object that received the highest vote score is the recognized object. A variation of this method using local frames and *geometric* hashing was proposed by Lamdan and Wolfson<sup>(22)</sup> for solving the matching problem, not the indexing problem.

Following the same idea of hashing, Stein and Medioni<sup>(23)</sup> were able to recognize 3-D objects from 3-D data using super-segments and surface-patches as features. Their *structural* hashing technique retrieves object hypotheses from the database using hash-table indexing. A similar approach was proposed by Breuel.<sup>(24)</sup>

The approach advocated in this paper capitalizes onto the representation of 3-D objects in terms of 2-D characteristic views and of characterizing such a view with a number of weighted graphs. An identical graph representation is extracted from the image of an unknown 3-D scene. A polynomial characterization of graphs allows us to organize the “model graphs” into hash-tables. Therefore, recognition consists of computing similar polynomial characterizations for the “image graphs” and of indexing in the pre-stored hash-tables. Finally, a voting process ranks a number of candidate characteristic views as potential recognized objects.

### 1.1. Paper organization

The remainder of this paper is organized as follows. Section 2 introduces the polynomial characterization of a binary graph that will be used, namely the second immanantal polynomial of the Laplacian matrix of

a graph. Then we briefly describe an extension of this representation to weighted graphs. Section 3 describes the graph indexing method that uses this polynomial characterization of graphs. It also describes an object representation in terms of weighted graphs, the organization of the database of objects to be recognized, and the indexing method itself which is based on hashing. Section 4 describes a representation of 3-D polyhedral objects in terms of 2-D views and a representation of these views in terms of weighted graphs. Section 5 describes how to extract these weighted graphs from images and how to remove irrelevant image data. Section 6 describes a recognition experiment carried out with a set of nine images of the same scene in the presence of a database of six objects. Finally, Section 7 draws some conclusions and gives some directions for future work.

## 2. POLYNOMIAL CHARACTERIZATION OF A GRAPH

The method that we propose in this paper in order to achieve indexing uses graphs to represent both images and objects. Let us suppose that one is able to extract a number of graphs from an image and let  $G_1$  be such an image graph that has the same number of nodes as a graph  $G_2$  extracted from an object. Such a graph (an image or an object graph) is defined by a set of vertices  $V$  and a set of edges  $E$ . The two graphs  $G_1 = (V_1, E_1)$  and  $G_2 = (V_2, E_2)$  are said to be isomorphic if there is a bijection  $\phi = V_1 \rightarrow V_2$  such that:

$$(v_1, v_2) \in E_1 \quad \text{if and only if} \quad (\phi(v_1), \phi(v_2)) \in E_2.$$

If  $A_1$  and  $A_2$  are the adjacency matrices of the two graphs, one can easily see that  $G_1$  is isomorphic to  $G_2$  if and only if there exists a permutation matrix  $P$  satisfying:

$$A_2 = PA_1P^{-1}. \quad (1)$$

Hence, there are two ways to decide whether two graphs are isomorphic:

(1) Find the permutation matrix  $P$  that satisfies the equation above.

(2) Find an algebraic characterization of the adjacency matrix of a graph that is *invariant* under a similarity transformation of the adjacency matrix. Such a characterization has been proved to be useful for graph classification.

One obvious characterization that is invariant under similarity is the characteristic polynomial associated with the adjacency matrix.<sup>(25,26)</sup> Indeed we have:

$$\begin{aligned} \det(xI - PAP^{-1}) &= \det(PxIP^{-1} - PAP^{-1}) \\ &= \det(P(xI - A)P^{-1}) \\ &= \det(xI - A). \end{aligned}$$

Therefore, the similarity of adjacency matrices is a necessary condition for isomorphism. Unfortunately it is far from being a sufficient condition. However, an important idea stems out from this example of graph characterization—one may seem to characterize

a graph, up to an isomorphism, by the coefficients of a polynomial associated with that graph. More formally, we seek a polynomial associated with a graph, say  $p(G)$  such that:

$$\begin{cases} \text{if } G_1 = G_2 \text{ then } p(G_1) = p(G_2) \\ \text{and} \\ \text{if } p(G_1) = p(G_2) \text{ then } G_1 = G_2. \end{cases} \quad (2)$$

Two graphs are said to be equal if they have the same number of nodes and if they are isomorphic. Two polynomials are equal if they have the same degree and if their coefficients are equal. If a polynomial satisfying the above condition exists, it follows that the problem of comparing two graphs of the same size is equivalent to the problem of comparing the coefficients of their associated polynomials. Notice, however, that graph characterization with a polynomial allows one to state whether two graphs are isomorphic or not, but it does not provide the node-to-node isomorphic mapping between the graphs. Graph characterization is, therefore, exactly what one needs for model indexing, i.e. rapidly state whether some sensed data equal some object data. The search of an isomorphic node-to-node mapping is the task of matching and not the task of indexing.

Polynomials which characterize a graph unambiguously up to an isomorphism have been thoroughly studied in the linear algebra literature.<sup>(26,27)</sup> Among these polynomials, the *second immanantal polynomial*—or the  $d_2$ -polynomial—is a good candidate.<sup>(28)</sup>

One may associate the second immanantal polynomial with the adjacency matrix of a graph. There are, however, some reasons to prefer the Laplacian matrix (defined below) to the computationally simpler adjacency matrix. The Laplacian matrix is a positive semidefinite symmetric of rank  $n - 1$  (if  $G$  is a connected graph). Generally speaking, second immanantal polynomials match up well with positive semidefinite matrices. The greater complexity of Laplacian matrices, when compared with adjacent matrices, suggests there may be fewer *algebraic accidents*.<sup>(28)</sup> If the time to compute the determinant of an  $n \times n$  matrix is  $n^3$ , the time to compute the coefficients of the second immanantal polynomial is  $n^4$ .

The elements of the Laplacian matrix of a *binary graph*,  $L(G)$ , are defined as follows:

$$l_{ij} = \begin{cases} d_i & \text{if } i = j \\ -1 & \text{if there is an edge between nodes } i \text{ and } j \\ 0 & \text{otherwise,} \end{cases} \quad (3)$$

where  $d_i$  is the number of graph edges meeting at node  $i$  and is called the degree of the node  $i$ . The interested reader may find in reference (29) a complete description of the properties of the Laplacian matrix of a binary graph.

The second immanantal polynomial associated with an  $n \times n$  Laplacian matrix of a graph,  $L(G)$ , can be

written in generic form as:

$$d_2(xI - L(G)) = c_0(L(G))x^n - c_1(L(G))x^{n-1} + \dots + (-1)^n c_n(L(G)). \quad (4)$$

The coefficients  $c_0, \dots, c_n$  of this polynomial are integers and they can be computed using the following formulae, which are detailed by Merris<sup>(28)</sup> ( $n$  is the number of nodes of the graph and  $m$  is the number of edges of the graph):

$$\begin{cases} c_0(L(G)) = n - 1 \\ c_1(L(G)) = 2m(n - 1) \\ \vdots \\ c_k(L(G)) = \sum_{X \in Q_{k,n}} (\sum_{i=1}^k l_{ii} \det(L(G)\{X\}(i)) - \det(L(G)\{X\})). \end{cases} \quad (5)$$

In these formulae  $l_{ii}$  denotes a diagonal term of  $L(G)$  and  $Q_{k,n}$  denotes the set of all the  $C_n^k$  strictly increasing sequences of size  $k$  ( $2 \leq k \leq n$ ) obtained from the set  $\{1, 2, \dots, n\}$ . For any  $n \times n$  matrix  $M$  and for  $X \in Q_{k,n}$  let  $M\{X\}$  be the  $k \times k$  principal submatrix of  $M$  corresponding to  $X$ .  $M\{X\}$  is the  $n \times n$  matrix:

$$M\{X\} = \begin{pmatrix} M[X] & 0_k \\ 0_k & I_{n-k} \end{pmatrix}, \quad (6)$$

where  $I_{n-k}$  is the identity matrix of size  $n - k$  and  $0_k$  is the null matrix of size  $k$ .  $M\{X\}(i)$  is the matrix obtained from  $M\{X\}$  by removing the  $i$ th row and the  $i$ th column.

An important property of the second immanantal polynomial associated with a graph is that it is preserved under the similarity permutation:<sup>(27)</sup>

$$d_2(xI - L(G)) = d_2(xI - PL(G)P^{-1}).$$

Therefore, a necessary condition for two graphs to be isomorphic is that they have the same second immanantal polynomial. However, it is not sufficient condition. In practice, however, there have been found very few examples of non-isomorphic graphs that have the same second immanantal polynomial.<sup>(25)</sup>

### 2.1. An example

In order to illustrate the above formalism let us consider two simple binary graphs and let us outline the computation of their associated second immanantal polynomials. An example of two four-node binary graphs are shown on Fig. 1. The Laplacian matrices are given by equation (3) and they are easy to compute:

$$L(G)_1 = \begin{pmatrix} 3 & -1 & -1 & -1 \\ -1 & 1 & 0 & 0 \\ -1 & 0 & 1 & 0 \\ -1 & 0 & 0 & 1 \end{pmatrix}$$

$$L(G)_2 = \begin{pmatrix} 2 & 0 & -1 & -1 \\ 0 & 1 & -1 & 0 \\ -1 & -1 & 3 & -1 \\ -1 & 0 & -1 & 2 \end{pmatrix}$$

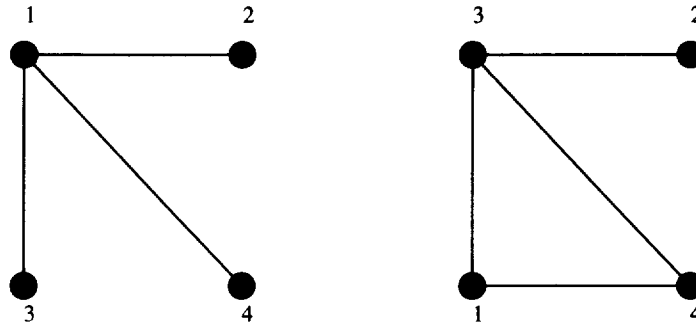


Fig. 1. These two graphs differ by one edge, but their associated second immanantal polynomials are quite different.

For  $n = 4$  the sets of all the  $C_4^k$  strictly increasing sequences of size  $k$  ( $2 \leq k \leq 4$ ) are:

$$Q_{2,4} = \{(1, 2), (1, 3), (1, 4), (2, 3), (2, 4), (3, 4)\}$$

$$Q_{3,4} = \{(1, 2, 3), (1, 2, 4), (1, 3, 4), (2, 3, 4)\}$$

$$Q_{4,4} = \{(1, 2, 3, 4)\}.$$

It is straightforward to compute the matrices

2.2. *Weighted graphs*

In general, binary graphs are not sufficient for describing the structure of either images or objects. Weighted graphs are graphs which have a weight  $w_{ij}$  associated with the edge linking nodes  $i$  and  $j$ . The definition of the Laplacian matrix may easily be extended to weighted graphs, as follows:

$$l_{ij}^w = \begin{cases} D_i & \text{if } i = j \\ -w_{ij} & \text{if there is a weighted edge between nodes, } i \text{ and } j \\ 0 & \text{if there is no edge between nodes } i \text{ and } j, \end{cases} \tag{7}$$

$L(G)\{X\}$  and the matrices  $L(G)\{X\}(i)$ , for  $X \in Q_{k,4}$ . For example,  $L(G)_1\{(1, 2)\}$  is a  $4 \times 4$  matrix obtained by appending the first two rows and first two columns of  $L(G)_1$  with  $I_2$  and  $0_2$  as follows:

$$L(G)_1\{(1, 2)\} = \begin{pmatrix} 3 & -1 & 0 & 0 \\ -1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{pmatrix}.$$

$L(G)_1\{(1, 2)\}(2)$  is a  $3 \times 3$  matrix obtained from  $L(G)_1\{(1, 2)\}$  by removing its second row and second column:

$$L(G)_1\{(1, 2)\}(2) = \begin{pmatrix} 3 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{pmatrix}.$$

After some straightforward computation we obtain the following coefficients for the associated polynomials:

$$d_2(xI - L(G)_1) = 3x^4 - 18x^3 + 33x^2 - 24x + 6$$

$$d_2(xI - L(G)_2) = 3x^4 - 24x^3 + 105x^2 - 68x + 24.$$

One may also compute the characteristic polynomials associated with the Laplacian matrices, i.e.

$$\det(xI - L(G)_1) = x^4 - 6x^3 + 9x^2 - 4x$$

$$\det(xI - L(G)_2) = x^4 - 8x^3 + 19x^2 - 12x.$$

From this example it is obvious that the second immanantal polynomial is a richer graph description than just the characteristic polynomial.

with  $D_i$  being equal to the sum of the weights of the edges meeting at the node  $i$ :

$$D_i = \sum_{j=1}^n w_{ij}. \tag{8}$$

This matrix has the same properties as the Laplacian matrix associated with a binary graph—it is symmetric semidefinite positive and of rank  $n - 1$ , which makes it suitable for computing the  $d_2$ -polynomial.

3. GRAPH INDEXING

The graph characterization in terms of the coefficients of the second immanantal polynomial allows one to assert whether two graphs with the same number of nodes ( $n$ ) are “equal”. The difference between two graphs  $G_1$  and  $G_2$  is given by the formula:

$$\text{Diff}(G_1, G_2) = \sum_{k=1}^n (c_k^1 - c_k^2)^2, \tag{9}$$

where  $c_k^i$  is the  $k$ th coefficient of the second immanantal polynomial associated with graph  $i$ . Since we assume that the above equation is valid only for graphs with the same number of nodes,  $c_0$  has been skipped out from the summation.

In the case of indexing we are faced with the problem of comparing an image graph with many database graphs and of deciding which are the few graphs in the database that are equal to the image graph. In that

case, the graph difference mentioned above is not efficient. One way to implement indexing efficiently is to use *hashing*.<sup>(30)</sup> Hashing can be briefly described as follows. Each database object has a numerical *key* associated with it. Then a *hash-function* maps this key onto the address of an array of a manageable size. The address thus computed for an object is also called the *hash-code* of that object. In practice, hashing is composed of an off-line step (database construction) and a runtime step (indexing):

- *Database construction* consists of computing a hash-code for each object to be stored in the database. Several objects may well have the same address (hash-code). Therefore, a list of objects will be associated with each address. The database therefore takes the form of an array (or a hash-table), a list of objects being stored at each array-element address.

- *Indexing* consists of computing the address (hash-code) of an unknown object in order to determine whether this object is in the hash-table or not.

Since a graph may be described by the integer coefficients of a polynomial, these coefficients may well be viewed as the hash-codes of the graph. Hence, a graph with  $n$  nodes can be mapped onto  $n$  hash-tables. For reasons that will be made clear below, the size of the graphs we deal with varies between five nodes and 10 nodes. Within this size range the second immanantal polynomial uniquely characterizes binary and weighted graphs. It follows that graph indexing will become an efficient technique because the hashing will have very few collisions associated with it.

Polynomial characterization of graphs combined with graph indexing will eventually allow us to perform object recognition by indexing. However, two important issues need be raised before we describe a practical object recognition system: object representation and database organization.

### 3.1. Object representation

Object representation has been thoroughly studied in Computer Vision and a recent paper by Flynn and Jain<sup>(11)</sup> provides a good state of the art. In general, there are two possible representation classes: object-frame centered and viewer-frame centered representations. Within our approach we use a representation that is not tight to a specific coordinate frame. An object is mainly described by a list of characteristic views. In the representation that we use the definition of a characteristic view (CV) should be understood in a broad sense. It is a network of object features and feature relationships that are simultaneously visible from some viewpoint. Such a representation is by no means limited to the aspect graph representation of an object. The features in the network may well be either 2-D or 3-D, object-centered or viewer-centered. The important characteristic here is not as much the dimensionality of the features or the coordinate frame to

which they relate, but instead the *intrinsic* properties of the feature network. As we have already mentioned, such a feature network can be conveniently represented by a weighted graph.

However, the data associated with some view of an object rarely encodes a whole characteristic view associated with that object. The data are corrupted by noise, occlusions, self occlusions, and accidental alignments. Therefore, it will not be very useful to directly store in the database the graph associated with a characteristic view. Instead, each characteristic view is further decomposed in a number of, possibly overlapping, "smaller" views or subviews, where each such subview is in fact associated with a subgraph of the graph describing the characteristic view. There are several reasons in support of the decomposition of a characteristic view into a number of subviews:

- Following the results of Section 2, one can compare only graphs with the same number of nodes. Since a graph extracted from the data has rarely the same number of nodes as the graphs associated with the characteristic views of the objects to be recognized, one may still attempt to compare an unknown-object-view with a characteristic-view by comparing subgraphs associated with subviews of these views.

- The cost of the computation of the coefficients of the second immanantal polynomial is proportional to  $n^4$ , where  $n$  is the number of nodes of the graph. Since "raw" characteristic views may have a large number of features associated with them, it may not be efficient to compute polynomial characterizations for very large graphs.

- Consider a data graph that is composed of a large network of features. It is very unlikely that such a large data graph belongs to a unique object. Recognition based on such large graphs will fail because these graphs are not present in the database.

It is therefore clear that object recognition by indexing must adopt a representation such that each characteristic view is decomposed in a number of subviews or subgraphs. A compromise must be made concerning the size of these subgraphs: Very small subgraphs are too ambiguous because they do not capture information that is object specific while large subgraphs are difficult to extract from the data. At the limit, a single-node graph belongs to all the objects in the database and indexing is useless in this case. At the other extreme, a very large graph encapsulates more than one object and the indexing process will fail to find this graph in the database.

### 3.2. Database organization and the indexing mechanism

Following the above discussion, the organization of the database follows a three-layer structure. A first layer contains a list of graphs of various sizes that are organized in hash-tables. The second layer contains characteristic views. A third layer contains descrip-

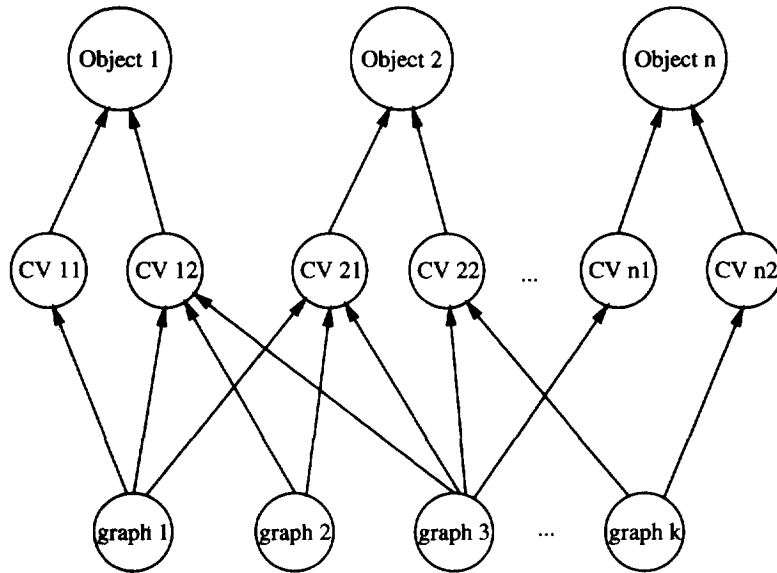


Fig. 2. The database has a three-layer structure: graphs, characteristic views and objects. An object may well have more than two characteristic views associated with it.

tions of the object themselves. This structure is shown in Fig. 2.

It is clear that a graph belongs to several characteristic views and hence it may belong to several objects. Therefore, an image graph that matches a graph in the database provides handles to more than one object. The interesting feature of this three-layer organization is that the graph list grows sublinearly with the number of characteristic views.

The indexing mechanism proceeds as follows. Let us suppose that an unknown object view has to be recognized. First, this unknown view is decomposed into subviews and a graph is associated with each subview. Polynomial characterizations are computed for these unknown graphs. Based on these characterizations and on the hashing technique just described, each unknown graph is assigned a **unique** graph in the database. As a consequence, a list of characteristic views may now be associated with each unknown graph in the image. In other terms, each unknown graph *votes* for a number of characteristic views. This process is repeated for each unknown graph belonging to the unknown view. The characteristic view that received the largest number of votes is the model that best matches the unknown view.

Consider, for example the database depicted in Fig. 2 and suppose that two unknown graphs are assigned **graph 1** and **graph 2**, respectively. It follows that two characteristic views (CV12 and CV21) received two votes while one characteristic view (CV11) received only one vote.

#### 4. 2-D CHARACTERISTIC VIEWS OF POLYHEDRAL OBJECTS

In order to recognize an object with the method described above, one has to represent that object in

terms of a few characteristic views and to describe each such characteristic view in terms of a set of weighted graphs. Unlike solutions that consist of computing characteristic views from CAD object descriptions, our approach for obtaining these views is to gather as many images of an object as characteristic views are needed for describing that object unambiguously. Although in this paper we use an *ad hoc* technique, more formal methods may be found in reference (31, 32): The authors define a set of characteristic views of a polyhedral object by partitioning a large set of object views into small sets of characteristic views.

The task of decomposing a characteristic view into subgraphs is not an easy one. The most general approach would be to implement a graph decomposition method. For example, one may attempt to partition a graph into a pre-specified number of subgraphs such that the number of connections (edges) between the subgraphs is minimized.<sup>(33)</sup> Here we prefer a more pragmatic solution. For example, one may consider all the nodes of a characteristic view and form subgraphs around each such node. A subgraph is thus formed by this node as well as the nodes that are at a distance less or equal to  $p$  edges away from this node. It turns out that this redundant decomposition of a characteristic view in small graphs of various sizes is one key to the success of our recognition method. Indeed, small perturbations in the topology of a view (due essentially to noise or to segmentation errors) will not affect the topology of *all* the subgraphs extracted from this view.

In the particular case of polyhedral objects, if the degree of a node is, on an average, equal to 3 and for  $P \leq 2$ , then the number of nodes of the associated subgraphs varies between 5 and 10. Figure 3 shows a characteristic view of a simple polyhedral object and some subgraphs extracted from this view with  $P = 2$ .

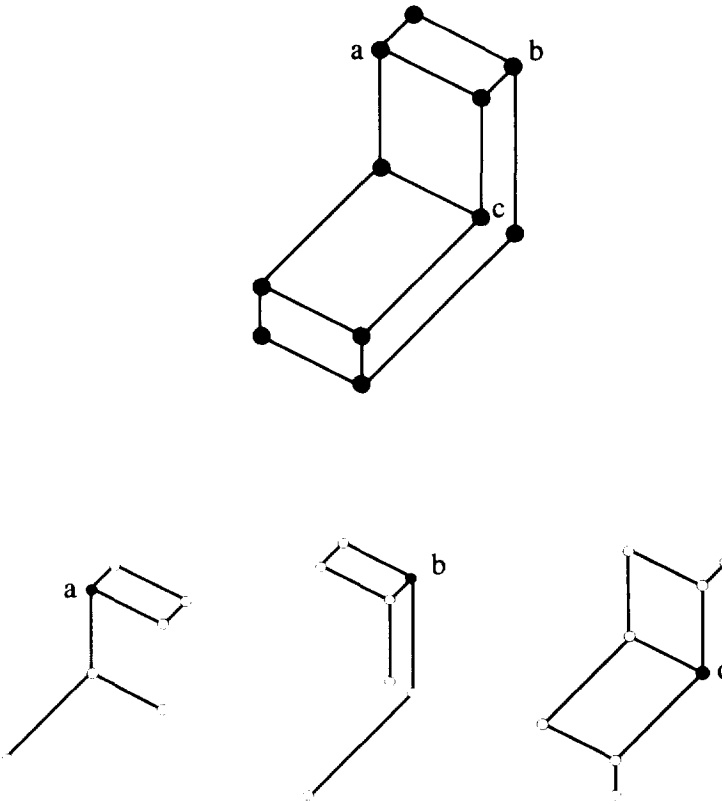


Fig. 3. An example of a characteristic view of an object and a few graphs extracted from this view.

As already discussed, the topology of a characteristic view is not sufficient for describing the view unambiguously. For example, Fig. 4 shows six different binary graphs which are isomorphic (they have the same topology). Clearly, one would like to be able to state that the top three graphs are different and the bottom three are identical. In other words, the top three do not look the same, although they have the same topology. The question of how to describe the 2-D appearance of a polyhedral object has already been addressed [see, for example reference (34)], but the question of how to represent such an appearance with a weighted graph has not.

One way to label an edge is to characterize it according to the structure of the vertices at each endpoint of that edge. If we consider polyhedra that have at most three edges meeting at a vertex, then we obtain a catalogue of possible edge structures or *edge appearances*. It is sufficient to assign labels to these various appearances and to associate a weight to each label. Figure 5 shows an exhaustive catalogue of edge appearances and their weights.

There are three possible vertex structures: a two-edge vertex or an "L", and two three-edge vertices, an "Y" and an "Arrow". Since an edge divides the plane into two regions—the left side and the right side—we obtain the following list of features that allows the labelling and the weighting of an edge (see Fig. 6 for an example of an edge labelled "15"):

- the type of the first vertex (**Arrow**);
- the type of the second vertex (**Arrow**);
- the number of edges associated with the first vertex lying on the left side of the edge (1);
- the number of edges associated with the first vertex lying on the right side of the edge (1);
- the number of edges associated with the second vertex lying on the left side of the edge (2);
- the number of edges associated with the second vertex lying on the right side of the edge (0).

## 5. IMAGE PROCESSING

In this section we describe the process by which a number of graphs is extracted from an image. This graph extraction process has some similarities with feature grouping since its goal is to provide a few "key" image features and reduce the complexity of the object recognition process. Image processing starts with extracting edges and with approximating these edges with straight lines. Junctions are next extracted. The junctions and the straight lines form a network of features, or a graph—the image graph.

If the scene is composed of just one object, then this image graph corresponds, up to some noise, to a view of that object. Single object scenes are used for building the database of characteristic views and it has been previously described.

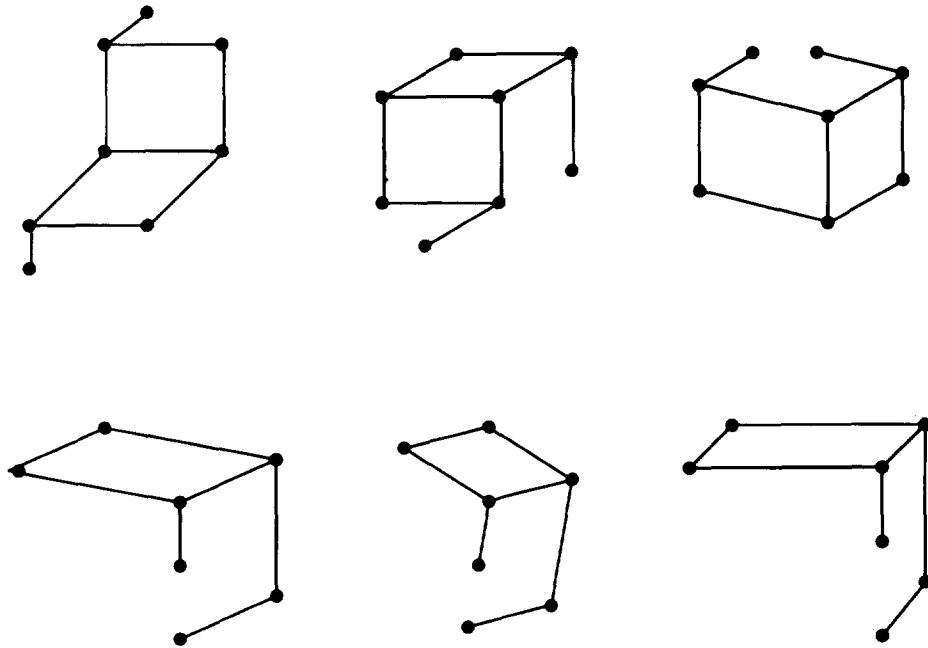


Fig. 4. Top—three graphs having the same topology. Bottom—three other graphs having the same topology and the same appearance.

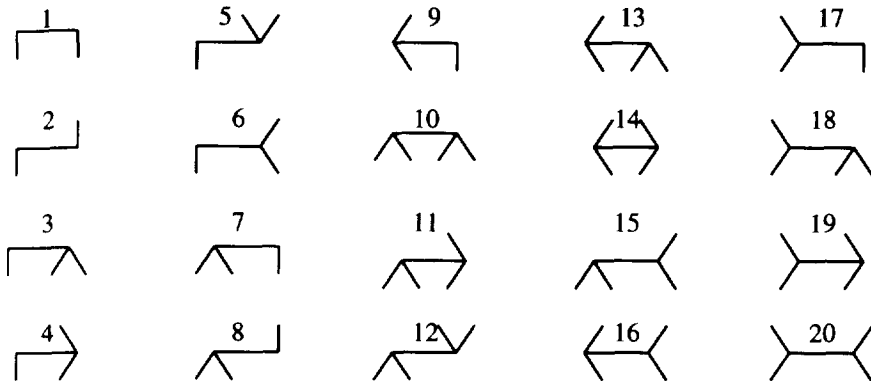


Fig. 5. An exhaustive catalogue of the possible appearances of the edges of a polyhedral object that has, at most, three edges meeting at a vertex.

If the scene is composed of more than one object and if the background is not uniform, then the image graph has to be further processed in order to be split into smaller graphs. Each small graph thus obtained is examined in order to decide whether it should be considered for recognition or not. To summarize, the process of extracting graphs from an image comprises the following steps:

- (1) image graph extraction;
- (2) image graph splitting;
- (3) graph evaluation.

*Step 1.* Image-graph extraction has been briefly outlined at the beginning of this section and is described in detail in reference (35).

*Step 2.* Image-graph splitting is based on a number of heuristics:

2.1 Isolated and “dangling” edges are thrown out.

2.2 It is assumed that “T” junctions arise from occlusions (an object in front of another object, an object in front of some background, or a self occlusion). Hence, the image graph is cut off at T junctions. Notice that this process may produce isolated edges which are immediately thrown out.

2.3 Sequences of collinear edges are assumed to arise from the same physical edge and hence, collinear edges are fused into a unique edge.

2.4 Finally, the image graph is decomposed into connected components.

*Step 3.* Graph evaluation considers each connected component, one by one, and evaluates it in order to decide whether it should be further considered for recognition or thrown out. Let  $n$  be the number of

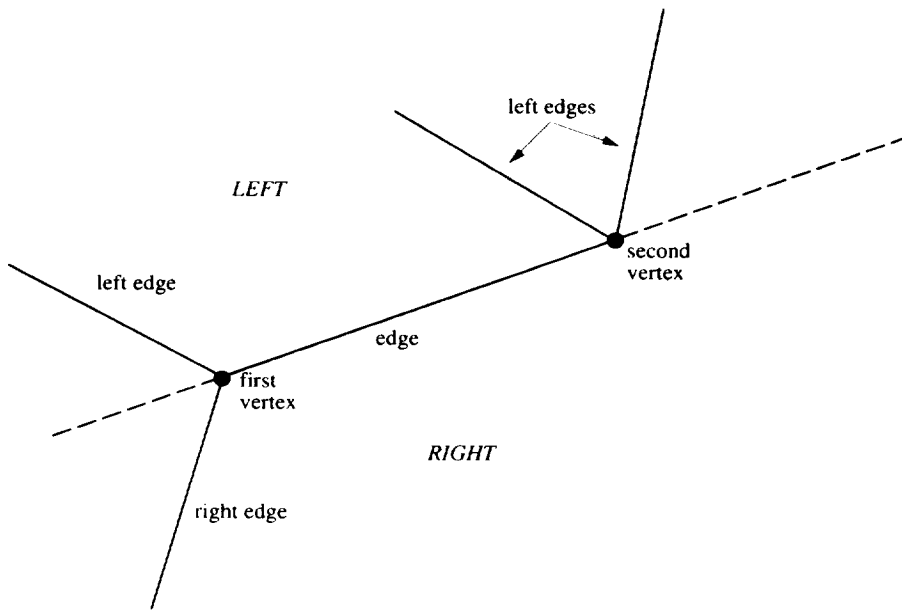


Fig. 6. The labelling of an edge depends on the structure of the two vertices at each endpoint of that edge (see text).

nodes of a graph and let  $d_i$  be the degree of node  $i$ , i.e. equation (3). The quality:

$$f(G) = \frac{1}{n} \sum_{i=1}^n d_i$$

allows one to measure the complexity of a graph. It is straightforward to notice that for  $f(G) = 2$ , the graph has at most one cycle. Since graphs without cycles are not really relevant, one may consider only graphs for which:

$$f(G) \geq 2.$$

The graphs that do not satisfy this constraint are considered irrelevant and therefore they are thrown out.

Let us illustrate with an example the graph extraction process that we just described. Figure 7 shows an intensity image (top-left) and a network of lines and junctions extracted from this image (top-right) from which isolated and dangling lines are removed (middle-left). The next image (middle-right) shows the T-junctions that are removed from the list of junctions. This T-junction removal process produces isolated lines and dangling lines on one hand (which are removed) and collinear lines which are fused into a unique line on the other hand (bottom-left). We are left now with a number of connected image graphs. Each such connected component is evaluated according to Step 3. The latter process leaves four connected components in the image (bottom-right). These remaining image graphs will be further decomposed into subgraphs in order to be used by the indexing process. The decomposition of the four image graphs into subgraphs is not shown.

## 6. EXPERIMENTS

All the object recognition experiments that we performed used the same database, namely 20 characteristic views associated with six objects, as shown in Figs 8 and 9. All the objects in this database are 3-D polyhedral shapes with one exception. The database was built by showing each object, view-by-view, to the camera and by applying the graph extraction process described in Section 5.

We carried out many experiments in which the input image varied from a very simple one with just one object against a uniform background to more complex images with many objects against a nonuniform background.

In one such experiment we grabbed nine images of the same scene and we processed these images identically (with exactly the same segmentation parameters). Figure 10 shows these nine images where the camera position and orientation varies with respect to the observed scene. Figure 11 shows the graphs extracted from these images. The images are numbered 1-9 from left to right and top to bottom.

Table 1 summarizes the results of recognizing the two objects based on the two graphs (labelled "0" and "1") extracted from the nine images. The figures in this table correspond to the scores (number of votes) received by each characteristic view when the image graphs are indexing the database. For each image graph the table records its highest score, sometimes the two highest scores. The first object (the graph labelled "0") has been correctly recognized seven times and incorrectly recognized twice (images 5 and 9). Notice the high scores (between 16 and 24) obtained in the

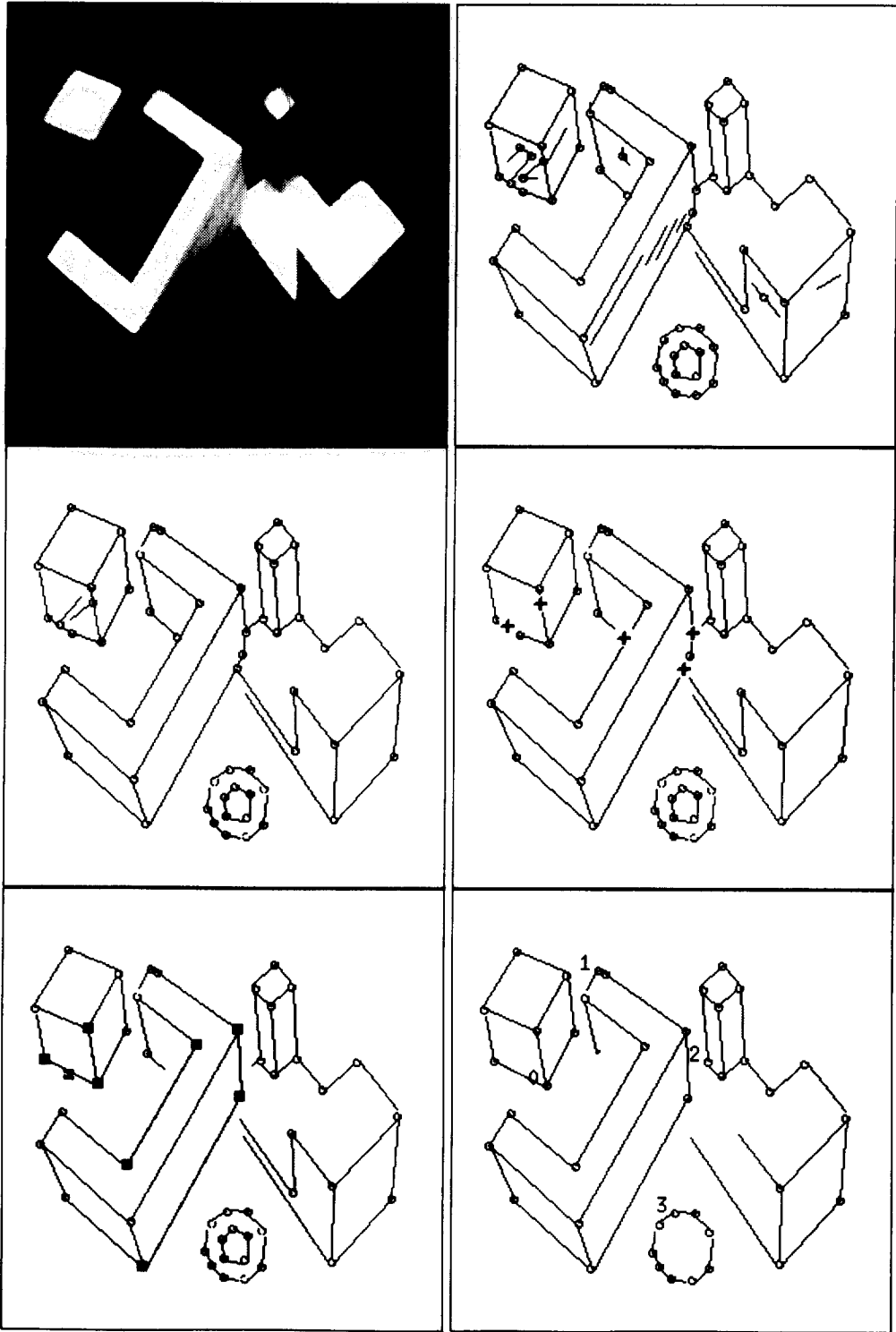


Fig. 7. An example of extracting a set of four image graphs from an intensity image (top-left): Lines and junctions are detected (top-right), isolated and dangling lines are removed (middle-left), T-junctions are detected and removed (middle-right), collinear lines are fused into longer lines (top-left), the remaining graphs are evaluated and four of them survive (bottom-right).

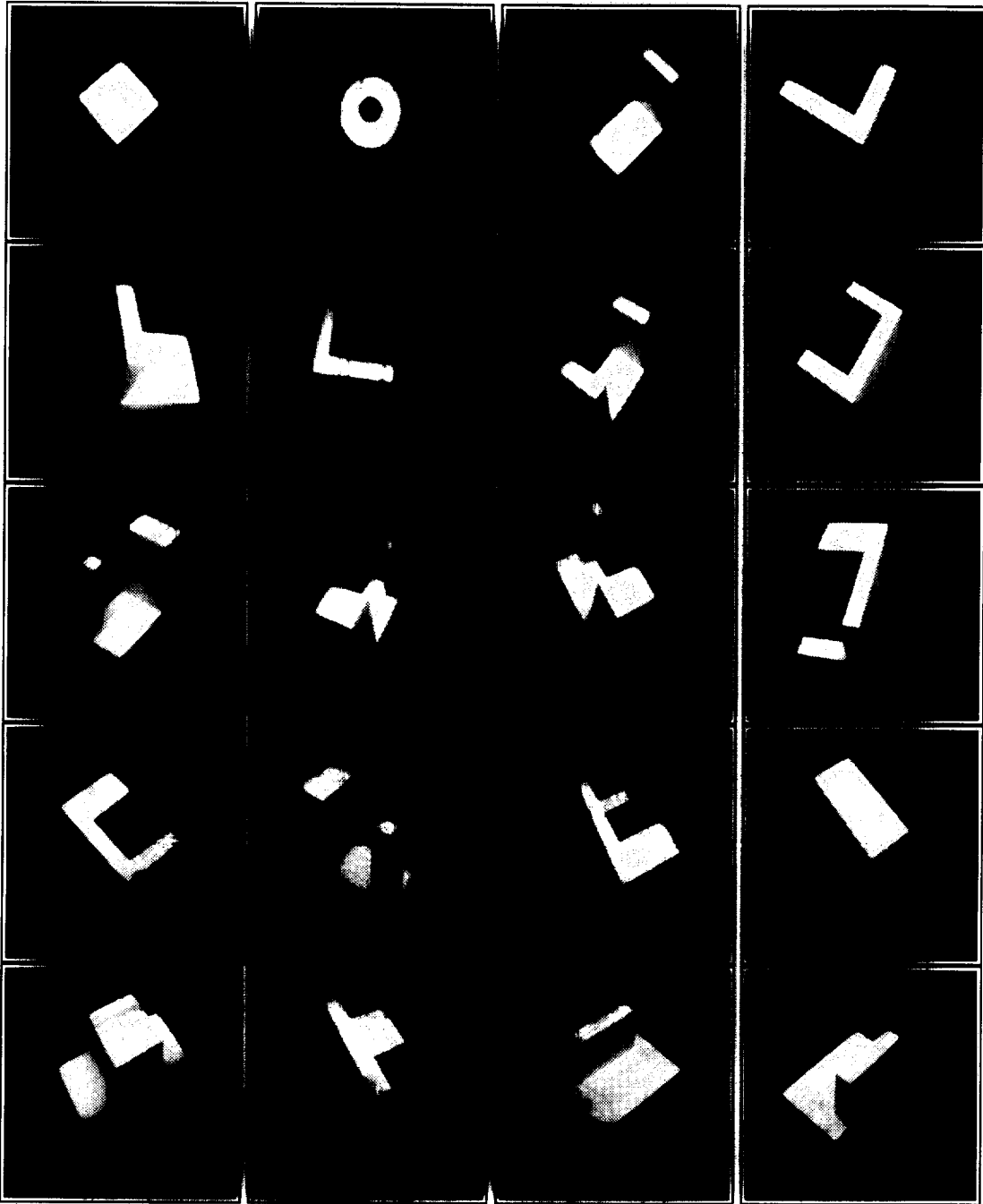


Fig. 8. The intensity images of 20 characteristic views associated with six objects. These views and objects constitute the database.

case of a correct recognition in comparison with the less high scores (between 6 and 8) obtained in the case of an incorrect recognition.

The same phenomena can be observed with the second object (the image graph labelled "1"), which has been correctly recognized six times (the score varies between 8 and 25) and incorrectly recognized three times (the score varies between 4 and 7). An interesting

remark is that all five incorrect recognition assigned **the same** characteristic view to the unknown image graphs, namely the view labelled "pieceLd02m" (the bottom-rightmost view in Fig. 9). One may notice the poor segmentation associated with this characteristic view.

The recognition results reported above are barely affected if one increased the size of the database of

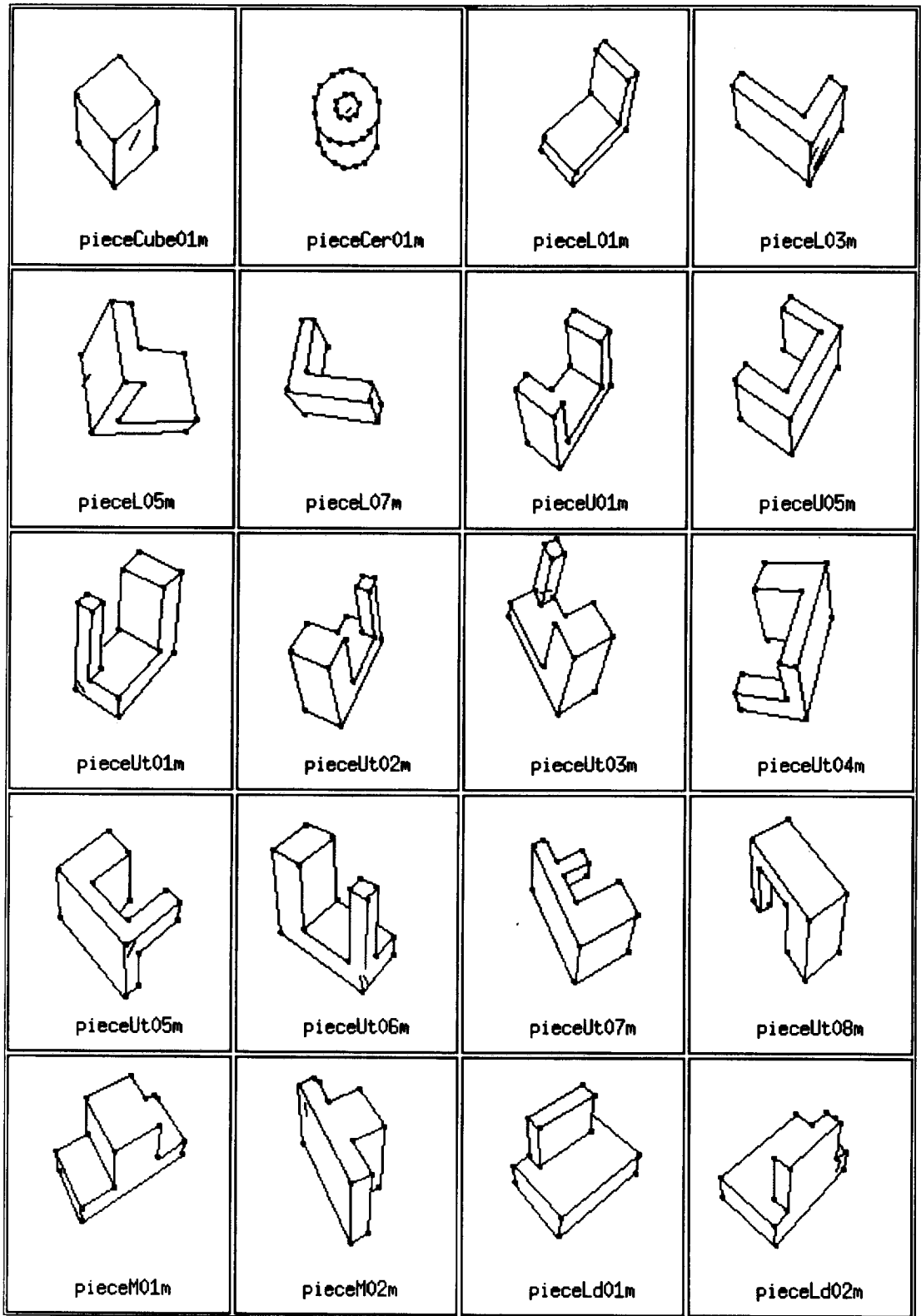


Fig. 9. The 20 graphs (lines and junctions) extracted from the previous images. These 20 graphs are decomposed into subgraphs (the decomposition is not shown) and stored in the hash-tables associated with the database.

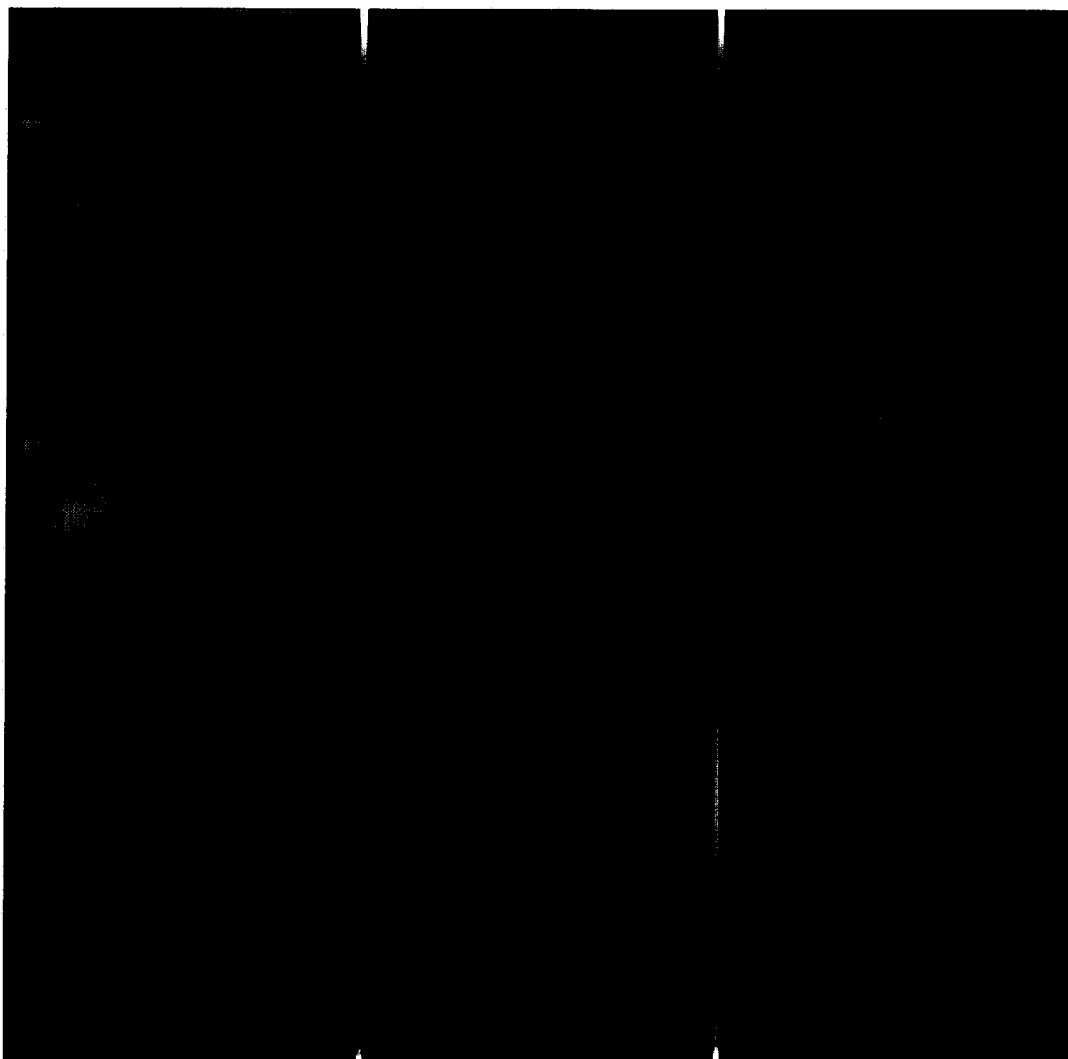


Fig. 10. Nine images of two objects to be recognized where the camera varies in position and orientation with respect to the two objects.

characteristic views by adding views of very different objects. Of course, if the database contains two very similar objects, the system will fail to discriminate between these two objects.

#### 7. DISCUSSION

Unlike the prevailing paradigm in computer vision that suggest image-feature-to-object-feature matching to solve for object recognition, we described an approach that uses an indexing technique to compare objects in an image with objects in a database. Our method does not rely either on precise knowledge about the geometry of the objects or on reliable feature-to-feature assignments. Instead we describe both the images and the models by weighted graphs and we

compare these graphs through their polynomial characterization, namely the second immanantal polynomial of the Laplacian matrix of a graph. This graph comparison was implemented in two steps (off-line and on-line): a database construction step (model graphs are stored in hash-tables) and an indexing step (an image graph indexes in the pre-stored hash-tables).

It is worthwhile to notice that, in the past, polynomial characterization of graphs has been used to represent and identify the topology of molecules.<sup>(36)</sup> To our knowledge, the graph theory literature does not describe any attempt to generalize polynomial characterizations to weighted graphs. It turns out that, at least for our purposes, this generalization is straightforward since the Laplacian matrix of a weighted graph has the same mathematical properties as the Laplacian matrix of a binary graph.

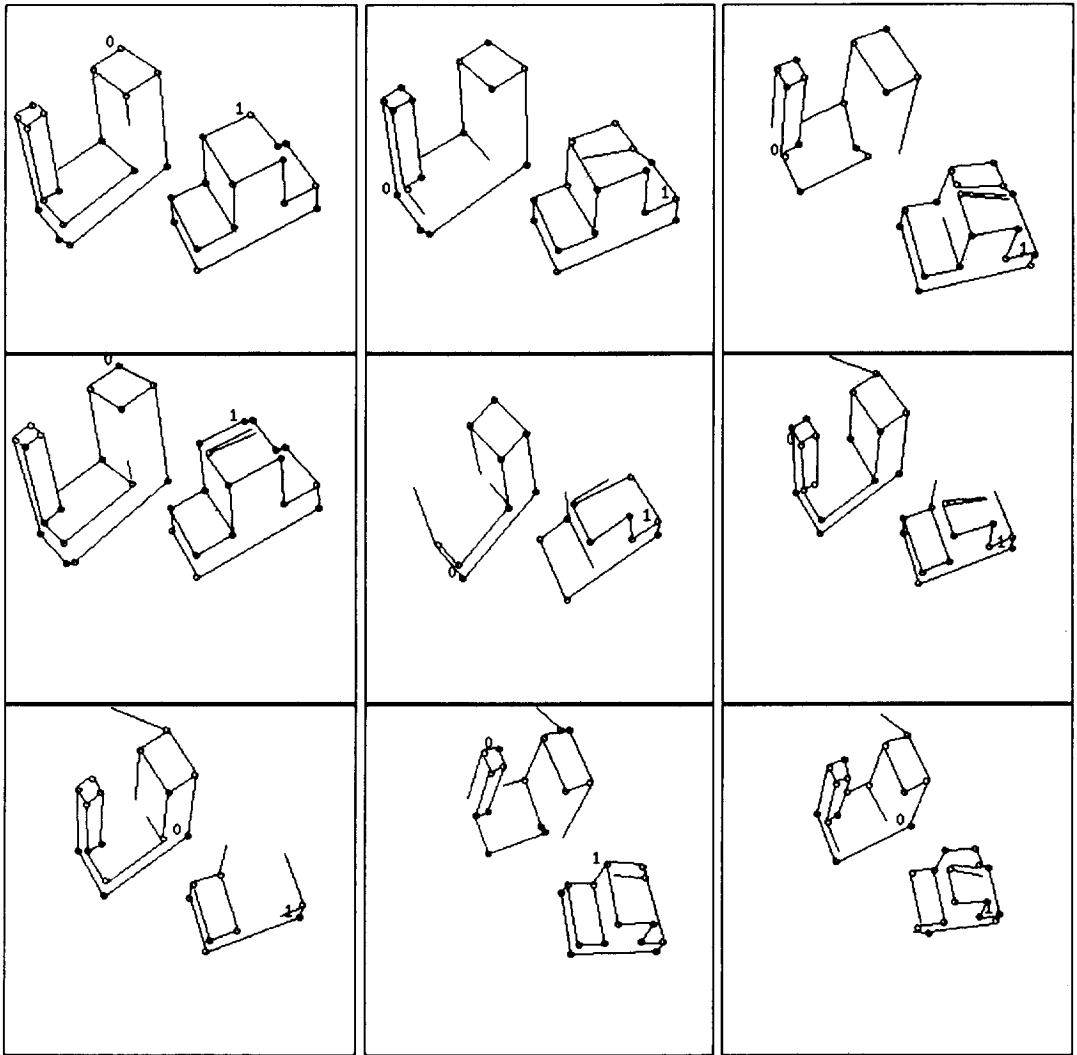


Fig. 11. The graphs extracted from the previous images. Notice that the noise corruption of these graphs varies extensively even if there is only a small change in camera position and orientation.

Table 1. The results of recognition for nine images of the same scene

Image number	Graph number	Characteristic view					
		Ut01m	Ut02m	M01m	M02m	L05m	Ld02m
1	0	18*	24*				
	1			23*			
2	0	16*	16*				
	1			17*			
3	0		18*				
	1			8*			
4	0	17*	24*				
	1			13*		10	
5	0				6		7
	1			9*			
6	0	17*	23*				
	1			6			7
7	0		23*				
	1						4
8	0		16*				
	1			4			5
9	0				6		8
	1			13*			

The figures correspond to scores (number of votes) as a result of the graph indexing process. The scores over-scripted by a \* correspond to a correct recognition.

We believe that our indexing scheme based on this algebraic graph representation has promising potential in computer vision and may provide in the future an interesting paradigm for indexing.

## REFERENCES

1. A. P. Ambler, H. G. Barrow, C. M. Brown, R. M. Burstall and R. J. Popplestone, A versatile computer-controlled assembly system. In *Proc. Third Int. Joint Conf. Artificial Intell.* pp. 298–307. Stanford University, California, U.S.A. (August 1973).
2. D. H. Ballard and C. M. Brown, *Comput. Vision*. Prentice-Hall Inc., Englewood Cliffs, New Jersey (1982).
3. R. C. Bolles and R. A. Cain, Recognizing and locating partially visible objects, the Local-Feature-Focus method. *Int. J. Robot. Res.* **1**(3), 57–82 (1982).
4. R. C. Bolles and R. Horaud, 3DPO: A three-dimensional part orientation system, *Int. J. Robot. Res.* **5**(3), 3–26 (Fall 1986).
5. O. D. Faugeras and M. Hebert, The representation, recognition, and locating of 3-D objects, *Int. J. Robot. Res.* **5**(3), 27–52 (Fall 1986).
6. N. Ayache and O. D. Faugeras, HYPER: A new approach for the recognition and positioning of two-dimensional objects, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-8**(1), 14–54 (January 1986).
7. W. E. L. Grimson and T. Lozano-Perez, Localizing overlapping parts by searching the interpretation tree, *IEEE Trans. Pattern Anal. Mach. Intell.* **PAMI-9**(4), 469–482 (July 1987).
8. C. Goad, *Fast 3D model-based vision. From Pixels to Predicates*, AL. B. Pentland, ed., Ch. 16, pp. 371–391. Ablex Publishing Corporation, Norwood, New Jersey (1986).
9. P. J. Flynn and A. K. Jain, BONSAI: 3-D object recognition using constrained search, *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(10), 1066–1075 (October 1991).
10. W. Y. Kim and A. C. Kak, 3-D object recognition using bipartite matching embedded in discrete relaxation, *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(3), 224–251 (March 1991).
11. P. J. Flynn and A. K. Jain, CAD-based computer vision: From cad models to relational graphs, *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(2), 114–132 (February 1991).
12. S. Dickinson, A. Pentland and A. Rosenfeld, 3-D shape recovery using distributed aspect matching, *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(2), 174–198 (1992).
13. R. Bergevin and M. D. Levine, Generic object recognition: Building and matching coarse descriptions from line drawings, *IEEE Trans. Pattern Anal. Mach. Intell.* **15**(1), 19–36 (January 1993).
14. S. Umeyama, An eigendecomposition approach to weighted graph matching problems, *IEEE Trans. Pattern Anal. Mach. Intell.* **10**(5), 695–703 (May 1988).
15. M. Hanajik, F. J. Kylstra and R. G. van Vliet, An analytical approach to the matching of attributed graphs, *Proc. 8th Scandinavian Conf. Image Anal.* pp. 419–425. Tromsø (August 1993).
16. H. A. Almoahamad and S. O. Duffuaa, A linear programming approach for the weighted graph matching problem, *IEEE Trans. Pattern Anal. Mach. Intell.* **15**(5), 522–525 (May 1993).
17. L. Héroult, R. Horaud, F. Veillon and J.-J. Niez, Symbolic image matching by simulated annealing, *Proc. British Mach. Vis. Conf.* pp. 319–324. Oxford, U.K. (September 1990).
18. V. Tresp and G. Gindi, Invariant object recognition by inexact subgraph matching with applications in industrial part recognition, *Proc. Int. Neural Network Conf.* pp. 95–98. Paris (July 1990).
19. R. Nevatia and T. Binford, Description and recognition of complex-curved objects, *Artific. Intell.* **8**, 77–98 (1977).
20. G. J. Ettinger, Large hierarchical object recognition using libraries of parameterized model sub-parts, *Proc. Comput. Vis. Patt. Recog.* pp. 32–41. Ann Arbor, Michigan, U.S.A. (5–9 June 1988).
21. A. Kalvin, E. Schomberg, J. T. Schwartz and M. Sharir, Two-dimensional model-based, boundary matching using footprints, *Int. J. Robot. Res.* **5**(4), 38–54 (1986).
22. Y. Lamdan and H. Wolfson, Geometric hashing: A general and efficient model-based recognition scheme, *Second Int. Conf. Comput. Vis.* pp. 238–249. Tampa, Florida, U.S.A. (December 1988).
23. F. Stein and G. Medioni, Structural hashing: Efficient 3-D object recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* **14**(2), 125–145 (February 1992).
24. T. M. Breuel, Adaptive model base indexing, *Proc. DARPA IU Workshop*, pp. 805–814 (1989).
25. D. M. Cvetkovic, M. Doob and H. Sachs, *Spectra of Graphs*. Academic Press, New York (1980).
26. J. Turner, Generalized matrix functions and the graph isomorphism problem, *SIAM J. Appl. Math.* **16**(3), 520–526 (May 1968).
27. R. Merris, K. R. Rebman and W. Watkins, Permanent polynomials of graphs, *Linear Alg. Appl.* **38**, 273–288 (1981).
28. R. Merris, The second immanantal polynomial and the centroid of a graph, *SIAM J. Alg. Disc. Meth.* **7**, 484–503 (1986).
29. G. M. Constantine, Graph complexity and the laplacian matrix in blocked experiments, *Linear Multilinear Alg.* **28**, 49–56 (1990).
30. R. Sedgewick, *Algorithms*. Addison-Wesley Publishing Company, Inc., Massachusetts (1988).
31. P. Gros and R. Mohr, Automatic object modelization in computer vision, *Proc. workshop "Advances in Structural and Syntactic Pattern Recognition"*, Bern, Switzerland, Vol. 5 of *Series on Machine Perception and Artificial Intelligence*. Bunke, ed., pp. 385–400. World Scientific (August 1992).
32. P. Gros, Matching and clustering: two steps towards automatic model generation in computer vision, *Proc. AAAI Fall Symp. Ser. Mach. Learning Comput. Vis. What, Why, How?* pp. 40–44. Raleigh, North Carolina, U.S.A. (October 1993).
33. L. Héroult and J. J. Niez, Neural networks and graph K-partitioning, *Complex Syst.* **3**(6), 531–576 (December 1989).
34. T. Kanade, Recovery of the 3D shape of an object from a single view, *Artific. Intell.* **17**(1–3), 409–460 (August 1981).
35. R. Horaud, F. Veillon and Th. Skordas, Finding geometric and relational structures in an image, *Computer Vision—ECCV 90, Proc. First Eur. Conf. Comput. Vis.*, O. Faugeras, ed., pp. 374–384. Antibes, France, Springer Verlag (April 1990).
36. Y. Kudo, T. Yamasaki and S. Sasaki, The characteristic polynomial uniquely represents the topology of a molecule, *J. Chem. Document.* **13**(4), 225–227 (1973).

**About the Author**—RADU HORAUD was born in Bucharest, Rumania, in 1953. He emigrated to France in 1972, where he received the “Diplôme d'Ingénieur” and the “Diplôme de Docteur-ingénieur”, both from the Institut National Polytechnique de Grenoble in 1977 and 1981, respectively. He spent 1982–1984 as a post-doctoral fellow at SRI International in the Robotics Department. While at SRI, he developed a laser-range-finder-based computer vision system which locates 3-D objects. He then spent 1984–1985 with the Laboratoire d'Electronique et de Technologic de l'Informatique and 1985–1986 with the Laboratoire d'Automatique de Grenoble. He is currently Director of Research at CNRS appointed at Laboratoire d'Informatique Fondamentale et Intelligence Artificielle. He serves as a scientific consultant for a number of industrial companies and for French Ministry of Transportation. His main research interests are in computer vision, robotics, nonlinear optimization and graph theory.

**About the Author**—JUAN HUMBERTO SOSSA-AZUELA received the B.S. degree in Electronics Engineering from the Universidad de Guadalajara, Mexico in 1981, the M.S. degree in Electrical Engineering from the CINVESTAV-IPN, Mexico in 1987 and the “Diplôme de Doctorat en Informatique” from the Institut National Politechnique de Grenoble, France in 1992. He currently holds a position of Assistant Professor at CINVESTAV-IPN, Mexico. His current areas of interest include computer vision and control of electromechanical systems using visual data.