Probabilistic reasoning in high-level vision

L Enrique Sucar and Dunan F Gillies

High-level vision is concerned with constructing a model of the visual world and making use of this knowledge for later *recognition*. In this paper, we develop a general framework for representing uncertain knowledge in high-level vision. Starting from a probabilistic network representation, we develop a structure for presenting visual knowledge, and techniques for probability propagation, parameter learning and structural improvement. This framework provides an adequate basis for representing uncertain knowledge in computer vision, especially in complex natural environments. It has been tested in a realistic problem in endoscopy, performing image interpretation with good results. We consider that it can be applied in other domains, providing a coherent basis for developing knowledge-based vision systems.

Keywords: probabilistic reasoning, Bayesian networks, knowledge-based vision

The process of vision involves the transformation of information, consisting of thousands of picture elements (pixels) obtained from the image(s) into a concise symbolic description that is useful to the viewer and not cluttered with irrelevant information[1]. This process has been divided into a series of levels, or a range of different representations, from low-level vision which involves operations acting directly on the images, through intermediate and high-level vision, that use the information obtained from the lower levels to build a simpler and more useful description. This process is represented schematically in *Figure 1*.

While low-level processing is essentially general purpose and does not make use of domain-specific knowledge, high-level vision is concerned with constructing a knowledge-base of the visual world and making use of this knowledge for later *recognition*. High-level vision seeks to find a consistent interpretation of the features obtained during low-level processing. This is called *sophisticated perception* by Pentland[3], and it uses specific knowledge about each particular domain to refine the information obtained through primitive perception. High level vision is based on *recognition*, that is matching our internal representation of the world with the sensory data obtained from the images. It involves the use of high-level specific models to infer from visual features the information required for subsequent tasks. At this level, model knowledge becomes more specific and plays a vital role in image interpretation.

According to their representation and recognition mechanisms we can classify vision recognition systems into two main categories: *model-based* and *knowledge-based*. In model-based systems the internal representation is based on a geometric model and the process of recognition consists in matching this model with a two dimensional (2D) or three dimensional (3D) description obtained from the image. The representation in knowledge-based systems is symbolic, including assertions about the objects and their relationships, and recognition is based on inference.

Imperial College of Science, Technology and Medicine, Department of Computing, 180 Queen's Gate, London SW7 2BZ, UK

Paper received: 29 September 1992; revised paper received: 12 July 1993

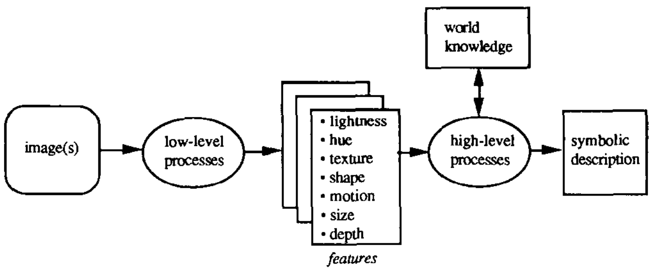


Figure 1 Levels of representation in vision. The different features[2] obtained from the low-level processes are fed into the high-level processes that use knowledge of the world for postulating what objects are in the image (recognition)

Model-based vision

Recognition in model-based vision involves matching the representation derived from the images with a set of predefined models. There are three main components in this type of system: feature extraction, modelling and matching. Feature extraction corresponds to what we called low-level vision, and in this case consists of deriving shape information from the images to construct a geometrical description. The modelling part involves the construction of internal geometrical models of the objects of interest, and it is usually done off-line prior to image analysis. Recognition is realized by geometric matching of the image description with the internal models.

Model-based vision systems are useful in certain domains, mainly in robot vision for industrial applications. Their success in these kind of tasks depends on three types of restrictions:

1. They use *simple models*. The objects to be recognized can be represented geometrically with few parameters and in a relatively precise way. This applies mainly to man-made objects.
2. They process *few objects*. The number of different objects in the domain is small. This is limited by the memory and processing requirements, especially if the system has to operate in real-time. Graph matching algorithms are computationally expensive. Indeed, in its worst case subgraph isomorphism is an *NP-complete* problem.
3. They have *robust feature extraction*. The features obtained through the perception processes are reliable and an object description can be easily constructed. This usually assumes known illumination conditions, surface properties and camera position, and sometimes the use of other sensors, such as multiple cameras and range finders.

Current model-based systems are limited as general-purpose vision systems[4] by the performance of the segmentation modules and their restricted representation of world knowledge. These limitations also apply to other domains that to not satisfy the restrictions mentioned above. They are not well suited for natural scenes such as those required for autonomous navigation in natural environments.

Knowledge-based vision

While model-based systems mainly use *analogical* models to represent objects, knowledge-based systems use *propositional* models, and recognition is realized by a process of inference which does not require the construction of an explicit model of the objects to be identified. They are a collection of propositions that represent facts about the objects and their relationships. The interpretation process consists of inferring, from the data and the known facts about the domain, the identity of the objects in the image. A general structure for this type of system is shown in *Figure 2*.

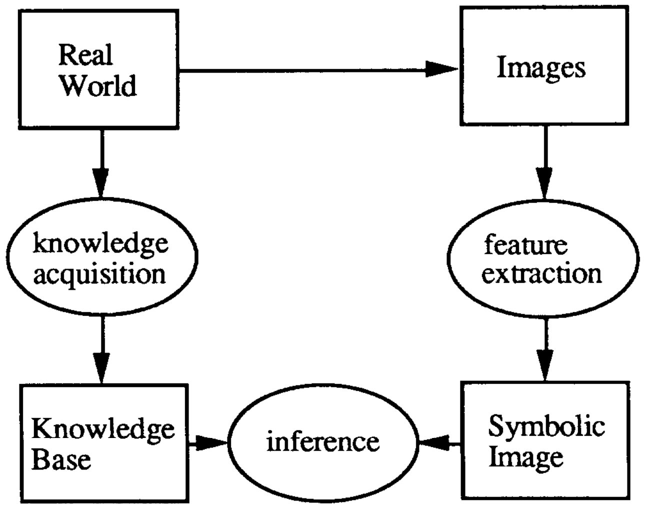


Figure 2 Knowledge-based vision system structure

This is similar to the general framework for model-based vision; the main differences are in the way we represent and use domain knowledge. In the model-based approach the representation is based on exact geometric models. In the knowledge-based case, the models use a symbolic description based on the different knowledge representation techniques developed in artificial intelligence (AI). We can divide a knowledge-based vision system into three processes:

- *Feature extraction*, which obtains the relevant features from the images and constructs a symbolic description of the objects used for recognition. These are integrated into what we call a *symbolic image*.
- *Knowledge acquisition*, which concerns the construction of the model that represents our knowledge about the domain. This is done prior to recognition and stored in the *knowledge-base*.
- *Inference*, which is the process of deducing from the facts in the knowledge-base and the information in the *symbolic image* the identity and location of the relevant objects.

Uncertainty

Our knowledge about the relationship between the image features (evidence) and the objects in the world (hypothesis) is often uncertain. If we knew the structure of the world we observe then we can categorically determine its projection in the image, as is the case in *computer graphics*, but given an image the relation becomes probabilistic, which is the case in vision. In both cases we represent the relationship between the world and the image in a *causal* direction, represented by a directed link from world to image.

Previous work in knowledge-based vision uses representations based on predicate logic, production systems, semantic networks and frames[5]. These systems have logic as their underlying principle, and reasoning is based on logical inference. Thus, handling uncertainty has been done by enhancing the basic representation and reasoning mechanisms, using some alternative numerical techniques such as Dempster–Shafer theory[6,7], certainty factors[8], fuzzy logic[9], a combination of them, or *ad hoc* methods developed for specific applications[10,11]. Most systems based their beliefs, certainty factors or probabilities on statistical data. Only probability provides a clear semantic interpretation, either as a statistical frequency (objective), or as a subjective estimate used traditionally in expert systems.

Probability theory provides an adequate basis for handling uncertainty in high-level vision. It is theoretically and practically appropriate in cases in which we require numerical predictions on verifiable events[12], which is the case in most computer vision systems. Probability theory satisfies the procedural adequacy criteria for a visual knowledge representation proposed by Mackworth[13], namely: soundness, completeness, flexibility, acquisition and efficiency. It also provides a preference ordering and efficient algorithms for a wide range of problems. The purely subjective Bayesian approach has been used in several vision applications[14], however, there are some difficulties in its application. Ng and Abrahmson[15] point out three main problems:

1. The difficulty in extracting from the expert(s) the knowledge to assess the required probability distributions.
2. The huge number of probabilities that must be obtained to construct a functioning knowledge-base.
3. The independence assumptions that are required to render the problem tractable.

Rimey and Brown[16] have made some progress towards reducing the large number of probabilities required by incorporating geometric properties of the scene into their computation, but their approach remains application oriented. In contrast, we have based our approach on objective probability, which provides an adequate framework for knowledge representation and reasoning in computer vision, and permits machine estimation of the probabilities.

KNOWLEDGE REPRESENTATION

Structuring visual knowledge

Within a knowledge-based framework, our visual knowledge is modelled by a series of facts about the objects and their relationships. These facts relate the features we obtain from the image with the objects of interest, possibly specifying, also, constraints and relationships between different objects. Given the uncertainty implicit in our model as well as in the features we obtain from the low-level vision processes, we need to integrate all the information we can obtain and our *a priori* knowledge to arrive at an interpretation. So our recognition mechanism should reason from the features or evidence obtained from the image to infer the identity of the objects in the domain.

The evidence comprises the features obtained from the image by domain-independent low-level vision, including uniform regions, contours, colour, 3D information, etc. The hypotheses are the different objects we expect to find in the world, which is usually restricted to a specific domain, although, of course, we can have several knowledge-bases, one for each domain. The knowledge-base relates the features to the objects, through intermediate regions, surfaces and sub-objects, depending upon the complexity of the objects in the domain. It can also include relationships and restrictions between objects, such as topological relations. These relations can also exist between sub-objects or lower-level features. Thus, we can represent our visual KB as a series of hierarchical relationships, starting from the low-level evidence features, to regions, surfaces, sub-objects and objects.

In probabilistic terms these relations correspond to *dependency* information, i.e. we can think of a series of image features giving evidence for certain objects; or an object *causing* the appearance of certain features in the image. This dependency information and its causal interpretation can be represented by a probabilistic network. Each node in the network will represent an entity (features, region, . . ., object) or relation (above, surrounding, near, . . .) in the domain, and the arcs will represent the dependencies between these entities. This representation makes explicit the dependency information, indicating that a certain object depends on a group of entities and is independent from the others. Thus, the structure of the probabilistic network will represent the qualitative knowlege about the domain. The probability distributions attached to the links will correspond to the strength of their dependencies. This representation reminds us of a semantic network representation, but in contrast, the structure of a probabilistic network has a clear meaning in terms of probabilistic dependencies, and it adds the representation of uncertainty to the relations, which are measured statistically. It also gives a rank ordering, in terms of probabilities, allowing us to select one of several conflicting interpretations of an image.

Probabilistic networks

Probabilistic networks, also known as Bayesian networks, causal networks or probabilistic influence diagrams, are graphical structures used for representing expert knowledge, drawing conclusions from input data and explaining the reasoning process to the user. A probabilistic network is a directed acyclic graph (DAG) whose structure corresponds to the *dependency* relations of the set of variables represented in the nework (nodes), and which is parameterized by the conditional probabilities (links) required to specify the underlying distribution. The variable at the end of a link is dependent on the variable at its origin, e.g. C is dependent on A in the probabilistic network in *Figure 3*, as indicated by link 1. We can think of the graph in *Figure 3* as representing the joint probability distribu-

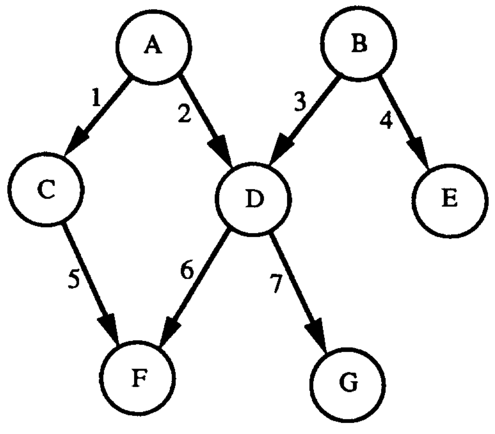


Figure 3 A probabilistic network

tion of the variables $A, B, \ldots, G$, as:

$$P(A, B, C, D, E, F, G) = P(G \mid D)\, P(F \mid C, D)\, P(E \mid B)\, P(D \mid A, B)\, P(C \mid A)\, P(B)\, P(A) \quad (1)$$

Equation (1) is obtained by applying the *chain rule* and using the dependency information represented in the network.

We can also view a probabilistic network as representing a rule-base. Each node corresponds to a variable, evidence or hypothesis, and the rules correspond to the dependencies represented by the links. A group of arrows pointing at a node represents a set of rules relating the variables at the origin of the links to the variable at the end, quantified by the respective conditional probabilities. For example, the links 5 and 6 in *Figure 3* represent the set of rules: *if* C_i, D_j *then* F_k; quantified in probabilistic terms as the conditional probability matrix $\boldsymbol{P}$, where $P_{ijk} = P(F_k / C_i, D_j)$. In a PN the arrows point from conditions to consequence, or causes to effects, denoting the causal structure in the physical world[17].

The topology of a probabilistic network gives direct information about the dependency relationships between the variables involved. In particular, it represents which variables are conditionally independent given another variable. By definition, A is conditionally independent of B, given D, if:

$$P(A \mid B, D) = P(A \mid D)$$

This is represented graphically by node D 'separating' A from B in the network. In general, $\boldsymbol{D}$ will be a subset of nodes from the network that if removed will make the subsets of nodes $\boldsymbol{A}$ and $\boldsymbol{B}$ disconnected. If the network represents faithfully the dependencies in the underlying probability distribution, this means that $\boldsymbol{A}$ is conditionally independent of $\boldsymbol{B}$ given $\boldsymbol{D}$. For example, in the PN of *Figure 3*, $\{E\}$ is conditionally independent of $\{A, C, D, F, G\}$ given {B}.

Formally, we define a probabilistic network as follows. Let V be a set of propositional variables with a joint probability distribution P, and $G = (V, E)$ a directed acyclic graph, which consists of a finite set of vertices or nodes V and an irreflexive binary relation E on V. The adjacency relation E specifies which nodes are adjacent, i.e. if $(v, w) \in E$ then w is adjacent to v, represented by an arc or edge from v to w. The adjacency relation specifies a dependency relation in probabilistic terms. Given the set of parents or causes of any node v denoted by C, v is conditionally independent of every other subset of nodes X, i.e.:

$$P(v \mid C, X) = P(v \mid C)$$

where X is a subset of propositional variables from V excluding v and v's descendants denoted by D, i.e. $X \subseteq \{V - (D \cup v)\}$. Then $\mathrm{PN} = (V, E, P)$ is called a probabilistic network.

Visual probabilistic networks

For the hierarchical description of our visual knowledge-base, we can think of the network as divided in a series of layers in which the nodes of the lower layer correspond to the feature variables and the nodes in the upper layer to the object variables. The intermediate layers have nodes for other visual entities, such as parts of an object or image regions; or represent relations between features and objects. The arrows point from nodes in the upper layers toward nodes in the lower layers, expressing a causal relationship. The structure of a probabilistic network for visual recognition is shown in *Figure 4*.

As we will discuss in the following section, probability propagation in a general network is computationally expensive, so in practice it is necessary to restrict our representation to certain types of networks. We require a network representation that is suitable for visual recognition, but at the same time efficient in terms of memory and processing requirements. The hierarchical visual network that we discussed before can be thought of as a series of trees, with each tree having its root at one object, connected to other nodes in the intermediate layers, and with the leaves of the tree as the feature nodes at the lower level. It is generally the case that we can represent the intermediate objects and features as separate nodes for each

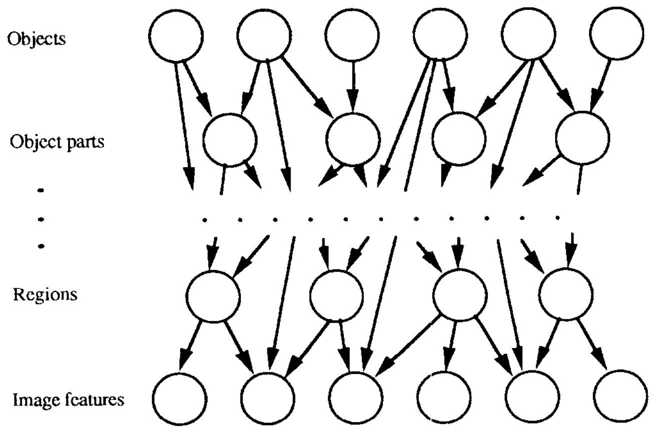


Figure 4 Hierarchical probabilistic network for visual recognition

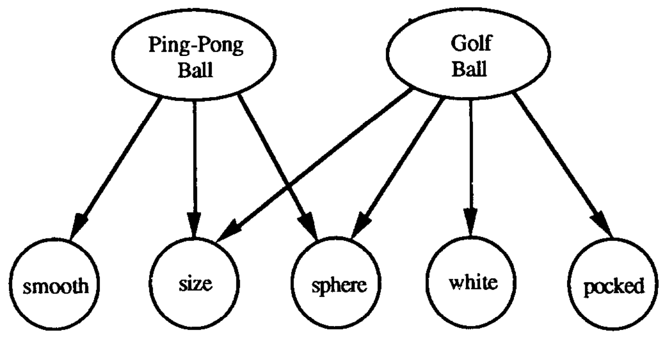


Figure 5 Example of a multitree for visual recognition. Feldman[2] gives this as an example of a description of Ping-pong and golf balls according to his functional model for vision. It is interesting to note that this model, derived from another perspective, is analogous to a probabilistic multitree (the figure is based on Feldman[2], p. 541)

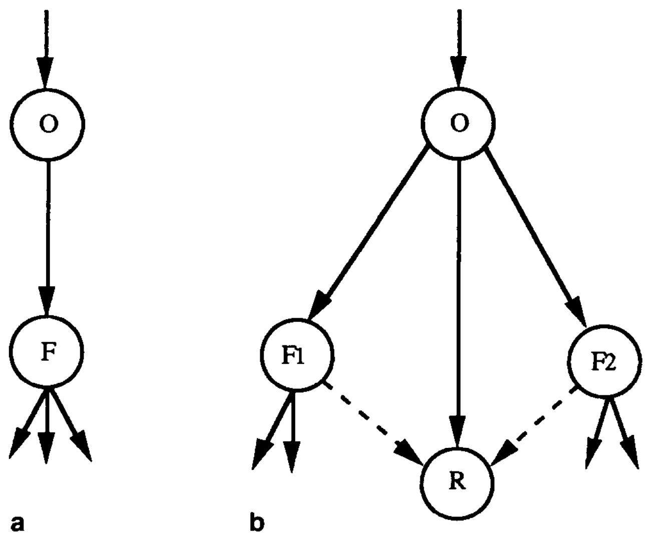


Figure 6 Expressing relational knowledge. The network in (a) represents a unary relation between O and F, and the network in (b) a binary relation between O and F_1, F_2 which is expressed through the relational variable R. The deterministic links between F_1, F_2 and R indicate that the value of R is a deterministic function of F_1 and F_2. →: Probabilistic link; - - -→: deterministic link

object hypothesis, with common nodes only for representing relations. The low level evidence or feature nodes will correspond to the information that is obtained directly from the image, so these nodes will be common to several objects, representing conflicting hypotheses. Thus, we have a network in which certain nodes have no parent (objects), other nodes have one parent (intermediate objects/features), and some nodes have multiple parents (features and relations). We will restrict the multiple parents variables to leaf nodes, that is, nodes with no descendants and call this type of network a *multitree*. An example of a multitree is depicted in *Figure 5*.

Before we can define formally a multitree, some additional definitions are required. Let $G = (V, E)$ be a DAG, and v and u vertices in V. If $(u, v) \in E$ then u is a *parent* of v and v is a *child*. If there is a path from u to v, u is an *ancestor* of v and v is a *descendant* of u. If u has no parents it is called a *root* and if u has no children it is called a *leaf*.

Now we can define a multitree as follows. Let $G = (V, E)$ be a DAG, and v a vertex in V. G is called a *multitree* if every non-leaf vertex $v \in V$ has at most one parent, i.e. that the root nodes have no parents, the leaf nodes have mutliple parents, and every other node has exactly one parent.

In a multitree there is one tree that encapsulates the rules for recognition for each object. Each recognition tree consists of a single root that represents the object of interest, intermediate objects/features, and several leaves that constitute the measured parameters. Although this structure could seem somewhat restrictive, it is generally adequate for representing a visual KB, and probability propagation can be done efficiently as we will show later.

Expressing relational knowledge

The causal relationships represented in the visual probabilistic network we have discussed previously express unary relationships between objects and features, i.e. the presence of a certain object in the world will *cause* the appearance of some specific features in the image, which are usually assumed independent. So there is a unary relationship between an object O and a feature F. This is represented graphically by the probabilistic network in *Figure 6a*. However, in many cases the relation between several parts or features are important for the recognition of an object.

Multiple objects in the world are easier to model as the composition of several sub-objects or parts, subdivided recursively until at the lowest level there are simple objects that will generally correspond to a region in the image. An example of this type of representation is the 3D models proposed by Marr[1] in which an object is described by a number of cylindrical parts and their relationships. In this case the recognition of an object is not only dependent on the presence of its subparts, but it is also dependent on the physical relation between them. We need to represent this dependency of the object on the relation between its components in the network.

A physical relation between the components could, for example, be the distance between two regions or their topological relation (adjacent, above, surrounding, etc.). These relationships are well defined, and so we can say that there is a *determinisitic link* from the components to their relation. However, the instance of a relation extracted from the image could either be caused by the object of interest, or be accidental or illusory. To represent this uncertainty, we consider that there is some *probabilistic link* from the object to the relation. For the case of one object and a binary relation between two features, the corresponding network is shown in *Figure 6b*. We represent a probabilistic link by an arrow and functional or deterministic links by a dotted arrow (*Figure 6c*).

We can extend the representation to include more variables, and in general a relation for n variables can be expressed by an analogous network, as depicted in *Figure 7a*. Assuming that the relation can be uniquely

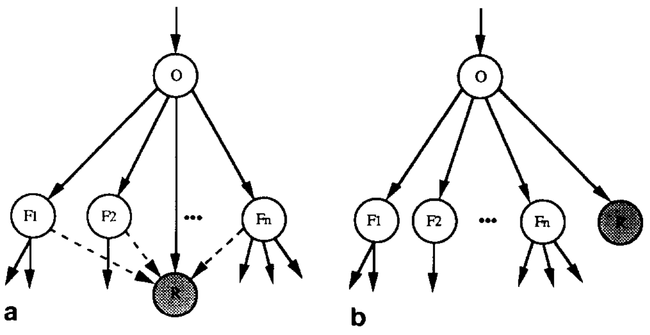


Figure 7 Probabilistic network for an n-variable relation. The network in (a) expresses the functional and probabilistic dependencies, which is transformed into the equivalent probabilistic network depicted in (b). The relation variable R is treated as an instantiated variable drawn as a shaded circle

determined from the related variables, node R becomes a deterministic variable in the same way as the input variables, shown as a shaded node in the network. Thus, if we eliminate the functional links from the network, we are left with the probabilistic network shown in *Figure 7b*. This network also has a tree structure and we can apply the algorithms that we will develop for probability propagation in *multitrees*.

Expressing temporal knowledge

We live in a dynamic environment in which the information we obtain from our visual sensors changes constantly as a function of our movement or the movement of the surrounding objects. We can think of this dynamic information as a series of images which together provide more information than a single static image. To be able to operate in a dynamic environment we are required to incorporate in our knowledge-base temporal information, which can be used efficently as an aid for recognition. We will focus on the navigation aspect, considering a series of images that are similar, with an incremental difference between one image and the next due to the movement of the camera or the environment; and we will consider two cases:

- *Semi-static recognition*, in which an object can be identified from a single image but observations from previous images can be useful as another clue for recognition.
- *Dynamic recognition*, in which a series of images are necessary to identify an object, which otherwise would be impossible or very difficult to recognize.

In semi-static recognition we consider an object O which can be identified by a number of image features $F_1, \ldots, F_N$ obtained from an image at time t_i. Assuming that the time difference Δt between images is small, the same type of object will be present at the same location in previous and successive images. We can state that object O appearing at time t_{i-1} *causes* the presence of the same object at time t_i, with an analogous relation between t_i and t_{i+1}. This can be represented graphically by the network in *Figure 8a*, where the conditional probabilities for the links connecting the matching objects at different times will be *high* for objects of the same type and *low* for objects of different types. For evaluating the network at time t_i, we can consider only the node for the object at time t_{i-1} obtaining the structure shown in *Figure 8b*. This simplification is based on the assumption that knowing the identity of the object at t_{i-1} makes the previous observations irrelevant for O at t_i. Thus, the posterior probabilities for an object in the previous image will influence the prior probability of the corresponding object in the current image. We will continue to have a tree structure although the object of interest is no longer at the root of this tree.

In dynamic recognition, an object is defined by the presence of other objects or certain image features across a series of successive images. We can think of this case as analogous to the hierarchical description of an object in terms of parts, regions, etc., but in which the sub-objects are in different images instead of the same one. Then we can represent a dynamic object with a probabilistic tree, where the presence of object O is dependent on the observation of objects $S_1, \ldots, S_N$ in different successive images, as it is shown in *Figure 9a*. The physical relation, such as the distance, between these objects could be also important and we can extend the representation by including this relational information in the way described in the previous section (*Figure 9b*).

Both types of dynamic knowledge representations keep a single dependency structure in which probability propagation can still be done efficiently.

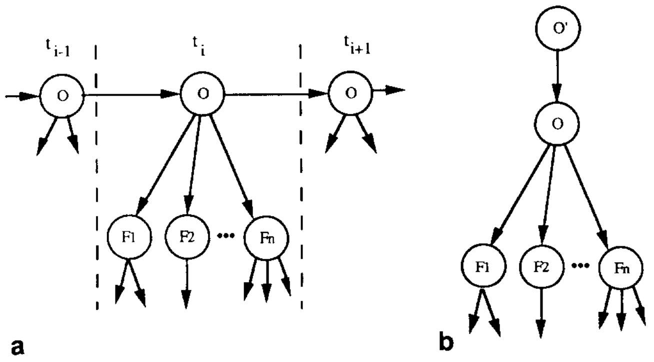


Figure 8 Semi-static recognition. The network in (a) represents the causal relations between an object in several images. This is simplified to the network in (b) for recognition at time t_i

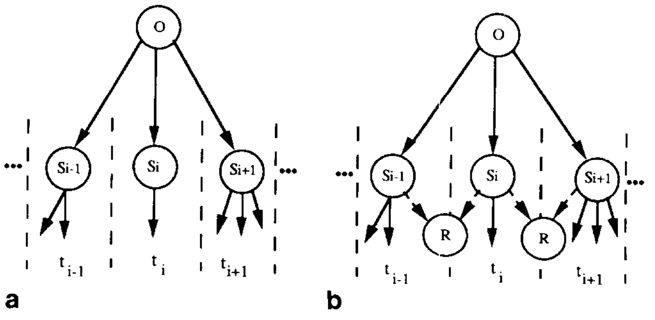


Figure 9 Dynamic recognition. The network in (a) corresponds to a probabilistic network for representing an object across several images, which is extended to include relations in (b)

PROBABILITY PROPAGATION

There are two basic modes of reasoning that we can use with a visual knowledge base represented as a probabilistic network. First, given an image or a series of images, we will use the network to recognize the object or objects in the image(s). This corresponds to *deductive reasoning* and will be implemented by *probabilistic propagation* techniques described in this section. The second mode of reasoning has to do with constructing the visual KB, that is learning or *inductive reasoning* (known as *knowledge acquisition* in knowledge-based systems), which will be discussed in the next section.

Probability propagation consists in instantiating the input variables, and propagating their effect through the network to update the probability of the hypothesis variables. In contrast with previous approaches, the updating of the certainty measures is consistent with probability theory, based on the application of Bayesian calculus and the dependencies represented in the network. Probability propagation in a general network is a complex problem, but there are efficient algorithms for certain restricted structures, and alternative approaches for more complex networks. Pearl[19] developed a method for propagating probabilities in networks which are tree-structured, and which was later extended to *polytrees*. For *multiply connected networks*, different approaches have been suggested. Lauritzen[18] developed a technique based on graph theory, which will work efficiently in arbitrary 'sparse' networks. Other techniques include *conditioning* and *stochastic simulation*[19]. We have developed an algorithm for probability propagation in multitrees[20].

Probability propagation in trees

Given a PN and the local conditional probabilities, it is desirable to be able to propagate the effect of new evidence by local computations, communicating via the links provided by the network. This mechanism, in analogy with logical inference, allows the tracing of the reasoning process for its explanation to the user, an important feature in expert systems; and it has potential for a parallel implementation. Pearl[17] developed a method for propagating probabilities in networks which are tree-structured, i.e. each node has only one incoming link or one parent. In a *probabilistic tree* every node has only one *parent* except one node, called the root, which has no incoming links. An example of a tree is shown in *Figure 10a*.

Each node represents a discrete multivalued variable, e.g. $A = \{A_1, A_2, \ldots, A_n\}$, and each link is quantified by a conditional probability matrix, $\boldsymbol{P}(B/A)$, where $P(B/A)_{ij} = P(B_j/A_i)$. Given certain evidence V, represented by the instantiation of the corresponding variables, the posterior probability of any variable B, by Bayes' theorem will be:

$$P(B_i \mid V) = P(B_i)\, P(V \mid B_i)/P(V) \qquad (4)$$

Given the dependencies represented in the tree, B separates it into two independent subtrees, one formed by all the descendants of B and the other by every other node. Pearl[19] shows that we can combine the evidence from the whole tree at node B using the equation:

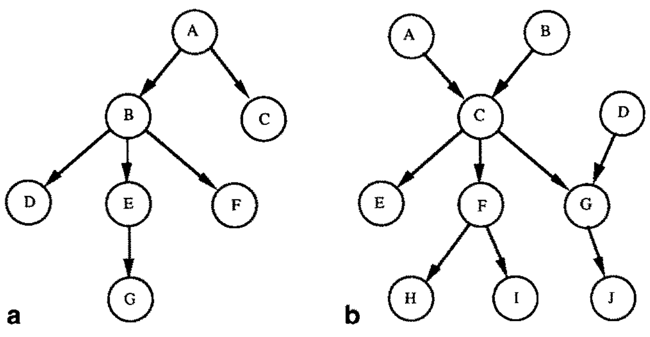


Figure 10 A probabilistic tree (a) and a polytree (b)

$$P(B_i/V) = \alpha\, \pi(B_i)\, \lambda(B_i) \qquad (5)$$

where α is a normalizing constant. $\lambda(B_i)$ is the evidence from the tree below B and is recursively defined as the product of the evidence from the sons of B:

$$\lambda(B_i) = \prod_k \left[\sum_j P(S_j^k \mid B_i)\, \lambda(S_j^k) \right] \qquad (6)$$

$\pi(B_i)$ is the evidence from above B and is given by:

$$\pi(B_i) = \sum_j P(B_i \mid A_j) \left[\alpha\, \pi(A_j) \prod_l \lambda_l(A_j) \right] \qquad (7)$$

where l ranges over all the descendants of A except B, that is the siblings of B. Equation (7) shows how to obtain the *causal* support for B from its ancestors.

Equations (5), (6) and (7) constitute the basis for the propagation mechanism in a probabilistic tree. For this we only need to store the vectors λ and π in each node, and update them with the corresponding parameters from its neighbours, and the fixed conditional probability matrix $\boldsymbol{P}$ for that node. This could be implemented by a message passing scheme[17] in which each node acts as a simple process which communicates with its neighbours (father and sons). Initially the network is in equilibrium. When information arrives, some nodes called *data nodes* are instantiated and the information is propagated through the network by each node sending messages to its parent and sons, in two directions:

- *Bottom-up propagation*. Each node (B) sends a message to its father (A), calculated as:

$$\lambda_B(A_i) = \sum_j P(B_j \mid A_i)\, \lambda(B_j) \qquad (8)$$

- *Top-down propagation*. Each node (B) sends a message to each of its sons (S), for the kth son it is computed by:

$$\pi_k(B_i) = \alpha\, \pi(B_i) \prod_{l \neq k} \lambda_l(B_i) \qquad (9)$$

Each node uses this information to update its local parameters, and update its posterior probability if required. For example, if nodes D and F were instantiated in the network of *Figure 10a*, they will send λ messages to B, which will then send a λ message to A and π messages to all of its sons. After the messages reach the root node, they will propagate top-down until they reach the leaf nodes, where the propagation terminates and the network comes to a new equilibrium. So the information propagates through the tree in a single pass, in a time (in parallel) proportional to the diameter of the network.

The previous propagation mechanism only works for trees, so it is restricted to only one cause per variable. An extension for more complex networks, called *polytrees*, was proposed by Kim and Pearl[17]. In a polytree each node can have multiple parents, as shown *Figure 10b*, but it is still a singly connected graph.

Probability propagation in multiply connected networks

Multiply connected networks are those in which there are multiple paths between nodes in the directed graph, implying the formation of loops in the underlying network. If we apply the propagation schemes of the previous section in this type of network, the messages will circulate around these loops, and we will not reach a coherent state of equilibrium.

For a sparse network, a solution is to transform it into a different structure, for which an efficient propagation scheme can be applied. For this, Lauritzen and Spiegelhalter[18] developed a method based on graph theory, which extracts an undirected *triangulated* graph from the probabilistic network, and creates a tree whose nodes are the *cliques* of this triangulated graph. Then the probabilities are updated by propagation in the tree of cliques. This technique will work efficiently if the number of variables in each clique is relatively small. It has been applied in *MUNIN*, a medical expert system for diagnosis and test planning in electromyography[18].

Pearl[19] suggests two other methods for dealing with multiply connected networks: *stochastic simulation* and *conditioning*. Stochastic simulation consists of assigning to each non-instantiated variable a random value and computing a probability distribution based on the values of its neighbours. Based on this distribution, a value is elected at random for each variable, and the combined values represents a sample point. The procedure is repeated a number of times, and the posterior probability of each variable is calculated as the percentage of times each possible value appears. Conditioning[19] is based on the fact that if we instantiate a variable it 'blocks' the paths for which the corresponding node forms part, changing the connectivity of the network. Thus, we can transform a multiply connected graph into a singly connected graph by instantiating a selected group of variables. We assume a set of fixed values for these variables, and so the probability propagation for the singly connected network. This is repeated for all the possible values of the separating variables, averaging the results weighted by the posterior probabilities.

These techniques provide efficient algorithms for probability propagation in certain classes of networks: tree, polytrees and sparse multiply connected networks. But in general, we have more complex structures, and it has been shown by Cooper[21] that probabilistic inference is *NP-hard*.

Probability propagation in *multitrees*

As we can see in *Figure 5*, a multitree is not singly connected, i.e. there can be multiple paths between nodes. Algorithms for multiply connected networks are more complex, and only work efficiently for certain kind of structures. Thus, we need to develop a technique for efficient propagation in multitrees, or else, a transformation of the network to make it singly connected. There are two important characteristics of the general structure of a visual PN that help to simplify probability propagation:

1. Probability propagation is usually bottom-up. That is, from the evidence nodes instantiated from the images towards the object nodes whose posterior probability we want to obtain.
2. The leaf nodes will usually correspond to instantiated or evidence variables. These nodes, the feature and relation nodes, will have fixed values obtained from the feature extraction processes of low-level vision.

By making use of these properties and the conditioning technique we described before, we can separate a multitree into a series of independent trees so that we can do probability propagation independently in each one by extending the technique for probability propagation in probabilistic trees.

Theorem 1 *Let $G = (V, E)$ be a multitree*, and $W \in V$ *the subset of all leaf nodes in V. If every node $v \in W$ is instantiated, then the multitree G can be separated in one or more probabilistic networks $G_1, G_2, \ldots, G_N$ where each network G_i is a probabilistic tree.*

Proof
By conditioning we block the pathways that go through the instantiated nodes. In the case of leaf nodes, this is equivalent to partitioning the network along this node, and connecting a copy of it to each one of its parents. Given that all the leaves are instantiated we can partition the network at every leaf node, meaning that each instantiated copy of the node will have only one parent. If every leaf has only one parent and, by definition, every other node in a multitree has at most one parent, then every node in the network has at most one parent. From this it follows that the network is a *forest*, that is a DAG where each node has at most one parent. A tree is a particular case of a forest where there is only one root node, and a forest is equivalent to one or more trees.

Figure 11 shows an example of a multitree which has

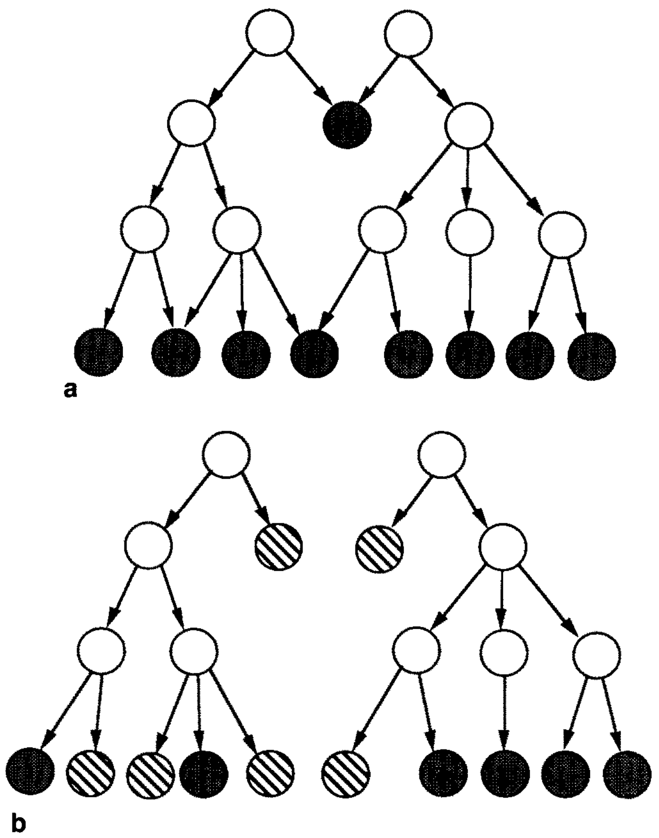


Figure 11 Multitree with instantiated nodes and equivalent trees. In (a) we show the network with the instantiated variables shown as filled circles, and in (b) the two equivalent trees that form a forest. Three instantiated leaf nodes have been partitioned to render the network singly connected; these are shown as circles with black stripes

been transformed into a forest by instantiating the leaf variables. In *Figure 11a* we show the network with the instantiated variables shown as filled circles, and in *Figure 11b* the two equivalent trees that form a forest. Three instantiated leaf nodes have been partitioned to render the network singly connected; these are shown as circles with black stripes.

After partitioning the knowledge base into multiple independent trees, we apply the probability propagation technique for trees to propagate the evidence through each tree independently and obtain a posterior probability for each object hypothesis. Given that all the leaf nodes are instantiated, we need only the bottom-up propagation part of the algorithm, combining the diagnostic support received from each child upwards, until we arrive to the root node. So by using both conditioning and propagation in trees, we obtain a probability propagation algorithm for visual recognition in a multitree.

Algorithm 1 ***Probability Propagation in Multitrees***

```
1. Decompose the network into $N$ probabilistic trees ($T_l$) by partitioning it in every leaf node with multiple parents.
2. Instantiate all leaf nodes by using the measured features obtained from low-level vision.
3. For every tree $T_l$ ($l = 1, 2, \ldots, N$):
    3.1 For all instantiated (leaf) nodes $B$, set $(B_i) = 1$ for the instantiated value, and 0 otherwise.
    3.2 From each leaf node $B$ send a message to its parent node $A$ calculated as:
            $\lambda_B(A_i) = \sum_j P(B_j \mid A_i)\,\lambda(B_j)$     (10)
    3.3 For each parent node $A$ update its $\lambda(A_i)$ by multiplying the messages received from its children:
            $\lambda(A_i) = \prod_k \lambda_k(B_i)$     (11)
    3.4 Make the previous parent nodes $A$ as the current 'leaf' nodes $B$. Repeat steps 3.2 and 3.3. until the root node is reached.
    3.5 Obtain the posterior probability of the root $B_i$ by:
            $P(B_i \mid V) = \alpha\,\pi(B_i)\,\lambda(B_i)$     (12)
        where $\pi(B_i)$ is its prior probability $P(B_i)$ and $\alpha$ is a normalizing constant.
```

This algorithm lends itself to a natural parallel implementation. We can propagate the probabilities for each tree in parallel, so that the objects' posterior probability can be updated in a time linearly proportional to the number of levels in the largest tree of the network. For image understanding in navigation, such as in endoscopy, we need to process images sequentially. In this case, the algorithm can be made even faster by pipelining, making each level in the tree one stage in the pipeline.

LEARNING

Given a KB represented as a probabilistic network, the process of knowledge acquisition is divided naturally into two parts[19]: *structure learning* and *parameter learning*. Given a certain structure, parameter learning consists of finding the required prior and conditional probability distributions. An advantage of using probability to represent uncertainty is that we make use of techniques developed in statistics and estimation theory, mainly in the parameter estimation phase. Structure learning has to do with obtaining the topology of the network, including which variables are relevant for a particular problem, and their dependencies. This is of major importance for visual probabilistic networks, because the structure of the network determines which features are relevant for each object, a critical aspect for recognition. Also, the independence assumptions we are using when doing probability propagation are dictated by the topology of the network, and if not valid could degrade the performance and make invalid the calculated probabilities.

Parameter learning in *multitrees*

We start with only qualitative information from the domain, that is the structure of the network, and obtain all the prior and conditional probability distributions from real data. In the case of a multitree, we need to estimate the prior probability distributions of each root node $P(A_i)$ denoted by $\boldsymbol{P}(A)$, and the conditional probability distributions of each node given its parent $P(B_j|A_i)$, denoted by $\boldsymbol{P}(B|A)$. We will assume that each node represents a discrete multivalued variable with N possible values, so each $\boldsymbol{P}(A)$ will be stored as an N dimensional vector and each $\boldsymbol{P}(B|A)$ as an $N \times M$ dimensional matrix. Each entry in $\boldsymbol{P}(A)$ and $\boldsymbol{P}(B|A)$ will be stored as an integer ratio. Given the nature of the vision domain and the limited resolution of image acquisition, most variables will be discrete or can be given a discrete approximation in a limited range. In the second case we use a method based on *Parsen windows*[22] to estimate the probability density function, approximating it by N intervals with constant value.

Given an observation and the actual value of each node, it is trivial to update the system parameters by incrementing the corresponding entries in the prior and conditional probability matrices. If we observe A_k and B_l, and the previous values are: $\boldsymbol{P}(A) = a_i/s$ and $\boldsymbol{P}(B|A) = b_j/a_i$, then the probabilities will be updated by using the following equations:

- prior probabilities:

$$P(A_i) = \frac{a_i + 1}{s+1}, \quad i = k \tag{13}$$

$$P(A_i) = \frac{a_i}{s+1}, \quad i \neq k \tag{14}$$

- conditional probabilities:

$$P(B_j|A_i) = \frac{b_j + 1}{a_i + 1}, \quad i = k \text{ and } j = 1 \tag{15}$$

$$P(B_j|A_i) = \frac{b_j}{a_i + 1}, \quad i = k \text{ and } j \neq 1 \tag{16}$$

$$P(B_j|A_i) = \frac{b_j}{a_i}, \quad j = k \tag{17}$$

We start with all values set to zero and update them with each observation. In this way, the probabilistic parameters are learned by the system and its performance will improve as more data is accumulated.

Another alternative to starting with no information is to start with a subjective estimate as an initial value. For this, the values given by an expert in the domain could also be represented as ratios, assuming an implicit sample size[23], and improved with statistical data. This approach can be useful in cases in which there is not enough data, providing an initial estimate for the required parameters. A problem is that unless the expert's estimate is close to the objective probabilities, the system may take more time to learn than in the case in which it starts from zero.

Previously, we have assumed that all the variables are directly observable from the images, or are provided as feedback by the users in the learning phase. But it could be that some intermediate nodes could not be measured directly. In this case, we propagate the probabilities in both directions in the tree (from the root and from the leaves) towards the unknown node and obtain a posterior probability for it. The value with the highest probability is assumed to be the 'observed' and all the conditional distributions are updated as above. Assuming that at least the root and leaf nodes are instantiated, we combine the bottom-up and top-down propagation mechanisms for probabilistic trees to develop an algorithm for estimating the probability distribution of 'hidden nodes' in a multitree.

Algorithm 2 ***Learning Hidden Nodes in a Multitree***

1. Decompose the network into N probabilistic trees (T_1) by partitioning it in every leaf node with multiple parents.
2. Instantiate all leaf nodes by using the measured features obtained from low-level vision.
3. Instantiate the root node and all possible intermediate nodes from feedback provided by an external agent (usually the expert).
4. For every tree T_l ($l = 1, 2, \ldots, N$):

   4.1 For all instantiated nodes B, set $\lambda(B_i) = 1$ and $\pi(B_i) = 1$ for the instantiated value, and 0 otherwise.

   4.2 From each instantiated node B send a λ message to its parent A and π messages to its children S, calculated as:

   $$\lambda_B(A_i) = \sum_j P(B_j|A_i)\,\lambda(B_j) \tag{18}$$

   $$\pi_S(B_i) = \alpha\, \pi(B_j) \prod_{l \neq s} \lambda_l(B_j) \tag{19}$$

   4.3 For each non-instantiated node B that receives λ messages from its children S and a π message from its parent A, update its current $\lambda(B)$ and $\pi(B)$ by:

   $$\lambda(B_i) = \prod_k \lambda_k(S_i) \tag{20}$$

   $$\pi(B_i) = \sum_j P(B_i|A_j)\,\pi_B(A_j) \tag{21}$$

   and send λ messages to its children and a π message to its parent using equations (19) and (21), respectively.

   4.4 Repeat step 4.3 until the propagation terminates by reaching instantiated nodes.

   4.5 Obtain the posterior probability of non-instantiated nodes B by:

   $$P(B_i \mid V) = \alpha\, \pi(B_i)\,\lambda(B_i) \tag{22}$$

   where α is a normalizing constant.
5. Set non-instantiated variables B to the value B_i that corresponds to the highest posterior probability, so that $P(B_i|V) \geqslant P(B_j|V)$, $\forall j \neq i$.

6. Update all the prior and conditional probabilities using equations (13) to (17).
7. Repeat steps 2 to 6 for each sample until the learning phase is finished.

In this algorithm we are using the simplest way to update the probabilities of unobservable or hidden nodes, in which the conditional probability with highest value is incremented by one implying that we assume this value with probability equal to one. But this could be a too strong assumption to make and in some cases could lead to an incorrect state from which the system cannot escape. A better alternative is to have a smaller increment depending on the posterior probabilities of the node. For this several techniques have been proposed, including *decision directed*, *probabilistic teacher* and *method of moments*[24]. A simplified version of Algorithm 2 will be used when all variables are observable, eliminating steps 4 and 5.

Structure learning

Structure learning consists of finding the topology of the network, that is the dependency relationships between the variables involved. Most KB systems obtain this structure from an *expert*, representing in the network the expert's knowledge about the causal relations in the domain. But our knowledge could be *deceptive*, especially in the domain of vision in which recognition seems to be at a subconscious level. Also, knowledge acquisition could be an expensive and time consuming process. So we are interested in using empirical observations to obtain and improve the structure of a probabilistic network. Relatively little research has been done on inducing the structure of a PN from statistical data. Chow and Liu[25] presented an algorithm for representing a probability distribution as a dependency tree, and this was later extended by Rebane and Pearl[19] for recovering causal polytrees.

Although these techniques allow us to obtain certain types of structures from second order statistics, there are several important limitations:

- They are restricted to predefined structures: trees or polytrees, and there is a guarantee of an optimum solution only for trees.
- They cannot obtain the direction of all the links in the network. Even for a tree the selection of the root node is arbitrary.
- They cannot find the actual causal relations between variables without additional information or external semantics, i.e. they cannot distinguish genuine causal relationships from spurious correlations[19].
- They make use of the closed world assumption, considering that all relevant variables are already included in the network.

Due to these limitations, instead of starting from no information about the structure of the network, we use the network derived from the subjective rules

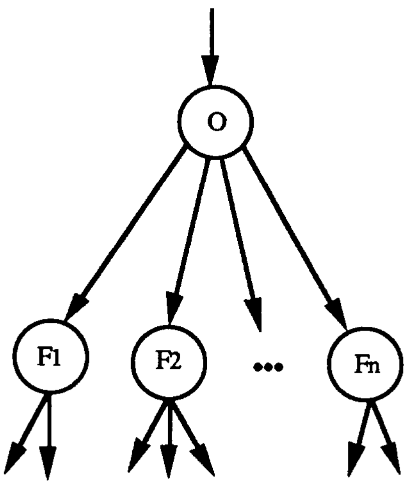


Figure 12 A recognition subtree

as an initial topology. Then we employ a methodology based on statistical techniques to improve the initial structure[26].

We decompose our visual PN model into several trees (multitree), with each one representing our knowledge for one object. In a tree, each node is directly dependent on its parent and sons and conditionally independent of all the other nodes. Thus, in a recognition subtree (see *Figure 12*) with object O and features $F_1, F_2, \ldots, F_n$, the evidental information for O is given by its sons F_i. So each *subtree* within a recognition tree, represents which entities (sons) are relevant for the recognition of another entity (parent). The structure of each subtree implies that all the feature variables F_i are conditionally independent given their parent O.

The topology of the visual network is mainly determined by the structure of all of its subtrees. So our methodology for structure improvement is based on validating the structure of each subtree in the network. This is done by testing the independence assumptions, and altering the structure if any of them is not satisfied.

Testing conditional independence

As in the Bayesian subjective approach, we start from subjective knowledge. We initially assume that the group of characteristics is independent given their cause (parent), but we then test out this assumption against the data. One way of testing for independences consists of calculating the correlation of pairs of the characteristics. Although a low correlation does not necessarily imply independence, it provides evidence that independence is a reasonable assumption to make; and, on the other hand, a high correlation indicates that the characteristics are not independent and our assumptions are invalid. If we find a high correlation between a pair of characteristics we must modify the structure of the probabilistic network. This could be done by introducing a link between the pair of dependent nodes, but it will cause the destruction of the tree structure. There are at least three other alternatives:

1. *Node elimination*: eliminate one of the dependent sons, including the subtree rooted at this node. We choose for elimination the variable with lower *discriminatory capability*. The concept of *probabilistic distance*[27] can be used for selecting the redundant node, in which case the node with the lowest value is selected for elimination.
2. *Node combination*: fuse the two dependent variables into one node which includes the information provided by the two dependent nodes, and link this new variable to the parent and sons of the original nodes.
3. *Node creation*: create a new variable which will be an intermediate node between the object and the two dependent features, linking this node to the object (parent) and the features (sons). This new node will, hopefully, make the two dependent variables conditionally independent, removing the requirement that they should be independent given their previous parent (O).

The three alternatives are illustrated in *Figure 13*. Each one implies an increasing level of complexity. In (1) we simply cut part of the network without any other effect: in (2) we need to recompute the probabilities in the links based on the joint distribution of the two variables; and in (3) we will usually require from some external agent to define the new variable, and we also need to obtain the required probabilities. Thus, for node combination we need to return to a parameter learning phase, and for node creation we need to return to the initial knowledge acquisition phase. Following our general principles, we try each alternative in sequential order, testing the system after each modification, until the performance is satisfactory.

For validating the independence assumption we use the correlation coefficients estimated from the sample data available. For this, we apply two different measures of association between random variables: *Pearson's correlation coefficient* (ρ), which indicates if there is a linear relationship; and *Kendall's correlation coefficient* (τ), which detects any increasing or decreasing trend curve present in the data. Although both measures do not correspond directly with dependency, in general they provide a reasonable approximation. So if either of them is above a certain threshold, we consider the variables not independent and we modify the dependency structure of the network accordingly.

Structure refinement algorithm

Based on the dependancy and discriminatory tests described in the previous section, we developed an algorithm for refining the structure of a visual KB represented as a probabilistic multitree. The algorithm tests the validity of the network using statistical data, it makes the possible improvements automatically, and it alerts us to the remaining problems which require external knowledge. The algorithm is the following:

Algorithm 3 ***Structure Refinement in Probabilistic Multitrees***

```
1. Decompose the network into $N$ probabilistic trees ($T_l$) by partitioning it in every leaf node with multiple parents. For every tree $T_l$ ($l = 1, 2, \ldots, N$) do 2 to 4.
2. Select the top subtree in the tree defining the root node as $O$ and its sons as $F_1, \ldots, F_n$.
3. For each pair $F_i$, $F_j$:
    3.1 Obtain the correlation coefficients $\rho$ and $\tau$.
    3.2 If $|\rho| < T_\rho$ and $|\tau| < T_\tau$ go to the next pair, otherwise continue.
    3.3 Node elimination – obtain the probabilistic distance ($D_P$) for $F_i$ and $F_j$ and eliminate the node with the lower $D_P$. Given that we are restricted to discrete variables and assuming conditional independence, we use the Patrick–Fisher[27] measure for $D_P$, which for each feature variable in the two class case is defined as:
```

$$D_P = \sqrt{\sum_j [P(F_j \mid O_1) - P(F_j \mid O_2)]^2} \qquad (23)$$

where F_j is the jth possible value of feature

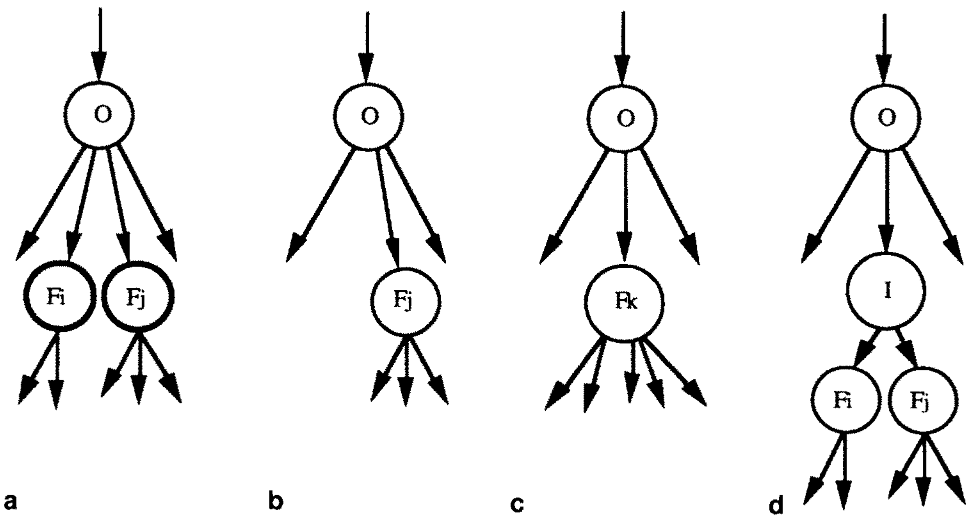


Figure 13 Subtree structure modification. The figure shows the original network (a) with two correlated variables (F_i, F_j), and the three possible network alterations: *node elimination* (b), *node combination* (c), and *node creation* (d)

variable F. Then test the system, if performance is satisfactory go to the next pair, otherwise go to 3.4.

3.4 Node combination – combine the nodes F_i and F_j into a single node F_k and obtain the conditional distributions $P(F_k \mid O)$ and $P(S_1 \mid F_k)$, where S_1 are the sons of F_i, F_j. Test the system, if performance is satisfactory go to the next pair, otherwise go to 3.5.

3.5 Node creation – create an intermediate node I by *consulting the expert*, and obtain the conditional distributions $P(F_i \mid I)$, $P(F_j \mid I)$, and $P(i/O)$.

4. Repeat 3 for every subtree in the tree, by selecting each of the child nodes as the parent, recursively, until the leaf nodes are reached.

In the previous algorithm, T_ρ and T_τ stand for the maximum thresholds for the respective correlation coefficients, and the words in italics indicate when external knowledge is required.

Although the algorithm seems complex, especially for large networks, several pragmatic considerations make its use practical. Firstly, most of the algorithm can be done without external intervention so it could be left to run without human supervision. Secondly, we expect that in practical systems the initial KB will pass the tests in most parts and only some critical elements will need alteration. Finally, it will be applied mainly during the initial knowledge acquisition phase so a real-time response is not strictly necessary.

APPLICATION TO ENDOSCOPY

Endoscopy is one of the tools available for diagnosis and treatment of gastrointestinal diseases. It allows a physician to obtain direct colour information of the inside surface of the human digestive system. Beside its diagnostic capabilities, endoscopes have therapeutic applications. They allow the removal of colonic polyps and other foreign bodies, and direct attack on bleeding lesions. The endoscope is a flexible tube with viewing capability. It consists of a flexible shaft which has a manoeuvrable tip. The orientation of the tip can be controlled by pull wires that bend the tip in four orthogonal directions (left/right, up/down). It is connected to a cold light source for illumination of the internal organs and has an optical system for viewing directly through an eye piece or on a TV monitor. The instrument has channels for transmitting air to distend the organ, a water jet to clean the lens and for sucking air or fluid. Additionally, it has an extra 'operating' channel that allows the passage of flexible instruments (biopsy forceps, cytology bushes, diathermy snares).

We are interested primarily in colonoscopy, which is especially difficult due to the complexity and variability of the human colon. The doctor inserts the instrument estimating the position of the colon centre (*lumen*) using several visual clues such as the darkest region, the colon muscular curves, the longitudinal muscle and others. If the tip is not controlled correctly it can be very painful and dangerous to the patient, and could even cause perforations on the colon wall. This is further complicated by the presence of many difficult situations such as the contraction and movement of the colon, fluid and bubbles that obstruct the view, pockets (*diverticula*) that can be confused with the *lumen* and the paradoxical behaviour produced by the endoscope looping inside the colon.

We are developing an expert system for colonoscopy[28]. Its primary objective is to help the doctor with the navigation of the endoscope inside the colon by controlling the orientation of the tip via the right/left and up/down controls. The control of the translation of the instrument (push/pull) and possible shaft rotation will still be controlled manually by the doctor. As well as a navigation system, it will also serve as an advisory system for novice endoscopists.

Probabilistic network for colonoscopy

Colonoscopy provides an interesting and challenging application for testing our methodology for visual recognition. Although it is a restricted domain, it has the complexity and uncertainty inherent in real-world image interpretation. A knowledge-base in colon endoscopy was compiled with the help of an expert colonoscopist[28]. Taking as a starting point this KB, we represented the visual part of it as a probabilistic network. In *Figure 14* we show the complete network representing our current visual knowledge for colonoscopy, derived from the qualitative rule-base. This network corresponds to a probabilistic multitree. The dotted lines indicate that the *relation nodes* (distance and topological) refer to the connected objects, and do not imply a probabilistic dependency.

Although this network includes most of the dependencies in the KB, not all of them are represented. For example, the presence of a *lumen* in the image can reduce the probability of having a *diverticula*, and *vice versa*, so we can argue that there must be a link between them or that they must be put together in a single node. A similar arugment can be made about some intermediate nodes such as *large dark region* and *small dark region*. We decided not to modify this model because we wish to maintain a multitree structure in which probability propagation is efficient, and we also want to to keep its semantic clarity by having a separate node for each object. We consider that these differences do not have a big impact in practice, due to the fact that we are only making bottom-up propagation and we are not interested in the posterior probabilities of intermediate nodes.

From this PN model for colonoscopy, we have focused on implementing the *lumen* recognition art, which is the most important for endoscope navigation. For this we are using the features obtained through two different low-level vision processes. The first one uses a 2D region-based segmentation algorithm to find the lumen, and the second uses 3D shape from shading information integrated in *gradient histogram* for esti-

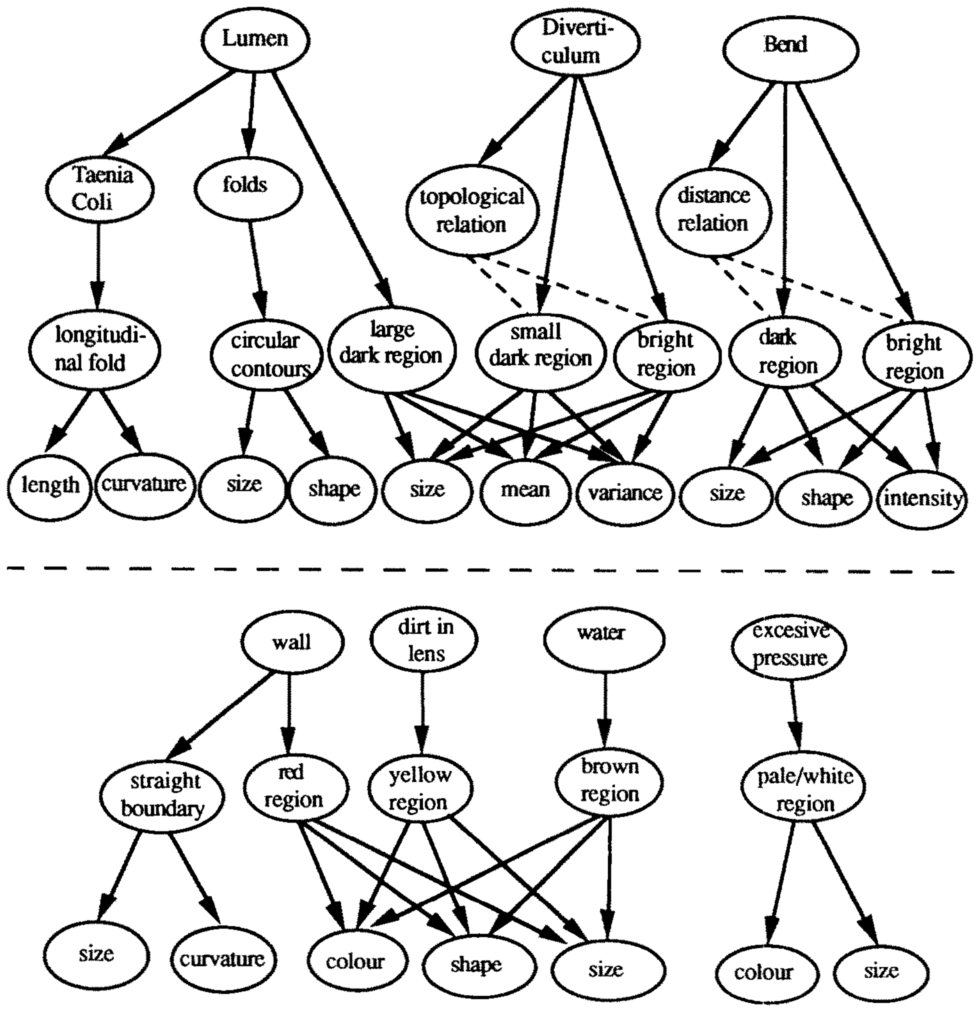


Figure 14 Probabilistic network representation of the visual model for colonoscopy

mating the *lumen* location. We will review both techniques in the following section.

Low-level processing

Due to the type of illumination of the endoscope, the darkest region in the image generally corresponds to the centre of the colon (lumen) because there is a single light source close to the camera. To obtain this information, Khan[29] has developed a new method for the extraction of the *dark region* in colon images. He uses a *Quadtree* representation in which the largest quadrant with a certain intensity level and variance is used as a seed region that is entended to 'cover' the lumen area. This technique is based on a Quadtree representation of the image and is suitable for parallel implementation in a pyramid architecture. From this process we obtain a dark region in the image with the following features: region size (SIZE), mean intensity level (MEAN), and intensity variance in the region (VAR).

A shape from shading technique suitable for endoscopy was developed by Rashid[30]. His model assumes a point light source very close to the camera which is a good approximation to a real endoscope. His shape from shading algorithm obtains the relative depth of the colon surface in the image. The depth map we obtain consists of a vector (p, q) per pixel that gives the orientation of the surface at this point with respect to two orthogonal axes (x, y) that are perpendicular to the camera axis (z). The surface orientation can vary from perpendicular to the endoscope's tip (camera) with $p = q = 0$, to parallel to the camera with p or q close to infinity. If we assume that the colon has a shape similar to a tube and in the image a section of the internal wall of this tube is observed, then a reasonable approximation of the position of the centre of the colon (*lumen*) will be a function of the direction in which the majority of the p–q vectors are pointing. For this we divide the space of possible values of p–q into a small number of slots and obtain a 2D histogram of this space, which we call a gradient or *pq histogram*[31]. The largest peak in this histogram will give an estimate of the position of the *lumen*. So from this process we obtain another estimate of the lumen, consisting of a *peak* in the *pq* histogram with the following attributes: number of pixels (NUMBER), distance from the centre (DISTANCE), and the difference in size to the next peak (DIFFERENCE).

The features obtained through these two low-level vision processes are integrated in high-level vision using our probabilistic reasoning techniques. The quantization of the variables results directly from the discrete nature of the measurement process. For example, since the seed regions are extracted using a regular quadtree decomposition, their size, measured as the pixel length one dimension, is always a power of 2. Similarly, the mean is quantized into integer values at the intensity resolution of the hardware we are using. The choice of resolution of the probability distributions must be a compromise between the speed of computation required, and the accuracy to which the required probabilities are computed. We have not investigated this relationship, but chosen a resolution which makes full use of the measured data.

Experimental results

To test the different aspects of our methodology, we have performed two sets of experiments. In the first one we used the dark region features to recognize the *lumen*, and differentiate it from small pockets in the wall, called *diverticula*, which can be confusing to the doctors. In this experiment we focus on the learning part, especially on structure learning. In the second experiment we combined the dark region and shape from shading algorithms to identify the lumen. This experiment focuses on the knowledge representation issues, using a more complex network that includes relational knowledge.

Experiment 1

For this experiment we used the part of our KB for *lumen* and *diverticula* recognition, obtaining the multitree shown in *Figure 15*. We analysed a random sample of about 125 colon images and generated the statistics from which the probability distributions were obtained using algorithm 2. We than evaluated the system with another sample of 90 colon images. Some typical cases, along with the *dark region* obained from low-level vision, and the interpretation given by the KB system, are shown in *Figure 16*. In *Table 1* we summarize the results for the images that were analysed.

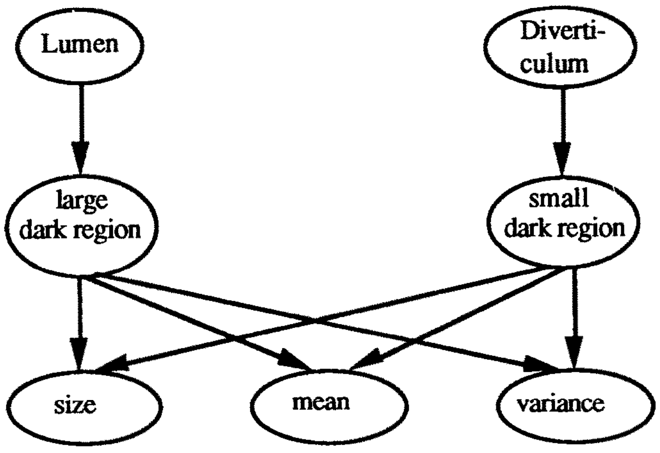


Figure 15 Probabilistic network for lumen and diverticula recognition

Table 1 Summary of the results of image interpretation and advice for 90 colon images (experiment 1)

Image Interpretation	Number	Percentage
Diverticula interpretation correct	40	85.1
Lumen interpretation correct	21	87.5
Expert's Rating		
Correct interpretation	72	80.9
Not correct interpretation	17	19.1
Predictive Score (Breir score)		
Total square error		11.6
Mean square error		0.13

The results presented in *Table 1* are divided into three parts. In the first one we show the performance of the lumen and diverticula recognition trees only, indicating the percentage of correct interpretation (true positives and true negatives) as evaluated by an expert colonoscopist. The second part of the table shows the performance for the interpretation and advice given by the system as rated by the expert endoscopist. The third section shows the system's performance by measuring the discrepancy between the system's prediction and the expert's assessment. This evaluation is done by computing the square difference between the posterior probability and the actual value according to the expert, to which we assign a probability equal to one. By summing this squared error for all the samples we obtain a measure of the system's predictive capability denominated 'Brier score'[12], which is defined as:

$$\text{Total square error:} \quad B = \sum_i (1 - P_i)^2 \qquad (24)$$

where P_i is the posterior probability of the *i*th sample being true. Dividing the Brier score by the number of samples (N) we obtain an estimate for the average predictive error which we define as:

$$\text{Mean square error:} \quad mB = \frac{B}{N} = \frac{\sum_i (1 - P_i)^2}{N} \qquad (25)$$

Although its performance was satisfactory for *lumen* and *diverticula* recognition, we thought it could still be improved. We were mainly interested in validating the

Table 2 Correlation between *lumen region* and *diverticula region* parameters

Correlation coefficients	*Size & mean*	*Size & variance*	*Mean & variance*
Lumen Subtree			
ρ	−0.146	0.264	0.482
τ	−0.089	0.116	0.342
Diverticula Subtree			
ρ	−0.039	0.147	−0.231
τ	−0.068	0.137	−0.161

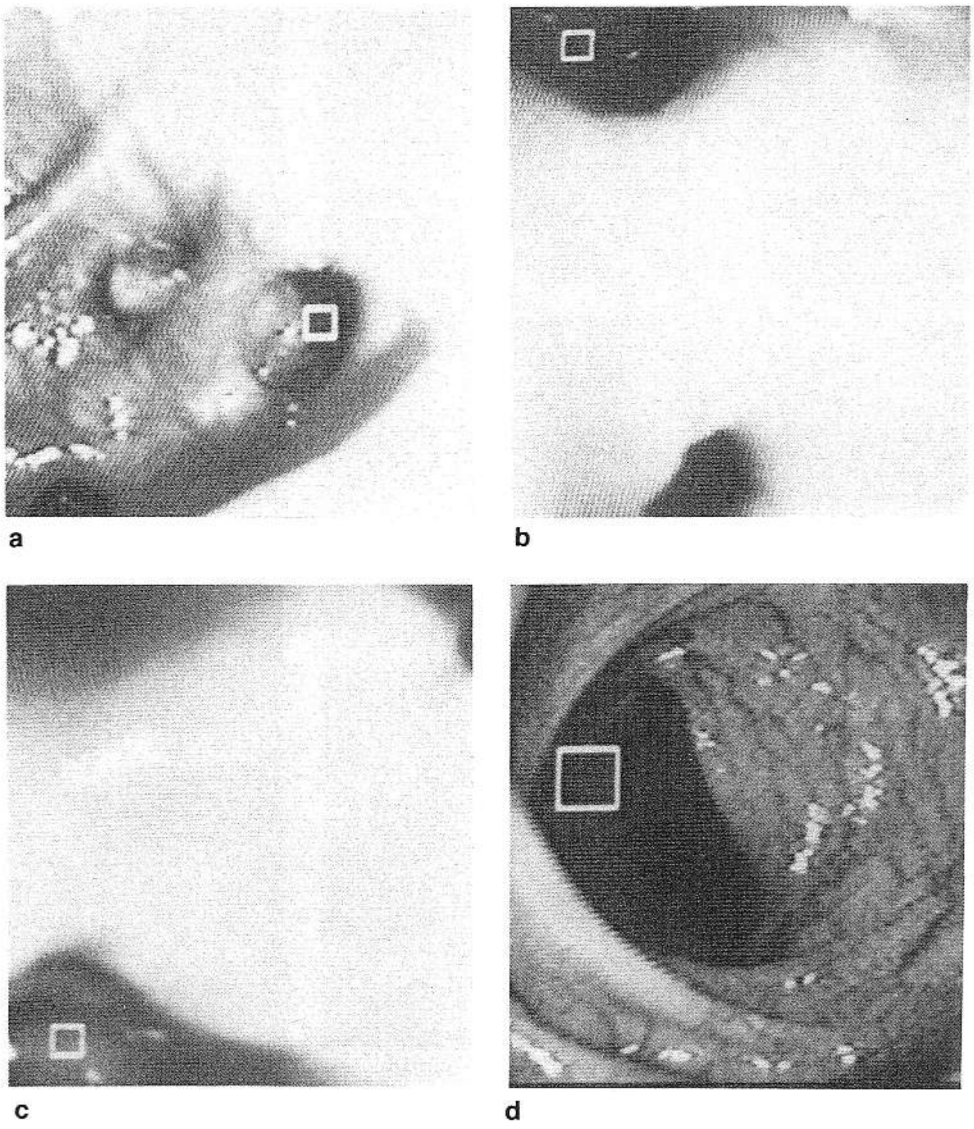


Figure 16 Interpretation given for different colon images (experiment 1). In each image, the 'dark region' obtained from low-level vision is depicted as a rectangle. Below we show the interpretation (including the posterior probability) as it was given by the system. Interpretations: (a) *diverticula* ($P = 0.93$); (b) *diverticula* ($P = 0.88$); (c) *diverticula* ($P = 1$); (d) *lumen* ($P = 0.68$)

independence assumptions that are represented in the network, and which in some cases seem difficult to justify. For this we applied the structure refinement algorithm using the same data as for the parameter learning phase. First, we calculated the correlation coefficients for the three pairs of variables given a *lumen region* and a *diverticula region* (*Table 2*).

In the lumen case there is a strong correlation between *mean & variance* that questions our independence assumption between these two variables. The other two values seem relatively low, so we will maintain that *size* is independent from *mean*, and from *variance*. These results are confirmed by plotting the sample points in a two dimensional *scatter diagram* for each variable pair[32]. As a first step, applying the *node elimination* technique, we eliminated one of the correlated variables. This resulted in two alternative trees for lumen recognition (*Figure 17*), one with only *size* and *mean* and the other with *size* and *variance*.

To select one of the these two alternatives, we calculated the probabilistic distance for *mean* and *variance*, and the results are shown in *Table 3*. It shows a greater probabilistic distance for variance, which

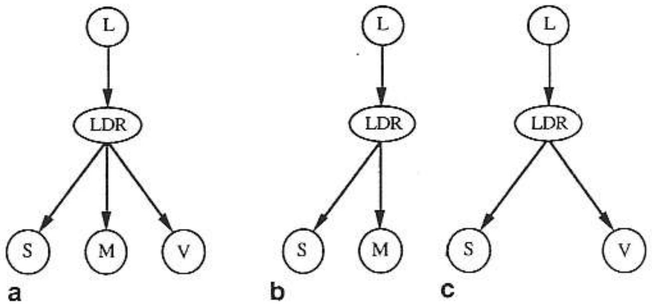


Figure 17 Alternative trees for lumen recognition. We show the original tree for *lumen* (a), and the two modified structures obtained by node elimination, eliminating *variance* (b) and eliminating *mean* (c)

Table 3 Probabilistic distance for *mean* and *variance* given a *lumen region*

Probabilistic distance (Patrick–Fisher)	*Mean*	*Variance*
D_p	0.221	0.265

Table 4 Performance results for different structures (experiment 1)

	All parameters (*S*, *M*, *V*)		*Eliminate V* (*S*, *M*)		*Eliminate M* (*S*, *V*)	
Image interpretation	No.	%	No.	%	No.	%
Diverticula interpretation correct	40	85.1	37	78.7	38	80.8
Total Diverticula	47	100	47	100	47	100
Lumen interpretation correct	21	87.5	21	87.5	22	91.6
Total Lumen	24	100	24	100	24	100
Expert rating						
Correct interpretation	72	80.8	68	76.4	73	82.0
Not correct interpretation	17	19.1	21	23.5	16	17.9

points to the structure in *Figure 17c* as the preferred alternative.

In this case of diverticula, all the correlations are low giving evidence of independence. So the diverticula subtree should remain with the same structure.

For comparison purposes, we tested the performance of the system with both alternative structures for lumen and diverticula, and the original structure. The performance evaluation was done with a similar sample of about 90 colon images, and the results are shown in *Table 4*. For lumen, although the performance is acceptable with the original structure (all parameters), there is a definite improvement when the *mean* parameter is eliminated, with a higher percentage of correct interpretations. Additionally, the network size was reduced and consequently object recognition was faster, so at the same time we obtain a more reliable and efficient system. Given the good performance obtained by node elimination, we did not consider in this case the alternative techniques. The results for diverticula also support our analysis by giving a better performance with all the parameters, compared with the structure with only two parameters (*S* & *M* or *S* & *V*). The overall performance is higher with *S* & *V* because for this experiment we eliminated one parameter in both trees, and *lumen* recognition improves. Thus, the best structure is given by using two parameters (*S*, *V*) in the *lumen* tree and three parameters (*S*, *M*, *V*) in the *diverticula* tree.

Experiment 2

In this second experiment we extended the probabilistic tree for lumen by including the features obtained in the *pq-histogram* from the shape from shading algorithm. We can think of the *pq-histogram* and dark region algorithms as two different ways of obtaining an *estimate* of the position of the *lumen*, which we should be able to combine in a probabilistic way. For representing this information in a probabilistic network we make two important considerations:

- The estimates provided by *pq-histogram* and dark region are *independent*.
- The proximity of the different estimates is an important factor for object recognition, so we add their *distance relation* as a node in the network.

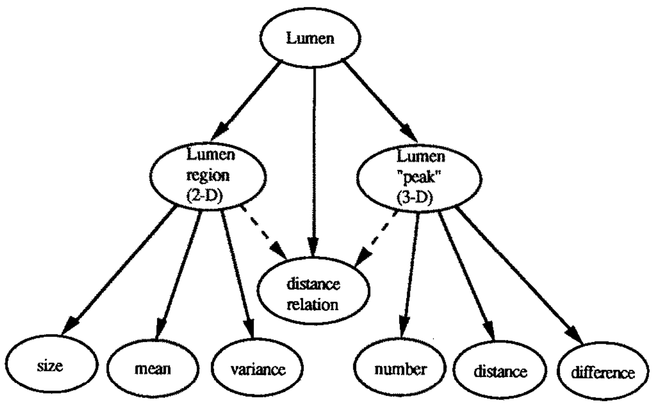


Figure 18 Probabilistic tree for lumen recognition, combining dark region (2D) and shape from shading (3D) features

Table 5 Summary of the results of image interpretation for 50 colon images (experiment 2)

Image interpretation		
	Number	Percentage
Lumen interpretation correct	34	97.1
NOT-Lumen interpretation correct	21	68.8
Predictive score (Breir score)		
Total square error		4.97
Mean square error		0.099

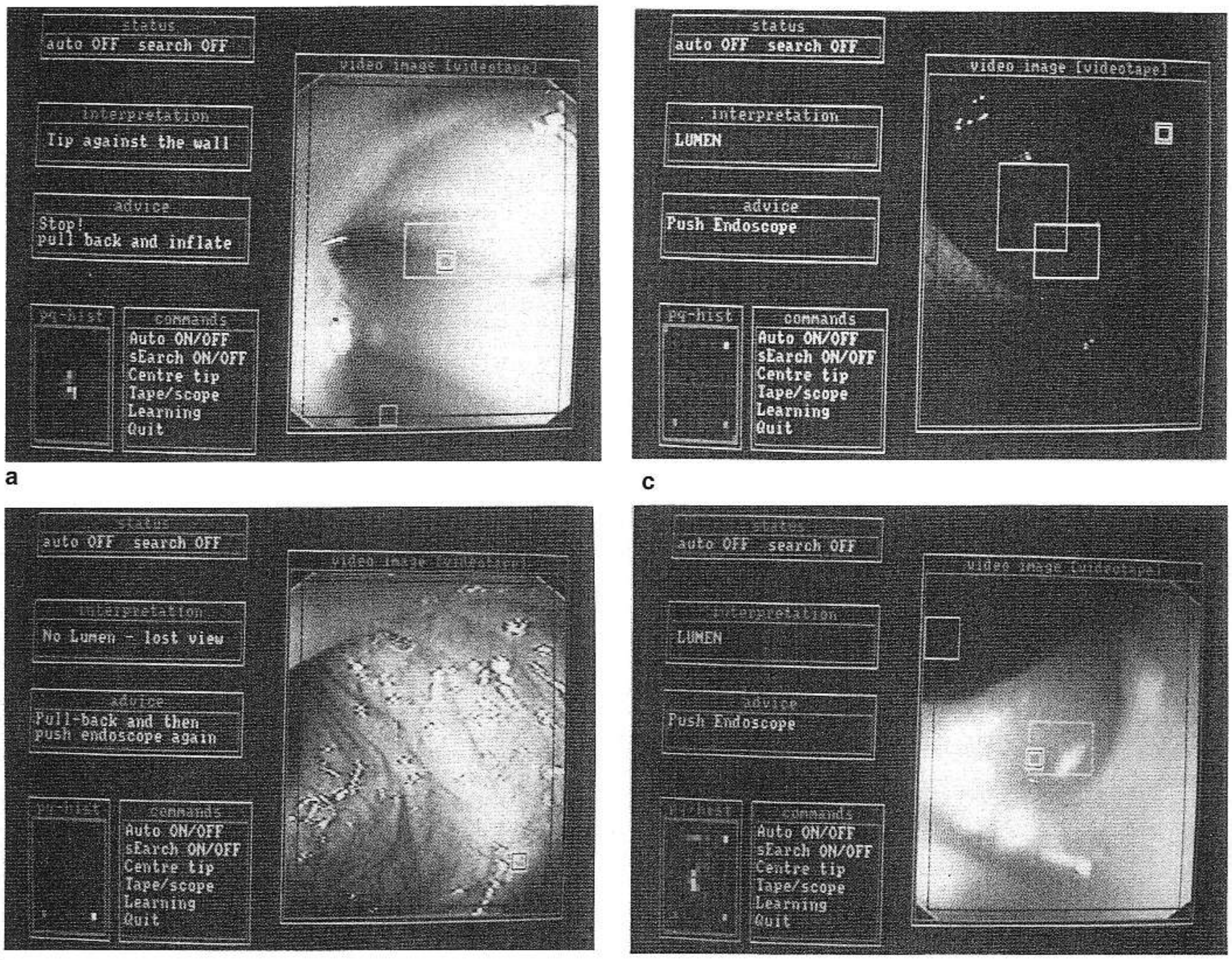
Based on these assumptions, the probabilistic network for *lumen* recognition will have the structure given in *Figure 18*.

As in the previous experiment, we trained the network from a sample of approximately 100 colon images, and then tested its performance with a different sample of 50 images. The results of this test are shown in *Table 5*. We note a definite improvement in the *lumen* recognition performance, which may be due to the fact that we are combining two independent techniques as well as using relational knowledge. The system predictive capability has also improved considerably, with a mean square error of less than 0.1.

The probabilistic network for lumen recognition has been integrated in an expert system for colonoscopy. It uses the results of image interpretation as input to a rule-based inference engine[28] to give advice to the physicians. In *Figure 19* we show some representative cases, including the image interpretation and advice given by the system.

CONCLUSIONS

We have proposed a general framework for representing uncertain knowledge in computer vision. Starting


a

status
auto OFF search OFF
interpretation
Tip against the wall
advice
Stop!
pull back and inflate
pq-hist
commands
Auto ON/OFF
sEarch ON/OFF
Centre tip
Tape/scope
Learning
Quit
video image [videotape]

b

status
auto OFF search OFF
interpretation
No Lumen - lost view
advice
Pull-back and then
push endoscope again
pq-hist
commands
Auto ON/OFF
sEarch ON/OFF
Centre tip
Tape/scope
Learning
Quit
video image [videotape]

c

status
auto OFF search OFF
interpretation
LUMEN
advice
Push Endoscope
pq-hist
commands
Auto ON/OFF
sEarch ON/OFF
Centre tip
Tape/scope
Learning
Quit
video image [videotape]

d

status
auto OFF search OFF
interpretation
LUMEN
advice
Push Endoscope
pq-hist
commands
Auto ON/OFF
sEarch ON/OFF
Centre tip
Tape/scope
Learning
Quit
video image [videotape]

Figure 19 Interpretation and advice given for different colon images (experiment 2). In each image, the 'dark region' is depicted as a rectangle and the lumen position inferred from the *pq-histogram* is shown as a double rectangle (the rectangle in the middle is fixed and only shown for reference). Below we show the interpretation as it was given by the system, and the posterior probability of lumen. Interpretations: (a) *wall* ($P = 0.08$); (b) *'lost view'* ($P = 0.30$); (c) *lumen* ($P = 0.99$); (d) *lumen* ($P = 0.97$)

from a probabilistic network representation, we developed a structure for representing visual knowledge, and techniques for probability propagation, parameter learning and structural improvement.

Our representation is based on a probabilistic network model whose structure corresponds to the qualitative visual knowledge in the domain. The network is divided into a series of layers, in which the nodes of the lower layer correspond to the feature variables and those of the upper layer to the object variables. The intermediate layers have nodes for other visual entities, such as parts of an object or image regions, or represent relations between features and objects. The links point from nodes in the upper layers toward nodes in the lower layers, expressing a causal relationship. The hierarchical visual network can be thought of as a series of trees, which we called a *multitree*. This representation has been extended to include *relational* and *temporal* knowledge, which are important in vision, especially for navigation.

By using both, probability propagation in trees and *conditioning*, we developed a probability propagation algorithm for visual recognition in a multitree. It is based on partitioning the network into a series of trees, instantiating the leaf nodes that correspond to the feature variables, and propagating their effect upwards to the object variables. This algorithm lends itself to a parallel implementation by processing all the trees in parallel, so the object's posterior probability can be updated in a time linearly proportional to the number of levels in the largest tree in the network.

We also developed algorithms for parameter and structure learning in multitrees. The required probabilities are estimated statistically from images of the intended application domain, and improved as more images are processed. The qualitative structure is initially derived from subjective knowledge about the domain, representing the relevant variables and their dependency relations. Then this structure is improved, by verifying the independence assumptions with statistical tests and modifying the structure of the network accordingly.

This framework provides an adequate basis for representing uncertin knowledge in computer vision, especially in complex natural environments. It has been tested in a realistic problem in endoscopy, performing image interpretation with good results. We consider that it can be applied in other domains, providing a coherent basis for developing knoweldge-based vision systems.

REFERENCES

1 Marr, D *Vision*, WH Freeman, New York (1982)
2 Feldman, J A 'A functional model of vision and space', in Arbib, M A and Janson, A R (eds.), *Vision, Brain and Cooperative Computation*, MIT Press, Cambridge MA (1987) pp 531–562
3 Pentland, P A 'Introduction: From pixels to predicates', in Pentland, A P (ed.), *From Pixels to Predicate*, Ablex, NJ (1986)
4 Binford, T O 'Survey of model-based image analysis systems', *Int. J. Robotics Res.*, Vol 1 No 1 (1982) pp 18–64
5 Rao, A R and Jain, R 'Knowledge representation and control in computer vision systems', *IEEE Expert*, Vol 3 No 1 (1988) pp 64–79
6 Wang, C and Srihari, S N 'A framework for object recognition in a visually complex environment and its application to locating address blocks on mail pieces', *Int. J. Comput. Vision*, Vol 2 (1988) pp 125–151
7 Wesley, L P 'Evidential knowledge-based computer vision', *Opt. Eng.*, Vol 25 No 3 (1986) pp 363–379
8 Perkins, W A 'Rule-based interpreting of aerial photographs using the lockheed expert system', *Opt. Eng.*, Vol 25 No 3 (1986) pp 356–363
9 Ohta, Y *Knowledge-based interpretation of Outdoor Natural Colour Scenes*, Pitman, Boston, MA (1985)
10 Hanson, A R and Riseman, E M 'Visions: A computer system for interpreting scenes', in Hanson, A R and Riseman, E M (eds.), *Computer Vision Systems*, Academic Press, New York (1978) pp 303–333
11 McKeown, D M, Harvey, W A and McDermott, J 'Rule-Based Interpretation of Aerial Imagery', *IEEE Trans. PAMI*, Vol 7 No 5 (1985) pp 570–585
12 Spiegelhalter, D J 'Probabilistic reasoning in predictive expert systems', in Kanal, L A and Lemmer, J F (eds.), *Uncertainty in Artificial Intelligence*, North-Holland, Amsterdam (1986) pp 47–68
13 Mackworth, A 'Adequacy criteria for visual knowledge representation', in Pylyshyn, Z (ed.), *Computational Processes in human vision: An Interdisciplinary Perspective*, Ablex, NJ (1988) pp 464–476
14 Agosta, J M 'The structure of Bayes networks for visual recognition', in *Uncertainty in Artificial Intelligence*, Elsevier, Amsterdam (1990) pp 397–405
15 Ng, K and Abramson, B 'Uncertainty management in expert systems', *IEEE Expert*, Vol 5 No 2 (1990) pp 29–48
16 Rimey, R D and Brown, C M 'Where to look next using a Bayesian net: Incorporating geometric relations', *Proc. 2nd Euro. Conf. on Comput. Vision*, Springer-Verlag, Berlin (1992) pp 542–550
17 Pearl, J 'On evidential reasoning in a hierarchy of hypothesis', *Artif. Intell.*, Vol 28 (1986) pp 9–15
18 Lauritzen, S L and Spiegelhalter, D J 'Local computations with probabilities on graphic structures and their application to expert sytems', *J. Roy. Statist. Soc. B*, Vol 50 No 2 (1988) pp 157–224
19 Pearl, J *Probabilistic Reasoning in Intelligent Systems*, Morgan Kaufmann, San Mateo, CA (1988)
20 Sucar, L E, Gillies, D F and Gillies, D A 'Handling uncertainty in knowledge-based computer vision', in Kruse, R and Siegel, P (eds.), *Symbolic and Quantitative Approaches to Uncertainty (ECSQAU-91)*, Springer-Verlag, Berlin (1991) pp 328–332
21 Cooper, G F 'The computational complexity of probabilistic inference using Bayesian networks', *Artif. Intell.*, Vol 42 (1990) pp 393–405
22 Duda, R O and Hart, P E *Pattern Classification and Scene Analysis*, Wiley, New York (1973)
23 Spiegelhalter, D J 'A statistical view of uncertainty in expert systems', in Gale, W A (ed.), *Artificial Intelligence and Statistics*, Addison-Wesley, Reading, MA (1986) pp 17–55
24 Spiegelhalter, D J and Lauritzen, S L 'Sequential updating of conditional probabilities on directed graphical structures', *Networks*, Vol 20 No 5 (1990) pp 579–606
25 Chow, C K and Liu, C N 'Approximating probability distributions with dependence trees', *IEEE Trans. Infor. Theory*, Vol 14 No 3 (1968) pp 462–468
26 Sucar, L E, Gillies, D F and Gillies, D A 'Handling uncertainty in knowledge-based sytems using objective probabilities', *The World Congress on Expert Systems*, Orlando, FL (December 1991) pp 952–960
27 Kittler, J 'Feature selection and extraction', in Young, T Y and Fu, K S (eds.), *Handbook of Pattern Recognition and Image Processing*, Academic Press, Orlando, FL (1986)
28 Sucar, L E and Gillies, D F 'Knowledge-based assistant for colonoscopy', *Proc. 3rd Int. Conf. on Ind. and Eng. Applic. of Artif. Intell. and Expert Systems*, Vol II, ACM Press, NY (1990) pp 665–672
29 Khan, G N and Gillies, D F 'A highly parallel shaded image segmentation method', in Dew, P M and Earnshaw, R A (eds.), *Parallel Processing for Vision and Display*, Addison-Wesley, Wokingham, UK (1989) pp 180–189
30 Rashid, H and Burger, P 'Differential algorithm for the determination of shape from shading using a point light source', *Image & Vision Comput.*, Vol 10 No 2 (1992) pp 119–127
31 Sucar, L E, Gillies, D F and Rashid, H U 'Integrating shape from shading in a gradient histogram and its application to endoscope navigation', *V Int. Symposium on Artif. Intell.*, Cancun, Mexico (December 1992)
32 Sucar, L E *Probabilistic Reasoning in Knowledge-based Vision Systems*, PhD Thesis, Imperial College, London (1992)