Distributed Bayesian Object Recognition A New Paradigm *

Isidore Rigoutsos
I. B. M. Corporation
T. J. Watson Research Center
P.O. Box 704
Yorktown Heights, NY 10598.

Robert Hummel
Courant Institute of Mathematical Sciences
New York University
251 Mercer Street
New York, NY 10012.

Email: rigoutso@watson.ibm.com, Telephone: (914) 784 7968, FAX: (914) 784 7455

November 1992

Abstract

The problem of model-based object recognition is a fundamental one in the field of computer vision and robotics, and we believe that it represents a promising direction for practical applications. For the purposes of this paper, we define a "realistic" object recognition system as a system that: (a) uses $2D$ intensity images, (b) can perform well with real-world inputs, (c) can recognize objects from a large database, (d) responds within reasonable time limits, and (e) the required processing power can be provided by readily available computational platforms. In this paper, we offer a new paradigm for performing "realistic" object recognition. First, we show how several intuitive notions in the context of geometric hashing can be translated into a well-founded Bayesian approach to object recognition; this interpretation, which is a new form of Bayesian-based model matching, leads to well-justified formulas, and gives a precise weighted-voting method for the evidence-gathering phase of geometric hashing. The second contribution of the paper is the description of a computational model for performing object recognition in a *distributed* fashion. Using the high-level language support provided by *Concert/C*, we demonstrate the validity of our paradigm by presenting a prototype system that has been implemented on a small farm of workstations. The resulting system is scalable, and can recognize models subjected to $2D$ rotation, translation and scale changes in *real-world* digital imagery. This is the first system of its kind that is scalable, can use large databases, handles noisy input data, and exhibits excellent performance with real-world scenes. The performance of the system is superior by a factor of 2 to that obtained for a similar system on a Connection Machine ($CM-2$).

Category: Pattern Recognition, Object Recognition, Real-time Vision and Architectures.

*During part of this research work, I. Rigoutsos was supported by an IBM Graduate Research Fellowship. R. Hummel was partially supported by AFAL Contract F33615-89-C-1087 reference 87-02-PMRE, and the kind hospitality of the Epidaure research group at INRIA-Rocquencourt.

Distributed Bayesian Object Recognition A New Paradigm

November 1992

Abstract

The problem of model-based object recognition is a fundamental one in the field of computer vision and robotics, and we believe that it represents a promising direction for practical applications. For the purposes of this paper, we define a "realistic" object recognition system as a system that: (a) uses $2D$ intensity images, (b) can perform well with real-world inputs, (c) can recognize objects from a large database, (d) responds within reasonable time limits, and (e) the required processing power can be provided by readily available computational platforms. In this paper, we offer a new paradigm for performing "realistic" object recognition. First, we show how several intuitive notions in the context of geometric hashing can be translated into a well-founded Bayesian approach to object recognition; this interpretation, which is a new form of Bayesian-based model matching, leads to well-justified formulas, and gives a precise weighted-voting method for the evidence-gathering phase of geometric hashing. The second contribution of the paper is the description of a computational model for performing object recognition in a *distributed* fashion. Using the high-level language support provided by *Concert/C*, we demonstrate the validity of our paradigm by presenting a prototype system that has been implemented on a small farm of workstations. The resulting system is scalable, and can recognize models subjected to $2D$ rotation, translation and scale changes in *real-world* digital imagery. This is the first system of its kind that is scalable, can use large databases, handles noisy input data, and exhibits excellent performance with real-world scenes. The performance of the system is superior by a factor of 2 to that obtained for a similar system on a Connection Machine ($CM-2$).

Category: Pattern Recognition, Object Recognition, Real-time Vision and Architectures.

1 Introduction

The problem of object recognition is a fundamental one in the fields of computer vision and robotics. We believe that a promising research direction in image analysis, and the one most likely to lead to industrial and commercial applications, is the area of object recognition where the search is confined within a finite set of observable models.

In all of the object recognition systems, one can distinguish four stages: data acquisition, feature extraction, matching and verification. Of these four stags, the matching is admittedly the most crucial component of any system. In the last decade, a number of techniques have been developed toward this end. But, in all cases there have been a trade-off between the reliability and the computation cost: techniques that produce very reliable results are computationally heavy, and vice versa. Most matching techniques have been based on cross-correlation ideas, tree- or graph-search, clustering, or indexing: the technique used generally depends on the types of features.

The matching technique should allow for partial occlusion, rotation, translation, and scale changes, as well as for small amounts of data perturbation. The output of the matching stage is a set of hypotheses regarding the identity of the models that are embedded in the scene. Associating a measure of belief to the answer(s) is also desirable: this measure allows the hypotheses' relative ranking. Together with the set of models, the matching stage also recovers the transformation that the corresponding model is assumed to have undergone.

Geometric hashing is a class of matching algorithms that facilitate object recognition in a computationally advantageous way. Similar to hashing methods for data retrieval, geometric hashing uses functions of geometric features in order to index from observed image data into a set of geometric models. Unlike typical hashing methods, the hash functions for geometric applications should not be random – it is important that small perturbations of the positions of features lead to a graceful degradation in the recognition capabilities. Accordingly, geometric hashing functions are based on transformation invariants, and depend on the class of transformations over which recognition is desired. Errors in the position and extraction of features can lead to perturbations of the hash values, and degrade recognition performance.

For the purposes of this discussion, we define a "realistic" object recognition system as a system that: (a) uses $2D$ intensity images, (b) can perform well with real-world inputs, (c) can recognize objects from a large database, (d) responds within reasonable time limits, and (e) the required processing power can be provided by readily available computational platforms.

In this paper, we describe a new paradigm for performing realistic object recognition using geometric hashing ideas and a distributed model of computation. In particular, we show how several intuitive notions in the context of geometric hashing can be translated into a well-founded Bayesian approach to object recognition. This interpretation, which is a new form of Bayesian-based model matching, leads to well-justified formulas, and gives a precise weighted-voting method for the evidence-gathering phase of geometric hashing. These formulas replace traditional heuristically-derived methods for performing weighted voting, and also provide a precise method for evaluating uncertainty. The second contribution of the paper is the description of a computational model for performing object recognition in a *distributed* fashion. We demonstrate the validity of our paradigm by presenting a prototype system that has been implemented on a small farm of workstations. The needed high-level programming language support is provided by Concert/C [1, 2] The resulting system is scalable, and can recognize models subjected to $2D$ rotation, translation and scale changes in real-world digital imagery. This is the first system of its kind that is scalable, can use large databases, handles noisy input data, exhibits excellent performance with real-world scenes, and makes use of readily available computational resources. Using a very coarse grain of parallelism, the performance of the system is superior by a factor of 2 to that obtained for a similar system on a Connection Machine ($CM-2$) [51].

A detailed description of the analysis that leads to the Bayesian interpretation of geometric hashing can be found in the PhD thesis of Rigoutsos [51]. The presentation here summarizes and simplifies that analysis. Although the use of weighted voting in geometric hashing is a logical extension, the Bayesian interpretation is startling in that it leads to well-justified formulas, indicating precisely the manner in which weighted voting should be incorporated. It is more usual, with fuzzy systems for example, to use heuristically-based evidence combination formulas. This paper presupposes a basic understanding of the geometric hashing algorithm. Summaries and discussions of the method may be found in [34, 41, 44, 52, 51, 54, 55, 56].

In section 2 we present a survey of the most representative object recognition systems and techniques that have been developed during the last two decades. Section 3 introduces the concept of weighted voting in the context of geometric hashing, while section 4 presents a Bayesian formulation of the geometric hashing algorithm. The density of entries in the hash space is discussed in section 5, and the description of the resulting bayesian object recognition algorithm is given in section 6. In section 7 we describe a distributed-computation model for performing Bayesian object recognition, together with a brief description of the Concert/C language for programming distributed systems. The paper concludes with a presentation of the implementation and the experimental results (section 8). We emphasize that our system is "complete" in the sense that the feature extraction process is automated.

2 A Survey of Object Recognition Systems and Techniques

In this section, we briefly describe some of the most representative object recognition systems that have been developed during the last two decades.

One of the earliest object recognition systems was developed by Roberts [57]. The system was able to recognize convex polyhedral objects under the weak perspective transformation. It controlled and pruned the search by considering only vertices that were connected by an edge, and thus could not handle occlusion. Unlike Roberts' system, the models in ACRONYM [13] were generalized cylinders. ACRONYM used symbolic constraints to control and effectively prune the search, and could handle both noise and occlusion.

A related approach to that of ACRONYM's was taken in Goad's system [27]. The system used quantitative (as opposed to symbolic) constraints to control the search. Goad's system also introduced the notion of the two stage (off-line stage, on-line stage) recognition algorithm, where data precomputed during a first phase (off-line) are used during the phase of actual recognition (on-line) in order to speed up the processing.

The use of geometric constraints (such as distance and angle) as an efficient way for pruning the search while matching image and model features, was advocated by Bolles in his LFF and 3DPO systems [10, 11]; LFF is used to recognize $2D$ objects from intensity images, whereas 3DPO is used for recognition of $3D$ objects from range data. Geometric constraints are also used in the RAF system of Grimson [29, 30]. In Grimson's system, the search is structured around an interpretation tree, and exhibits exponential time complexity if the input image (intensity data) includes spurious data. More recently, the BONSAI system [23] exploits *unary* and *binary* constraints to control the search of the interpretation tree, and prune the search space: the input to BONSAI comprises range images of parts that have been designed using a CAD tool.

The HYPER system of Ayache and Faugeras [3] also belongs to the category of systems that attempt to determine correspondences between sets of model and image features. HYPER is used to recognize $2D$ objects from intensity images. However, its success is dependent on the quality of the polygonal approximations of the input image's contours. Furthermore, it is sensitive to noise and does not deal with the occlusion of edges.

Lowe's system, SCERPO [47], is a complete object recognition system that recognizes polyhedral $3D$ objects from intensity images, under the perspective transformation. SCERPO attempts to reduce the complexity of the search by performing perceptual groupings of image features. However, it typically deals with only one or two models in the model database at any given time.

More recently, work by Kak [37] shows that efficient algorithms can prove beneficial in reducing the complexity in the case of systems that search over the sets of features. In particular, Kak uses bipartite matching in conjunction with the notion of discrete relaxation to perform recognition of $3D$ objects using the output of a structured-light scanner.

In addition, a number of other systems have been developed that search the space of allowed transformations. The classic representative of this approach is the generalized Hough transform [4, 5, 65]: the method is a generalization of the Hough transform [33] and is used to detect arbitrary shapes. In the generalized Hough transform framework, the recognition of objects is achieved by recovering the transformation that brings a large number of model features in correspondence with image features. The transformation is described in terms of a set of transformation parameters, and votes for these parameters are accumulated by hypothesizing matchings between subsets of model and image features. The generalized Hough transform requires the quantization of a range of values for each of the parameters, thus resulting in decreased accuracy. The space requirements are exponential in the number of the parameters.

The system by Mundy and Thompson [48, 49] uses large Hough tables to perform recognition of $3D$ objects from $2D$ input data, under the weak perspective transformation. To constrain the space of possible transformations, the system uses the notion of the "vertex-pair." A vertex pair consists of two vertices and the two edges forming one of the vertices. An improved version of Mundy and Thompson's system [60] remedies the problem of fixed parameter quantization by iteratively refining the quantization around volumes of interest (histogram peaks), until the required precision was achieved. A similar system is the one of Linnainmaa [46] which introduced the notion of the "triangle-pair;" triplets of vertices from the image are matched against triplets of model vertices in order to hypothesize a transformation under the perspective projection model; however, the system provides multiple alternatives that require examination.

The approaches of the last three systems can be considered as special cases of a more general scheme called "*alignment*" [69]. In alignment, one seeks a model from the model database together with a transformation from the allowed class of transformations such that the object being viewed and the transformed model are in correspondence; for those transformations where the number of corresponding features exceeds a certain threshold, a verification procedure is invoked. Another alignment-based system, RANSAC [22], is used to recognize objects under perspective transformation, for a known camera position. Huttenlocher's ORA system [35] on the other hand, performs recognition assuming the weak perspective transformation model. More recently, alignment ideas have been combined with efficient string matching in order to perform *unoccluded* polygonal object recognition [58].

The approach of Ullman and Basri [70] is considerably different. The basic idea here is that each *topologically different* model view can be expressed as a linear combination of a small number of $2D$ views of the model. The method assumes that the transformation of the model to the scene can be modeled by an orthographic projection, and can handle $3D$ rigid as well as non-rigid transformations of the models. Scene clutter causes considerable problems. The scheme has been treated mostly theoretically. Some preliminary results indicate reasonable performance with databases containing a handful of objects.

Another general scheme that also involves search over the space of transformations is the geometric hashing scheme. Although based on the same geometric principles as alignment, geometric hashing differs from alignment in the algorithmic approach.

The following survey is mostly limited to treatments that led directly to the geometric hashing method or used the geometric hashing terminology. It is clear that related ideas and essentially equivalent concepts occurred frequently in the development of object recognition systems. For example, the "feature sphere" used in Chen and Kak's $3D$-POLY [18] is essentially a hash function that permits indexing into a smaller set of models. Likewise, work by researchers at IBM T.J. Watson Research Center has been based on ideas of indexing into model bases for many years [9, 14, 15].

The idea of geometric hashing, at least in its modern incarnation, has its origins in work of Professor Jacob Schwartz [36]. The first efforts were concentrated on the recognition of $2D$ objects from their silhouettes. Hence, efficient curve-matching techniques were developed. The use of "footprints" to describe properties along the curves was later extended by Wolfson and Hong [32] and resulted in a recognition system that was able to recognize about ten $2D$ objects partially occluding each other. The objects were taken from a library of a hundred models, and recognition was performed allowing planar rigid motion (rotation and translation). A landmark in the application of curve matching and combinatorial optimization methods was their use to assemble (graphically rather than physically) all the pieces of two hundred-piece commercial jigsaw puzzles, from separate photographs of their individual pieces [72]. The assembly was based on shape information only. In all two-dimensional curve-matching work, footprints were used to limit the number of candidate curves accessed by the matching system.

However, hash functions (still called *footprint information*) were much more essential when used for $3D$ curve matching obtained from depth data of objects. Using depth data obtained from a fast but approximate depth sensor [16], Wolfson and Kishon developed a practical method for locating and matching curves on rigid $3D$ objects [38], and extended the work by using a different hashing technique [59]; all $3D$ curve-matching systems used measures of the local curvature as index values into a table.

Application of the geometric hashing idea as an approach to model-based vision object recognition was introduced by Lamdan, Schwartz, and Wolfson. Much of the work is summarized in the dissertation of Lamdan [40]. Algorithms were developed for recognition of flat rigid objects assuming the affine approximation of the perspective transformation [44, 45] and the technique was also extended to the recognition of arbitrary rigid $3D$ objects from single $2D$ images [42].

Stein and Medioni [63] present a system for the recognition of planar objects from intensity images. The system uses a hash table that contained the gray-encodings of groups of consecutive edge segments ("supersegments"), of varying cardinalities. For recognition of general $3D$ objects from single $2D$ images, promising results have been obtained in the dissertation of Lamdan [40], where many viewpoint-centered models are generated of simple $3D$ models, and the work of Gavrila and Groen [26], who generate viewpoint-centered models based on experiments that determine limits of discriminability. Stein and Medioni's TOSS system [64] uses "structural hashing" to recognize $3D$ shapes from dense range data from which characteristic curves and local differential patches are extracted. The method allows only for rigid transformations (rotation and translation), and despite the use of high-dimensional indices, the verification stage is very costly.

Forsyth *et al.* [25] present and use descriptors based on *pairs* of planar curves; the descriptors are invariant under affine and perspective transformations. They obtain good results, but their method is sensitive to occlusion and its performance depends strongly on the quality of the segmentation.

For the recognition of rigid $3D$ curves extracted from medical imagery, Guéziec makes use of hashing methods to speed matching based on spline curve approximations [31]. For recognition of $3D$ objects from range data, Flynn and Jain [24] use hashing and local feature sets to generate hypotheses (without a voting procedure), and report a more efficient search than a constrained search (hypothesize and verify) approach, when applied to two dozen models. Representing $3D$ objects by means of their characteristic view, and regarding object recognition as a graph matching problem, Sossa and Horaud [61] develop hash functions for graphs to provide improved matching capabilities.

All of the systems that have been described so far, as well as those based on the geometric hashing scheme, typically represent the database models using a small number of homogeneous, local features. These features "define" the objects. Furthermore, the objects under consideration are treated in isolation from the rest of the scene. Unlike these systems, CONDOR [66] is the first system that takes the approach of performing context recognition first, and then instantiates the individual components. Natural objects such as *sky*, *ground*, *foliage* are included in the system's vocabulary. A special-purpose database contains all the necessary information about the world. The introduction of context results in increased flexibility at the expense of a major increase of the computational complexity. The input to the system can be any combination of intensity, range, color or other data modalities. The output is a labeled $3D$ model of the input image, with the labels referring to the object classes that can be recognized by the system.

The system by Swain [67] can recognize deformable objects and substances described by mass nouns by making use of color information. Thus it is similar in flavor to CONDOR. However, its use of precomputed invariants for the different database models in a two-stage algorithm, brings the system closer to the ones that are based on hashing/indexing ideas. Notably, Swain's system can recognize objects independent of background and viewpoint variations, occlusion, scale, and lighting conditions.

Vayda and Kak's INGEN system [71] performs object classification based on the overall shape of the object. For certain object recognition tasks it suffices to categorize the objects based on their general shape (e.g. parallelepiped, cylinder, etc.) and independently of their size. Because of the large variations in size, feature-based object recognition techniques are not easily applicable. INGEN uses a *hypothesize-and-verify* approach to determine the pose and generic shape of objects from range data. Hypotheses generated for each region in the segmented range data can be combined using either information contained in a combinability graph, or proximity and continuity heuristics. Use of the combinability graphs controls the combinatorial explosion by efficient pruning of the search space.

Another system with no knowledge of a *geometric* or *structural* model for each of the database objects is the one by Stark and Bowyer [62]. In their system, object classes are described in terms of the functional properties shared by all the $3D$ objects in the class. The various functional properties are represented using procedural knowledge. The system has been successfully tested with a database of 100 objects belonging to the "chair" class; the output of a CAD tool was used to provide the test input to the system.

Dickinson [20] presents yet another approach to $3D$ object recognition from intensity images. His system uses a small set of volumetric primitives which can be assembled to form the objects that can be recognized. An important component to the system is a hierarchy of $2D$ features (such as contours, faces, groups of faces) that are generated by projecting the primitives based on a set of viewer-centered orientations; conditional probabilities capture the relation between nodes at different levels of the hierarchy, and can be computed off-line. The number of orientations is fixed and thus independent of the number of models the system can recognize. During recognition, the system uses a bottom-up approach and precomputed conditional probabilities to extract primitives in the input image, as well as the connectivities of the primitives. Using the primitive and connectivity information, the system indexes into the model database to recover the identity of the viewed object. Bergevin's PARVO system [6] takes a similar approach to that of Dickinson's but makes use of "geons" [7] as the modeling primitives.

Kriegman and Ponce's approach [39] also exploits the relation between the shape of intensity image contours and the models of $3D$ objects of revolution. Under the assumption that the image contours are the projections of either surface discontinuities or occluding contours, Kriegman and Ponce use *elimination theory* to construct the implicit equations of the contours under perspective and weak perspective projections; the equations are parameterized by the object's position and orientation. In the current system, edge segments are grouped into contours manually, and no extraneous data are fed into the algorithm. Also, contours that are neither surface nor occluding discontinuities are manually removed. Although they obtain good results, the success of the approach relies heavily on the quality of segmentation.

In an effort to further the improve response time, parallel implementations of geometric hashing, for a Connection Machine, have been attempted by Medioni [12] and by Rigoutsos [52, 56].

Some general observations can be made with regards to the characteristics of the above described systems. Typically, all of them make use of small model databases. With the exception perhaps of CONDOR, the test inputs are simple scenes, where the number of extraneous features typically does not exceed the number of features belonging to the embedded model. Almost no system exploits the parallelism that may be inherent in the employed algorithm, whereas a number of systems do not allow for any level of occlusion. In almost all cases, there is a lack of a *natural* way to quantify the confidence the system has in the reported answer. Very few systems make use of Bayesian reasoning, e.g. SCERPO, CONDOR. Finally, an observation that applies to practically all of the systems is that segmentation *is* the limiting factor: all of the systems/techniques perform well with ideal input data but their performance degrades more or less gracefully in the presence of input noise. The PhD thesis of Rigoutsos [51] presents a first attempt to successfully deal with all of these issues: a complete object recognition system able to recognize aircraft and automobile models in real-world photographs is described therein. The system is implemented on a Connection Machine ($CM-2$).

There have been numerous efforts to add an error model to geometric hashing, and to investigate its performance in the presence of positional noise of the features. Costa *et al.* [19] investigate the variation of the standard hash functions in the presence of noise, and suggest a weighted-voting scheme. Lamdan and Wolfson [43] investigate both analytically and empirically the false-alarm rate, and conclude that acceptable filtering is possible, although a degradation in performance can be expected for affine-invariant matching. Grimson and Huttenlocher [28] give rather pessimistic predictions for affine-invariant matching using geometric hashing. Gavrila and Groen [26] report good filtering capabilities of similarity-invariant model matching in the presence of noise. Rigoutsos and Hummel [53, 54] look at error rates in the presence of noise for both similarity and affine invariance, and conclude that use of weighted voting can greatly improve performance.

3 Weighted Voting in Geometric Hashing

We suppose that we are given m models $M_1, M_2, M_3, \ldots, M_m$, each consisting of a pattern of n points, say $\mathbf{q}_{k,1}, \ldots, \mathbf{q}_{k,n}$ for model M_k, with each $\mathbf{q}_{k,i} \in \mathbb{R}^2$. For similarity-invariant recognition, the hash table consists of a collection of $mn(n-1)(n-2)$ entries, each of the form (x, y, M_k, i, j, ℓ). Each entry means that model M_k, using basis $(\mathbf{q}_{k,i}, \mathbf{q}_{k,j})$, gives rise to a hash value at location (x, y) when the coordinates of $\mathbf{q}_{k,\ell}$ are computed in the appropriate coordinate system. We typically use the hash function and coordinate system as depicted in Figure 1. There will be a separate entry in the hash table for every ordered triple of distinct points (i, j, ℓ) and every model M_k.

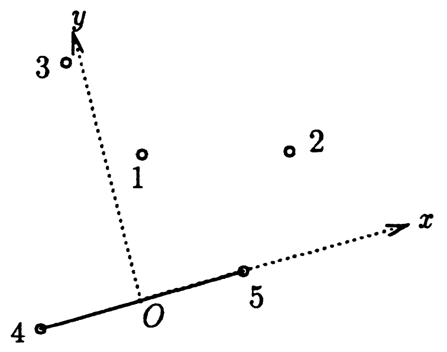

Figure 1. Hash coordinate computation using a two-point basis.

During the recognition phase, a collection of scene points $\mathbf{p}_1, \ldots \mathbf{p}_S$, are observed, with each $\mathbf{p}_\ell \in \mathbb{R}^2$. Candidate basis pairs are selected, say $\mathbf{p}_\mu$ and $\mathbf{p}_\nu$, and the remaining scene points are used to compute hash locations in the coordinate system determined by the basis $(\mathbf{p}_\mu, \mathbf{p}_\nu)$. When a point hashes to a location (u, v), we desire to locate all entries in the hash table that lie nearby the point (u, v), and register a weighted vote for each such entry. That is, every hash table entry of the form (x, y, M_k, i, j) should receive a weighted vote when (x, y) is close to (u, v). The weight of the vote should depend on the distance between (u, v) and (x, y). Generally, the weight of the vote should drop as the distance increases. This process is more fair than the simple-minded binning strategy which registers a single vote whenever (u, v) lands in the same quantized bin as (x, y). There is a more graceful degradation in the total number (i.e., weight) of votes for a model/basis combination as noise corrupts the positions of the hash locations.

What weight should we give to a vote for a record (x, y, M_k, i, j, ℓ) given a hash to location (u, v)? Several heuristic functions are easy to define. We could choose some decreasing function of the Euclidean distance between (x, y) and (u, v), such as the inverse of the distance, to determine the weighted vote. To simplify the process, the weighted vote could be a linear function of the Euclidean distance, dropping to and being clipped at zero beyond some threshold distance. Alternatively, and closer to our actual practice, we could analyze the expected distribution about (x, y) of the expected hash location of a noise-corrupted model M_k, using basis pair (i, j), embedded into the scene, based on expected variations in the positions of the basis points and the point that is being hashed, to define a statistical covariance C. The weighted vote in response to a hash to location (u, v) can then be related to the Mahalanobis distance $(d_x, d_y)C^{-1}(d_x, d_y)$, where $(d_x, d_y) = (u - x, v - y)$.

What is startling, and what we will show in the next section, is that a Bayesian formulation may be used to provide a precise formula, using an exponentially decreasing function of a scaled Mahalanobis distance, providing a well-justified weighted voting formula.

4 A Bayesian Formulation

Recall that the model database consists of models $\{M_k\}$, for $k = 1, \ldots, m$, and that we are also given a scene with a set of s points, $\mathcal{S} = \{\mathrm{p}_\ell\}_{\ell=1}^{S}$. We assume that two points of $\mathcal{S}$ are chosen as a basis pair, say, $\mathcal{B} = \{\mathrm{p}_\mu, \mathrm{p}_\nu\}$. Thus, given $\mathcal{S}$ and $\mathcal{B}$, we wish to determine if a model is present with $\mathcal{B}$ as a basis.

Throughout, we assume that $\mathcal{B}$ is fixed. Thus our formulation depends on a fixed chosen basis set in the scene. If an adequate model is not found by a verification step as a result of using $\mathcal{B}$ as a basis, then the entire analysis must be repeated with other choices for the basis pair. Further, we set

$$\mathcal{S}' = \mathcal{S} - \mathcal{B};$$

i.e., $\mathcal{S}'$ comprises the points of the scene less the two points in $\mathcal{B}$. The set $\mathcal{S}'$ provides the evidence that will be used to determine if a model is present.

We pose the following query: What is the probability that model M_k is present, with points i and j of the model respectively matching $\mathbf{p}_\mu$ and $\mathbf{p}_\nu$ of the chosen basis set $\mathcal{B}$ in the scene based on the information given by $\mathcal{S}'$? We use the following notation:

- (M_k, i, j) means that model M_k is present with point i of M_k matching basis point $\mathbf{p}_\mu$ and point j of M_k matching point $\mathbf{p}_\nu$;
- $\mathcal{B}$ means that the basis pair $\mathcal{B} = \{\mathbf{p}_\mu, \mathbf{p}_\nu\} \subset \mathcal{S}$ has been chosen and fixed;
- $\mathcal{S}'$ means that a collection of hash values (corresponding to the points of set $\mathcal{S}'$) are computed and present, corrupted by noise, relative to the fixed basis set $\mathcal{B}$.

Using this notation, we wish to compute

$$\Pr\left((M_k, i, j)\,|\mathcal{B}, \mathcal{S}'\right) \tag{1}$$

which is the probability that model M_k is present, with points i and j of the model respectively matching $\mathbf{p}_\mu$ and $\mathbf{p}_\nu$ of the basis set $\mathcal{B}$, based on the information from the hash locations as computed by the scene points $\mathcal{S}'$ relative to the selected basis set.

A maximum likelihood approach to object recognition results if we ask to find the maximum of Eqn. (1) over all possible M_k, i, and j. The resulting model/basis combination is the most likely match given the basis $\mathcal{B}$ and the collection of hash values generated from $\mathcal{S}'$. If there is no match, then the probability value of even the maximum winner will not be large. If there are several possible matches, then several model/basis combinations will share a large part of the total probability. Because we intend on passing winning combinations to a verification phase, it suffices to find the few combinations that lead to the largest probabilities. We will determine the relative probabilities, and not the actual values.

Next, we make certain conditional independence assumptions. Let $\mathbf{p}_\xi$ be one of the points of $\mathcal{S}'$, and let $\mathcal{S}''$ be any nonempty subset of $\mathcal{S}'$ not containing $\mathbf{p}_\xi$. We make the assumption that

$$\Pr\left(\mathbf{p}_\xi|\mathcal{S}'', (M_k, i, j), \mathcal{B}\right) = \Pr\left(\mathbf{p}_\xi|\,(M_k, i, j), \mathcal{B}\right) \tag{2}$$

for every possible (M_k, i, j). These are conditional independence assumptions, and the meaning can be summarized as follows. Under the assumption that some model M_k matches points in a scene with points i and j in the model matching the basis $\mathcal{B}$ in the scene, then the expected probability distribution of a hash point relative to the basis $\mathcal{B}$ is independent of any collection of other hashed values, whether they corroborate the model or not. That is, once the assumption is made that a match occurs, then the density function for hash values is fixed; namely, there is a large expectation of hash values near the points in the hash table where (M_k, i, j) hash entries occur, and a uniform density (or some fixed density) elsewhere, regardless of what other hash values are known to occur.

With the above assumptions, and using standard probabilistic derivations based on Bayes' theorem [17], the probabilities of Eqn. (1) may be rewritten as

$$\Pr\left((M_k, i, j) \,|\mathcal{B}, \mathcal{S}'\right) = K \cdot \Pr\left((M_k, i, j) \,|\mathcal{B}\right) \cdot \prod_{\mathbf{p}_\xi \in \mathcal{S}'} \frac{\Pr\left((M_k, i, j) \,|\mathcal{B}, \mathbf{p}_\xi\right)}{\Pr\left((M_k, i, j) \,|\mathcal{B}\right)}. \tag{3}$$

The constant of proportionality K will be independent of (M_k, i, j), and it is noteworthy that only the conditional independence assumptions (Eqns. (2)) are necessary, and not unconditioned independence assumptions. Applying Bayes' theorem once more, each term in the product of the right hand side of Eqn. (3) can be written as

$$\frac{\Pr\left(\mathbf{p}_\xi | \left(M_k, i, j\right), \mathcal{B}\right)}{\Pr\left(\mathbf{p}_\xi | \mathcal{B}\right)}. \tag{4}$$

Since the logarithm function is monotonic, maximizing the probabilities (1) over model/basis combinations is equivalent to maximizing the logarithms of those probabilities. We can thus apply the logarithm to both sides of (3), noting that the constant of proportionality becomes a constant additive factor, whence we see that we must maximize

$$\log\left(\Pr\left((M_k, i, j) \,|\mathcal{B}\right)\right) + \sum_{\mathbf{p}_\xi \in \mathcal{S}'} \log\left(\frac{\Pr\left(\mathbf{p}_\xi | \left(M_k, i, j\right), \mathcal{B}\right)}{\Pr\left(\mathbf{p}_\xi | \mathcal{B}\right)}\right) \tag{5}$$

over all possible model/basis combinations.

This is in essence what geometric hashing does. We first posit a basis set $\mathcal{B}$ from the scene points. Each and every model/basis combination then accumulates votes, looking at individual points from the scene and their hash locations in the hash table computed relative to the basis $\mathcal{B}$. The model/basis combinations receiving a lot of votes (or a large weighted vote) is a probable instance of a model with the basis combination from that model matching the chosen basis $\mathcal{B}$. We see from Eqn. (5) that the contribution that a particular hash value based on a point $\mathbf{p}_\xi$ in the scene lends to a model/basis combination (M_k, i, j) should be equal to a log-probability ratio

$$\log\left(\frac{\Pr\left(\mathbf{p}_\xi \mid (M_k, i, j), \mathcal{B}\right)}{\Pr\left(\mathbf{p}_\xi \mid \mathcal{B}\right)}\right).$$

The ratio compares the probability of obtaining a hash to the location where $\mathbf{p}_\xi$ hashes using the basis $\mathcal{B}$, under the assumption that model (M_k, i, j) is present with the basis (i, j) matching the chosen basis $\mathcal{B}$, to the probability without this assumption. The log-probability ratio essentially measures the logarithm of the factor by which the probability increases or decreases due to the conditioning hypothesis. It will be based on the unnormalized density functions of hashes of scene points in the hash space. By unnormalized, we mean that the integral of the expected density functions will give the total number of points, and not unity. In our case, clearly the probability increases by a large factor in the regions near hash locations of the points of M_k computed using the basis (i, j).

We also note from the first term of Eqn. (5) that initially each model/basis combination should have a bias amount equal to the logarithm of its a priori probability. If every model and basis combination is equally likely, then this term may be dropped.

5 Density of Entries in Hash Space

The previous section shows that geometric hashing, with weighted voting, can be interpreted as a Bayesian maximum likelihood object recognition system. However, the weights that should be used depend on the unnormalized density functions of hash values in the hash space, under assumptions of a particular model/basis combination and a particular basis set selection in the scene. Accordingly, in this section, we compute those density functions for similarity transform hashing, using the hash function computation indicated in Section 2.

Our concern here is to compute the log-probability ratio

$$\log\left(\frac{\Pr(\mathbf{p}_\xi|(M_k,i,j),\mathcal{B})}{\Pr(\mathbf{p}_\xi|\mathcal{B})}\right).$$

We denote by (u,v) the location in the two-dimensional hash space to which $\mathbf{p}_\xi$ hashes; we use the similarity transform invariant hash function of the point $\mathbf{p}_\xi$ under the basis $\mathcal{B}$. We may assume that the two points of $\mathcal{B}$ correspond to points i and j of model M_k, possibly corrupted by noise. The probability ratio will depend on the density functions, which will be given below. However, it is important to note that the probabilities depend on the existence of *some* point $\mathbf{p}_\xi$, which happens to hash to a location (u,v), and not the probability that a given, specific point hashes to a particular location. In the latter case, the integral over all space will give a unit value. In the former case, the integral of the values over space will give the expected number of points in the scene.

Let us denote the $n-2$ points in the hash table corresponding to entries due to model M_k with basis (i,j) by $\{(x_q,y_q)\}_{q=1}^{n-2}$. If (u,v) is distant from all of the points (x_q,y_q), then in all likelihood the point $\mathbf{p}_\xi$ has nothing to do with the model M_k, and thus the assumption that model M_k is present has no influence on the probability of a hash to location (u,v). Thus the probability ratio is close to one, and the log-probability ratio is close to zero. Thus we will be able to discard hash points (u,v) that do not lie close to any of the (x_q,y_q). We will define "close" more precisely soon.

If we assume there is no noise and no occlusions, then the assumption that model M_k is present makes it certain that points will hash to each of the (x_q,y_q) locations. In this case, the probability ratio will be infinite at those points, and one elsewhere. However, if we assume that every point of the model embedded in the scene is subject to random perturbation, then the probability of a hash is merely increased in a region about each hash location (x_q,y_q).

Consider three points from model M_k, namely basis (i,j) and a third point ℓ. An entry in the hash table is recorded for this set of data; the entry will be at a location (x,y) and contain the data (M_k,i,j). Suppose that the model is rotated, translated, and scaled into the scene, and that the points are perturbed by Gaussianly-distributed location inaccuracies with variance σ^2. The three points give rise to three corresponding scene points, $\mathbf{p}_\mu=(\mu_1,\mu_2)$, $\mathbf{p}_\nu=(\nu_1,\nu_2)$,

$\mathbf{p}_\xi = (\xi_1, \xi_2)$, respectively. If the first two points are chosen as basis points, and the third point is used as the point for hashing, the resulting hash location (u, v) will satisfy the matrix equation

$$\begin{pmatrix} \nu_1 - \mu_1 & -\nu_2 + \mu_2 \\ \nu_2 - \mu_2 & \nu_1 - \mu_1 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \xi_1 - (\mu_1 + \nu_1)/2 \\ \xi_2 - (\mu_2 + \nu_2)/2 \end{pmatrix}. \tag{6}$$

Although we will not prove it here, the Gaussian perturbations of the three points lead to an approximately Gaussian distribution of the hash location (u, v), centered at (x, y), and with covariance C given by

$$C = \frac{(4(x^2 + y^2) + 3) \cdot \sigma^2}{2 \left\|\mathbf{p}_\mu - \mathbf{p}_\nu\right\|^2} \cdot I \tag{7}$$

where I is the identity matrix (see [54, 55]). (Note that the covariance function is diagonal, and depends on both the location of the hash point (x, y), and also the separation distance between the basis points in the scene.)

Accordingly, if we know that model M_k is in the image, and that the positions (x_q, y_q) are the hash locations due to the $n - 2$ non-basis points of the model, then the expected density function in hash space is simply the superposition of $n - 2$ gaussian distributions, each centered at a distinct point (x_q, y_q) and each with an appropriate covariance:

$$f(u, v) = \sum_{q=1}^{n-2} \frac{1}{2\pi\sqrt{|C_q|}} \exp\left(-\frac{1}{2}(u - x_q, v - y_q) C_q^{-1} (u - x_q, v - y_q)^t\right). \tag{8}$$

Here, C_q is the covariance matrix at (x_q, y_q) given by Eqn. (7). Note that the total mass of the density function is $n - 2$, reflecting the fact that $n - 2$ points are expected based on the knowledge that M_k is present.

Next, suppose that three random scene points are chosen, with two points as the basis pair, and the third point is used for hashing to a location, using again Eqn. (6). The distribution function of the hash location (u, v) will depend on the distribution of points in the scene. Suppose that the scene points are distributed according to a Gaussian distribution with a fixed but unspecified variance. Using the analysis in [52], we can find that the expected distribution of hashes in hash space is independent of the actual variance value, and is given by

$$g(u, v) = \frac{12}{\pi} \frac{1}{(4(u^2 + v^2) + 3)^2}. \tag{9}$$

Since there are roughly S expected scene points, then the expected density function, without any other assumptions, is $S \cdot g(u,v)$.

We are now ready to compute the log-probability ratio. We already know that the density function in hash space, due to hashes from scene points, is $S \cdot g(u,v)$. If we know that the model M_k appears, then this *adds* $n-2$ points, and adds to the expected density function with Gaussian distributions centered at the locations (x_q, y_q), by means of the function $f(u,v)$. Thus the resulting density function is $S \cdot g(u,v) + f(u,v)$. The log of the ratio gives us the formula

$$\log\left(\frac{(\Pr(\mathbf{p}_\xi|(M_k,i,j),\mathcal{B})}{\Pr(\mathbf{p}_\xi|\mathcal{B}))}\right) = \log\left(1 + \frac{f(u,v)}{S \cdot g(u,v)}\right) \tag{10}$$

where point $\mathbf{p}_\xi$ hashes to location (u,v) using basis $\mathcal{B}$, f is given by Eqn. (8), and g is given by Eqn. (9).

6 The Bayesian Geometric Hashing Algorithm

We now summarize the algorithm that allows us to view geometric hashing as a Bayesian maximum-likelihood model-matching system.

First, in the preprocessing phase, every model and every basis pair within that model computes the hash locations of every other point within the model. Each such hash location (x,y) is accompanied by the information of a model M_k, a basis pair (i,j), a model point ℓ other than the basis points, and a predicted normalized covariance radius, which is simply $\tau = (4(x^2+y^2)+3)\cdot\sigma^2$. Records containing this information are organized in such a way that given a location (u,v), all such records having an (x,y) value lying nearby are easily accessed. We find it convenient to include the value τ in the record.

In the recognition phase, feature points are extracted from the scene, and then a pair of these points are chosen as a trial basis, say $\mathbf{p}_\mu$ and $\mathbf{p}_\nu$. For every other point in the scene, the coordinates of the point are computed relative to the basis set, and a hash takes place to a location (u,v) in the hash domain. All nearby records of the form (x,y,M_k,i,j,ℓ,τ) are accessed. For each such nearby record, we record a weighted vote for the model/basis combination $(M_k;i,j)$. From

Eqn. (10), the amount of the vote should be equal to

$$z = \log\left(1 + \frac{\left(4\left(u^2+v^2\right)+3\right)^2 \|\mathbf{p}_\mu - \mathbf{p}_\nu\|^2}{12s\tau} \cdot \exp\left(\frac{\left(-\|(u,v)-(x,y)\|^2\right)}{\tau/\|\mathbf{p}_\mu - \mathbf{p}_\nu\|^2}\right)\right). \tag{11}$$

By nearby, we simply mean that z is greater than some threshold. Note that the formula incorporates the value of σ, the expected error in positioning of the scene points, the number of scene points, s, and the basis-pair separation distance.

As a simple enhancement, we can make sure that no scene point contributes more than one vote for a particular model/basis combination, by the following method. When a point hashes to location (u,v), and an increment z is to be recorded for a particular record (x,y,M_k,i,j,ℓ,τ), we record the value z along with the location (u,v) of the hash point. If another hash (u',v') results in a competing increment z' to the same entry, then we compare the positions of (u,v) and (u',v') with the entry location (x,y), and update the recorded increment corresponding to the hash that lies closer. After all scene points are processed, then the vote for a particular model/basis combination (M_{k_0}, i_0, j_0) is determined from the sum over ℓ of increments z recorded for entries of the form $(x,y,M_{k_0},i_0,j_0,\ell,\tau)$.

When all weighted votes are tallied, model/basis combinations whose total accumulated evidence exceeds some threshold are passed onto a verification process. The algorithm continues by iterating through different candidate basis pairs from the image.

7 A Distributed-Computation Model

It is clear from the description of the previous section that the Bayesian geometric hashing algorithm is inherently parallelizable. The granularity of parallelizability is potentially very fine. Indeed, observe that the *a priori* information that we have stored in the database is in the form of records (x,y,M_k,i,j,ℓ,τ): there is one record for each 3-tuple one can form using distinct features from the set of features for model M_k. There is a total of $Mn\,(n-1)\,(n-2)$ such records.

During recognition, we will need to access all these records that lie nearby locations (u,v) to which hashes are generated. For each nearby record, the expression of Eqn. 11 will have to be evaluated.

One can envision the following fine grain mapping of the computation to processing elements. Each processing element knows of exactly one record $(x, y, M_k, i, j, l, \tau)$; for each of the generated hashes (u, v), the processing element evaluates the expression of Eqn. 11 using the entries of its local record.

Progressively coarser grains of parallelism are also possible. For example, a processing element could maintain locally the $(n-2)$ records having identical values for M_k, i, and j. Or, a processing element can maintain locally the $(n-1)(n-2)$ records having identical values for M_k, and i, etc. At the limit, the processing element maintains locally the records for the entire database (uniprocessor implementation).

It should be clear that the communication requirements between the various processing elements increases as the grain of parallelism becomes finer. Also, one needs to program in progressively lower-level languages as the grain becomes finer.

For models that are represented by 16 features, there is a total of approximately $3K$ such records generated per model. If the number of features extracted in the scenes under consideration is in the order of 100 features, each processing element will have to perform roughly 6 Mflops for each basis-pair selection and for each model. The resulting storage and computational requirements are certainly within the capabilities of modern workstations. It thus appears reasonable to assign the records corresponding toa given database model to a single workstation.

The above design decision introduces the need for means of communication between the various workstations. Let us examine in greater detail the communication requirements. For the purposes of the discussion that follows we assume that one of the workstations has been declared the "master" whereas the remaining workstations operate as "servers."

For a database of M models, we will need a total of $M + 1$ workstations. At the beginning of the computation, each of the servers will be assigned a rank: the i-th server will store locally and maintain the records corresponding to the i-th database model. These records can either be communicated to the i-th server via a message from the master, or directly accessed by the server

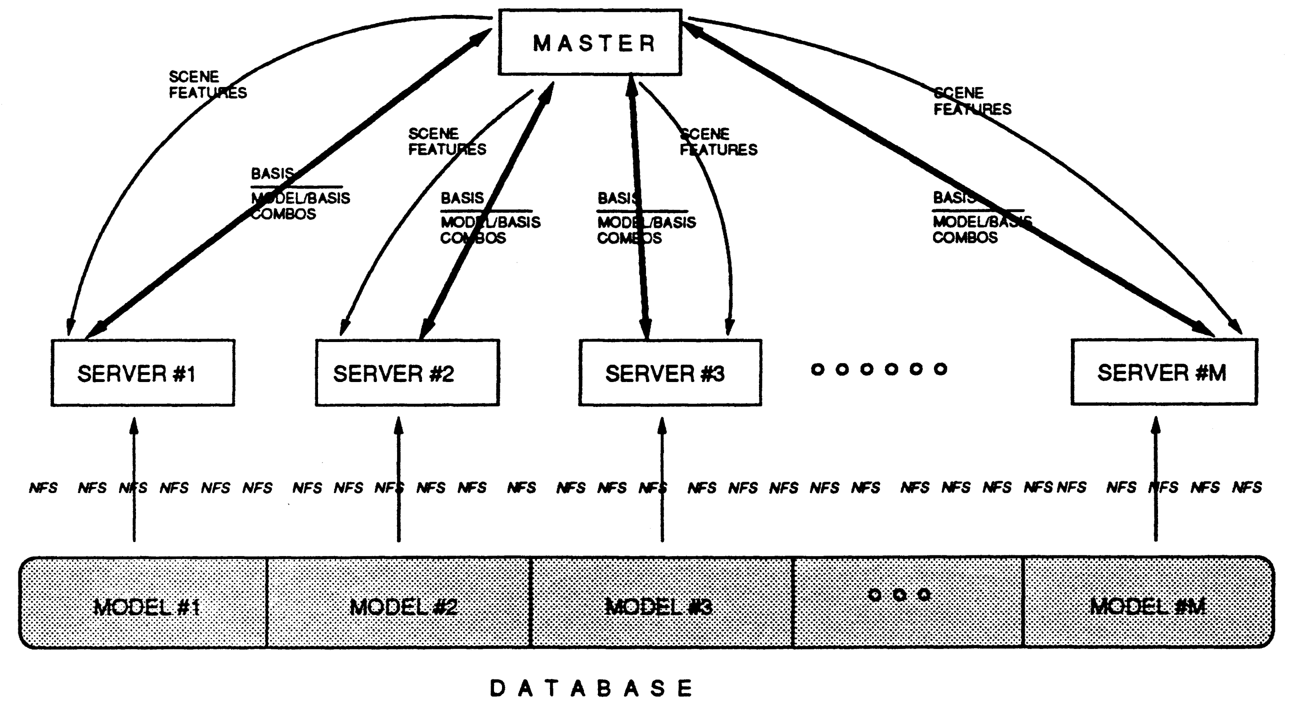

Figure 2. A block-diagram of the master-server model that we use in the implementation of our system. Although the diagram shows a particular grain selection (one database model per server), our paradigm does not impose any restrictions: the size of the grain is decided by the amount of available processing power (i.e. workstations), and the size of the used database.

via NFS.[1] Either way, this transfer of information need only happen once, and before the actual recognition begins.

Once the master is presented with the test scene (i.e. a digitized photograph), the master will need to extract the S features in the scene, and subsequently communicate them to each of the M servers. At this stage, the master can begin the iterative part of the recognition process by selecting pairs of point features from the test scene. For each selected pair, the master communicates the four coordinate values to each of the servers. Using the coordinates of the selected features, each server computes the $S - 2$ many hits (u, v), to hash space that the remaining scene features generate; the server then proceeds to evaluate Eqn. 11 for each hit and each record that is nearby (u, v). When all the nearby records have processed, each server sends back to the master the top T model/basis combinations, in order of decreasing accumulated evidence, which correspond to the given basis selection. The master then histograms the $(M \cdot T)$ model/basis combinations it received from the M servers and decides whether to select a new basis pair and repeat the process.

It is clear from the above description, that during the actual recognition, the communication requirements are minimal. Indeed, the messages that the master and the servers need to exchange are not longer than a few tens of numbers. The messages travel along the existing LAN connections between the master and the servers. In Figure 2, we graphically show the computational model we just described.

Built around the so-called "process model", Concert/C [1, 2] language for programming distributed applications. Concert/C is the C language augmented with a small number of extensions, and supports process creation and termination, asynchronous and synchronous interprocess communication, and inter-process interaction. In the context of our distributed-computation model above, Concert/C allows the user to generate C-like code for both the master and the servers, and also define the *contract* that determines the way in which the master and servers will interact. During run-time, the master begins by creating "copies" of the server code on each one of a set of named workstations. Then the computation proceeds as we described above. The master has the additional responsibility to send "terminate" signals to each of the server copies upon suc-

[1]This alternative is expected to prove more preferable in the case of large databases.

cessful completion (i.e. recognition). [1] contains a more detailed description of the capabilities of Concert/C.

8 System Implementation and Results

In this section, we describe in more detail the automatic feature extraction, the basis selection mechanism, and the actual implementation. We also present some experimental results.

We incorporated an automatic feature extraction mechanism, making use of a boundary following algorithm (eight-connectivity is assumed) applied to the output of the edge detection stage (the Cox-Boie edge detector), followed by a divide-and-conquer polygonal approximation algorithm [21]. The output of the edge detection stage is typically a collections of curves, and polygonal approximations for each of those curves are determined. Curves shorter than 30 pixels are not considered. The vertices of the different approximating polygons comprise our feature set.

More sophisticated approaches, such as spline fitting, could be used to determine the point feature set. The above choice for the feature detection mechanism reflects our desire to determine the limitations of the proposed object recognition system, and proved sufficiently stable to provide robust point locations.

A-4 *Skyhawk*	A-6 *Intruder*	A-10 *Thunderbolt*
F-14 *Tomcat*	F-15 *Eagle*	F-16 *Falcon*
F/A-18 *Hornet*	Mig-21 *Fishbed*	Mig-23 *Flogger*
Mig-29 *Fulcrum*	Mig-31 *Foxhound*	Mirage 2000
Sea Harrier	Panavia Tornado	JA37-Viggen

Table 1. The models of the database.

In order to select among basis sets during the recognition phase, we simply chose randomly among the different bases: the two basis members are selected, with replacement, uniformly from the point feature set. A basis set may be discarded if the basis separation is too small or too big; the cut-off values for our experiments were 150 and 550 pixels respectively. The experiments indicate that for our scenes of approximately 100 points, this simple randomized algorithm requires approximately 35 basis selections before recognition occurs.

The database used for our experiments with the prototype system contained the models of fifteen military aircraft: the profile drawings of fifteen military aircraft from [50] were scanned using a Microtek 300 color/gray-level scanner. These drawings are *not* photographs: they are schematic drawings that we presume are drawn roughly to scale. The scanner is capable of a resolution of 300 dpi, however, we used 120 dpi resolution to digitize the drawings. The fifteen aircraft types contained in our database appear in Table 1.

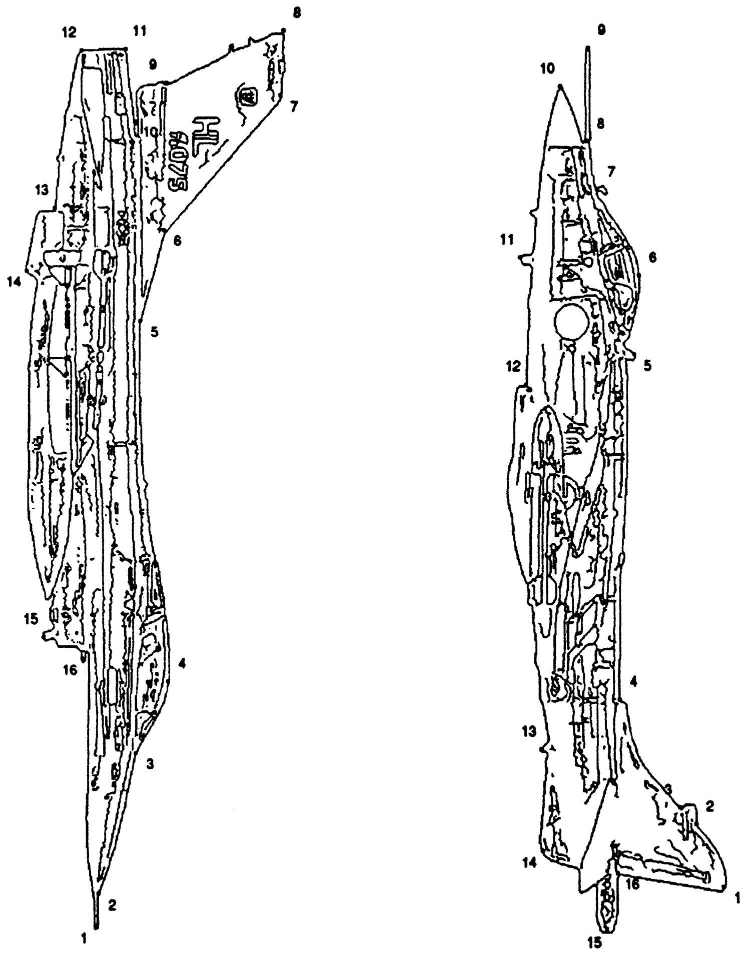

Figure 3. The edge maps and the selected feature points for the database models of the F-16 *Falcon*, and the Sea Harrier.

The digitized images were then processed using the Cox-Boie edge detector; the value of the filter's σ was typically equal to 2.0 [8]. We will call the result of this stage the *edge map*. No other filtering or preprocessing was performed. Example edge maps of the models are shown in Figure 3.

Next, 16 points were selected from each model. The points were chosen to coincide with either points of discontinuity in the tangent direction of the model's contour (i.e., vertices or points of very high curvature), or points of maximum curvature. The model point selection is performed manually during the construction of the database, and thus performed off-line. Candidate feature points may be presented by means of the edge tracking algorithm, but it is not generally required. Figure 3 shows the selected points for two of the database models: the F-16 *Falcon*, and the Sea Harrier.

We then selected a number of photographs of the same military aircrafts, but from a different source [68]. The photographs were chosen on the basis of being taken from approximately the same viewpoint as the drawings in the model database. That is, since the model drawings are side views, and since our recognition algorithms use only similarity invariance, recognition will only be possible with views taken generally from the side. Notably, finding such photographs is not easy, since the pictures must be taken by chase-planes. However, we emphasize that the test images are real photographs, and not drawings nor simulated data. Nor are the models taken from the same source as the photographs. The only thing that the test images and the model database have in common, other than the chosen viewpoints, is the aircraft types.

All the test photographs were digitized using an uncalibrated CCD camera. The result was that distortions and warpings were introduced, not only from the perspective projection of the 3-D plane onto the photograph, but also from the digitization process. However, such distortions might be typical of a working vision system, and all such distortions are approximable by a similarity transformation. Edges were extracted from the resulting gray level images, using the Cox-Boie edge detector that was also used with the models. Again, no preprocessing or other filtering of the test images was performed. A polygonal approximation of the different edge maps provides the points of the feature set. Figures 4 through 7 show the digitized photographs for two of our test inputs together with the corresponding edge maps and extracted point features.

In Figure 4, we can see that the original photograph of the F-16 was taken with the camera positioned below the airplane's midline and towards the back of the aircraft. Further, the airplane is banking to the left.

The original photograph of the Sea Harrier (Figure 6) was very small; a juxtaposed pencil helps estimate the actual size of the original. The original picture was taken with the camera positioned in the front of the aircraft, as evidenced by the visible interior of the left engine intake. The airplane appearing at the bottom of the photograph is a Hunter T-8M.

The experiments were run using a group of 16 workstations (IBM RS/6000). One workstation served as the "master" and there were fifteen "servers", i.e. one per database model. The workstations were connected via a local area network, but they were not all physically located at the same place: several of the servers were at the Yorktown site of the T. J. Watson center, whereas the remaining as well as the master were 12 miles away, at the Hawthorne site of the center. The experiments took place in the middle of a typical working day while the workstations were being used by their regular customers: the server processes shared the CPU with the other processes running on the workstation. As the timing results in each of the experiments indicate, the network delay was *not* noticeable. In terms of performance, and for a given basis-pair selection, the processing required ≈ 15 $\underline{wall-clock}$ milliseconds (*not* CPU milliseconds) per scene point and this figure included both the actual CPU and LAN requirements. This represents an improvement in performance by almost a factor of 2 over a Connection Machine ($CM-2$) implementation of similar size database (see [51]). The small differences between the elapsed time on the master and the slowest of the servers show how much time the master spends in bookkeeping. This performance can be further improved by an appropriate decrease of the grain of parallelism.

A final point can be made based on the diagram of Figure 8. The diagram depicts the time required to process a scene point as a function of the models a server is controlling. Although, one would anticipate a linear increase, the actual diagram indicates that this is not the case. This can be explained by the fact that the server processes are *light-weight*: they do not use but a small percentage of the corresponding CPU power; doubling the models assigned to a

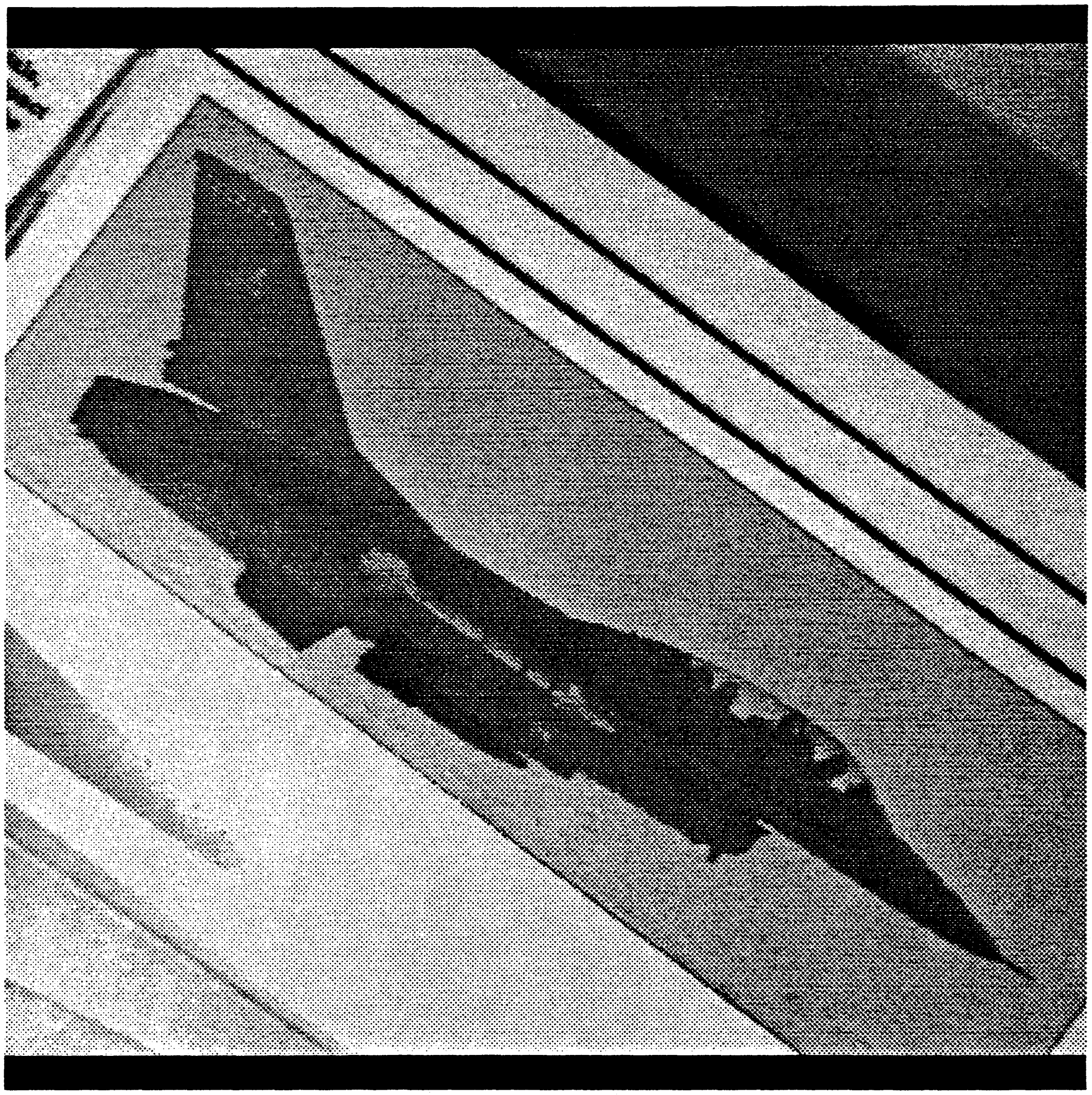

Figure 4. A test image for the recognition algorithm. The photograph of the F-16 is taken from page 183 of [68].

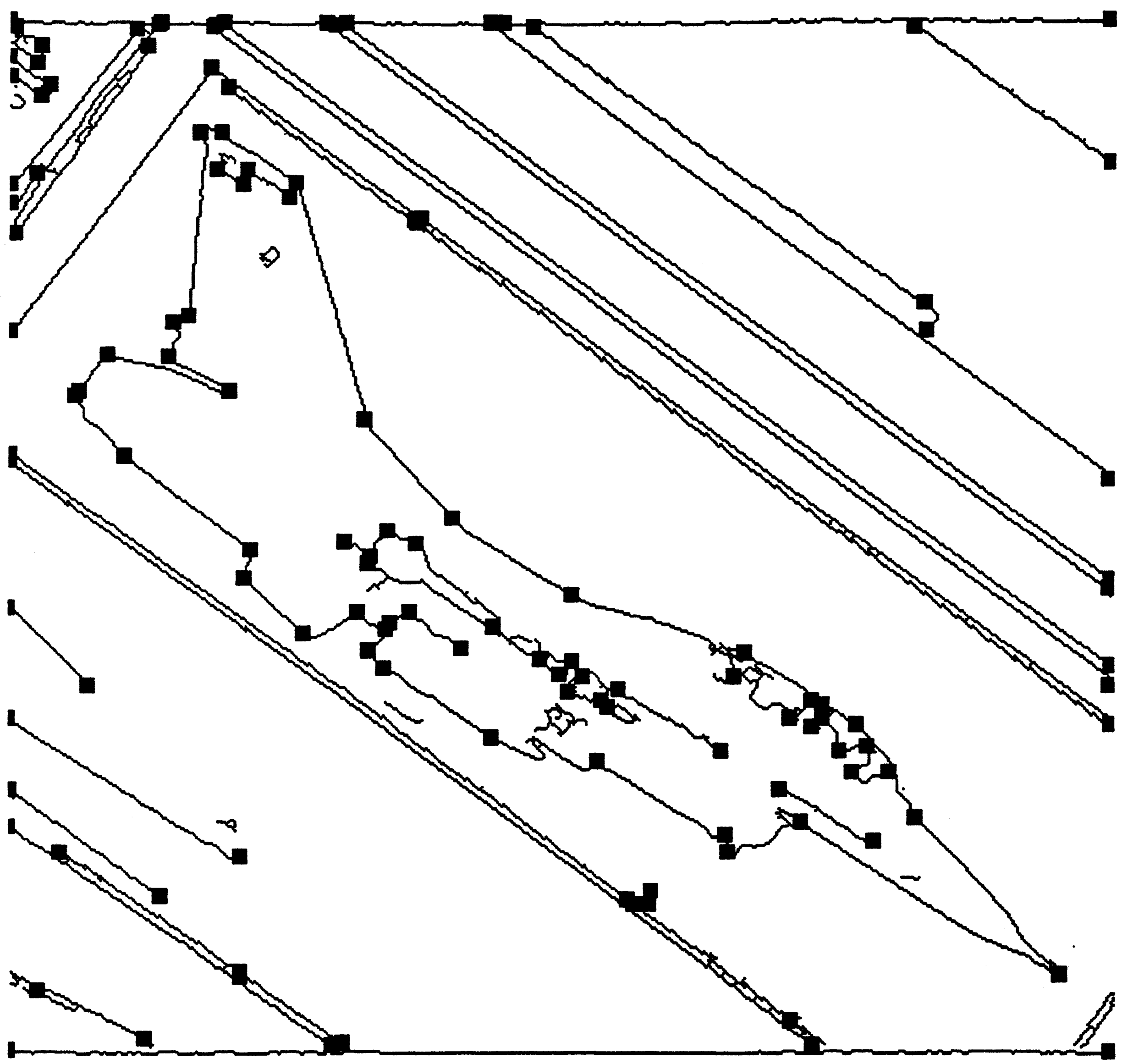

Figure 5. The edge map extracted by the Cox-Boie edge detector (the value of σ was 2.0) for the F-16 test image. Also shown are the 111 automatically extracted features.

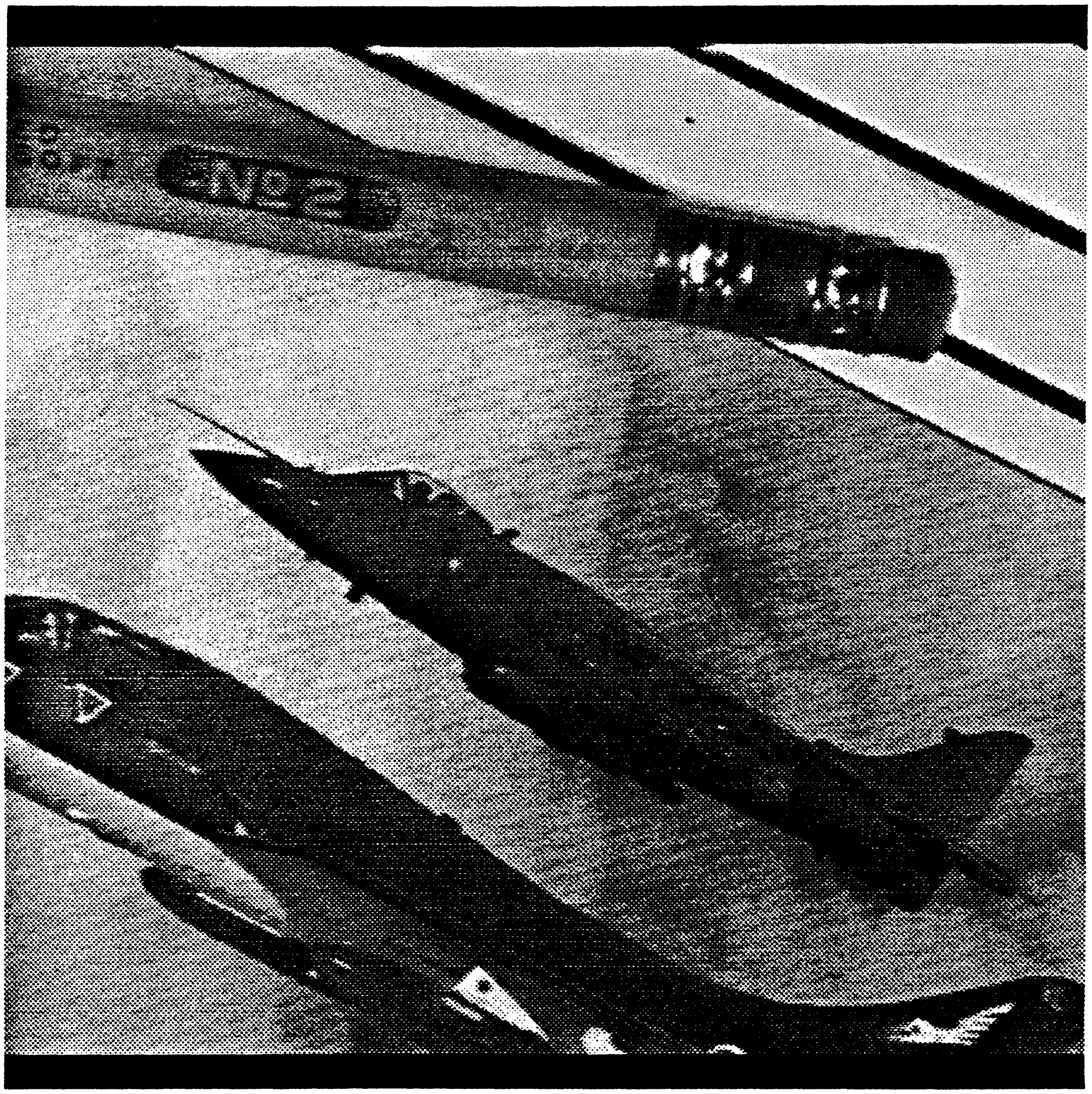

Figure 6. The image of a Sea Harrier. The airplane at the bottom of the picture is a Hunter T-8M. From page 365 of [68].

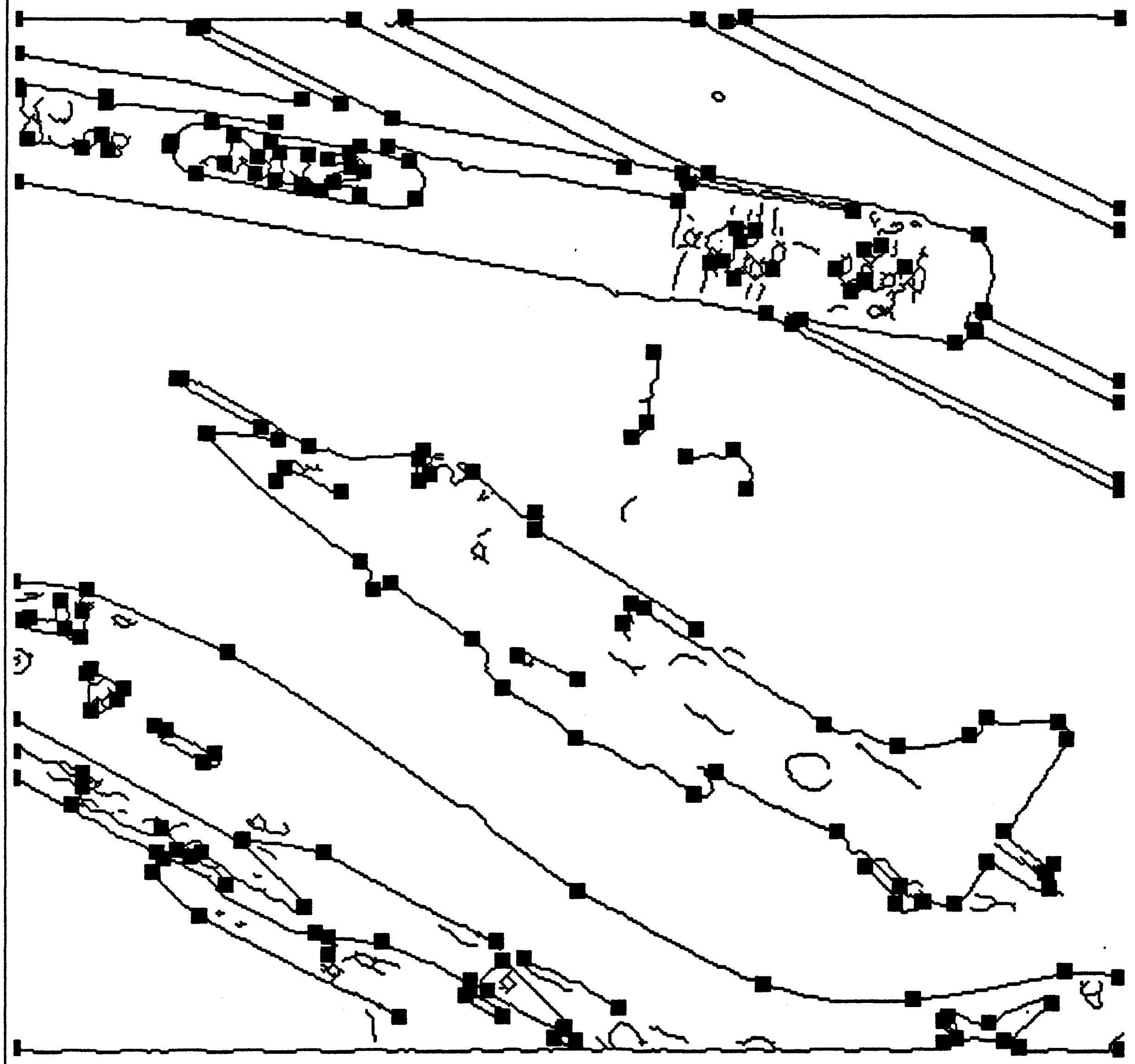

Figure 7. The edge map extracted by the Cox-Boie edge detector (the value of σ was 2.0) for the Sea Harrier test image. Also shown are the 208 automatically extracted features.

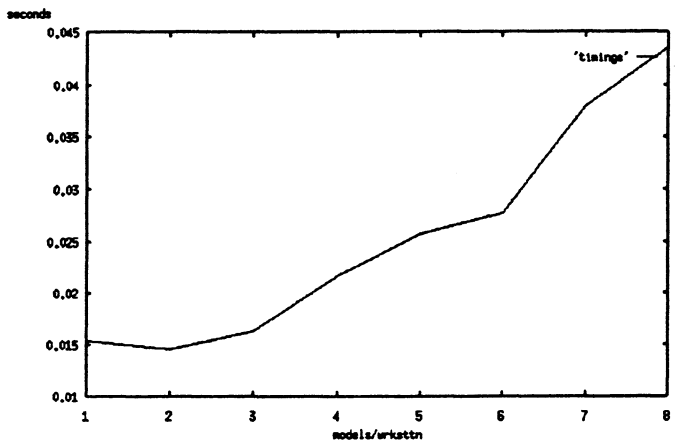

Figure 8. The time required to process a scene point, as a function of the models a signle server (workstation) is controling.

given workstation simply results in the use of twice as much CPU time on that workstation: the performance remains unchanged. This observation has the following ramifications: (a) one can use the distributed recognition algorithm without imposing any observable burden on the server workstations; and (b) as long as the main memory size of the server workstations allows, one can at least double the size of the database without incuring any slowdown.

In Figures 9 and 10 we show the output of our system's implementation for two test inputs. The retrieved database model appropriately scaled, rotated, and translated is shown overlayed on the test input. In the bottom half of each screendump, the 9 top retrieved database models are shown in order of decreasing accumulated evidence (column-major order). For each of the 9 models, its name, and the retrieved basis are also indicated. The point features corresponding to each basis are marked along the contour of the corresponding model. Above each model, bars providing a length-encoded representation of the evidence accumulated for each model/basis combination are also shown. It should be noted here that the recovery of the transformation was based solely on the basis pair, and *not* on a best least-squares match between all the corresponding model and scene feature pairs.

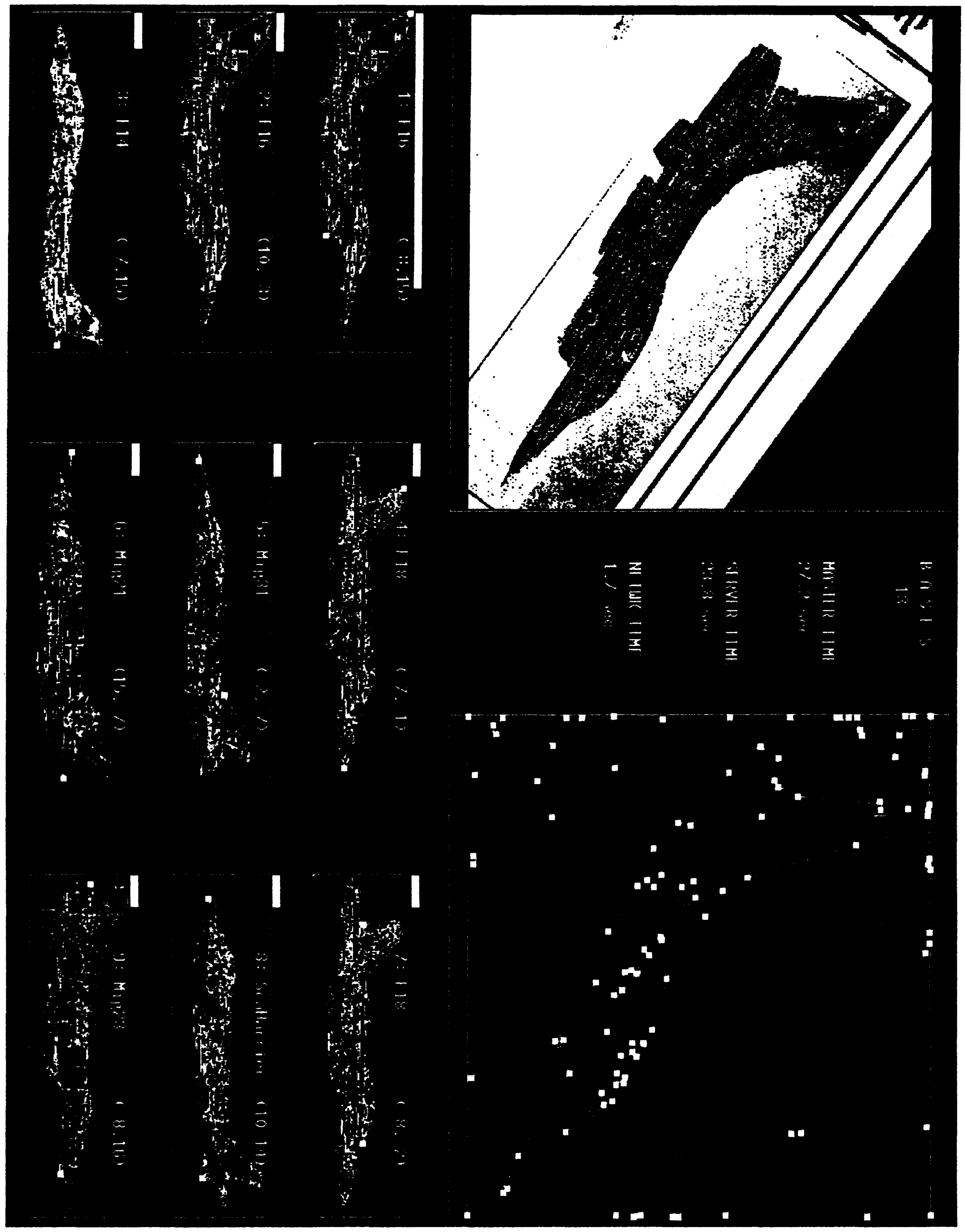

Figure 9. The output of the implementation of our system using 15 servers. The test input (F-16) is shown on the top left. The edge map together with the automatically extracted point features is shown on the top right; the basis selection that led to recognition is also marked. A total of 13 basis selections was required, and the elapsed time was 27.2 seconds (NB. this figure does not include the edge detection and feature extraction stages). The bars above each of the 9 top retrieved models provide a length encoding of the total accumulated evidence for the corresponding model/basis combination. The retrieved database model appropriately scaled, rotated and translated is shown overlayed on the test input.

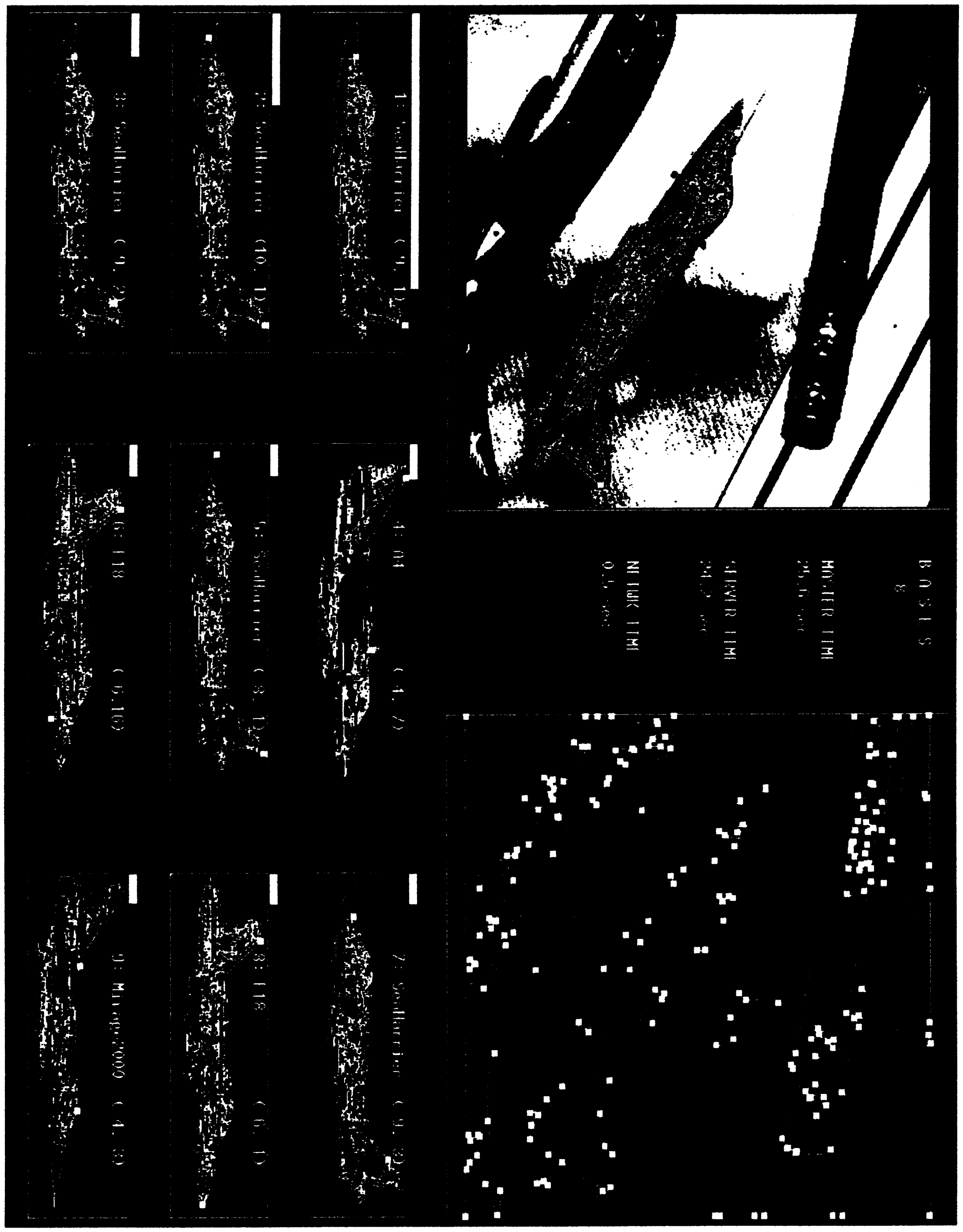

Figure 10. The output of the implementation of our system using 15 servers. The test input (Sea Harrier) is shown on the top left. The edge map together with the automatically extracted point features is shown on the top right; the basis selection that led to recognition is also marked. A total of 8 basis selections was required, and the elapsed time was 25.6 seconds (NB. this figure does not include the edge detection and feature extraction). The bars above each of the 9 top retrieved models provide a length encoding of the total accumulated evidence for the corresponding model/basis combination. The retrieved database model appropriately scaled, rotated and translated is shown overlayed on the test input.

In all our experiments, the true model/basis combination was always discovered as the pair with the largest weighted vote, i.e., evidence. However, even if the correct model were not found as the maximum winner, it is assumed that a postprocessing stage would be used to verify a number of possible matches. For example, with our database of 15 models, there are 3600 possible model/basis combinations. If the first 9 or so matching model/basis combinations from the hashing algorithm are checked, we have still achieved a considerable speedup over the alternative of checking all possible matchings. Accordingly, the fact that the accumulated evidence for the ninth model/basis combination is considerably less than the evidence for the winning model/basis combination indicates that the method is robust.

9 Conclusion

We have presented a new paradigm for performing object recognition. We believe that this paradigm makes the concept of realistic object recognition more tangible.

We described a new framework that explicitly takes into account the fact that the input data have been distorted by noise, and which uses a Bayesian approach to combine evidence that has been accumulated from the various scene points.

We also suggested the use of an alternative computational model, one that takes the distributed approach to computing and makes use of a farm of readily available, low-cost workstations, instead of more expensive, less accessible, specialized hardware.

We demonstrated the validity of the resulting paradigm by describing a prototype system that was implemented on fifteen workstations, and which can recognize aircraft models that have undergone similarity transformations, from real-world photographs. The system scales readily, can use large databases, handles noisy input data, and exhibits performance that is superior by a factor of 2 over that obtained for a similar system using a Connection Machine ($CM-2$).

We beleieve that appropriate selection of the grain of parallelism will allow the straightforward realization of systems that will achieve recognition results in sub-second times, and which need only use clusters of public workstations.

References

[1] **Auerbach, J., D. Bacon, A. Goldberg, G. Goldszmidt, A. Gopal, M. Kennedy, A. Lowry, J. Russell, W. Silverman, R. Strom, D. Yellin, and S. Yemini.** High-level language support for programming reliable distributed systems. In *Proceedings of the International Conference on Computer Languages*, Oakland, California, April 1992.

[2] **Auerbach, J., M. Kennedy, J. Russell, and S. Yemini.** Interprocess Communication in Concert/C. Technical Report RC17341, IBM T.J. Watson Research Center, October 1991.

[3] **Ayache, N. and O. Faugeras.** HYPER: A New Approach for the Recognition and Positioning of Two-dimensional Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 8(1):44–54, 1986.

[4] **Ballard, D.** Generalizing the Hough Transform to Detect Arbitrary Shapes. *Pattern Recognition*, 13(2):111–122, 1981.

[5] **Ballard, D. and D. Sabbah.** Viewer Independent Object Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(6):653–660, 1983.

[6] **Bergevin, R. and M. Levine.** Generic Object Recognition: Building Coarse $3D$ Descriptions from Line Drawings. In *Proceedings of the IEEE Workshop on Interpretation of* $3D$ *Scenes*, pages 68–74, Austin, Texas, November 1989.

[7] **Biederman, I.** Human Image Understanding: Recent Research and a Theory. *Computer Vision, Graphics, and Image Processing*, 32:29–73, 1985.

[8] **Boie, R. and I. Cox.** Two Dimensional Optimum Edge Recognition using Matched and Weiner Filters for Machine Vision. In *Proceedings of the International Conference on Computer Vision*, London, England, December 1987.

[9] **Bolle, R., A. Califano, and R. Kjeldsen**. A Complete and Extendable Approach to Visual Recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14:534–548, May 1992.

[10] **Bolles, R. and P. Horaud**. 3DPO: A Three-dimensional Part Orientation System. *The International Journal of Robotics Research*, 5:3–26, 1986.

[11] **Bolles, R. and R. Cain**. Recognizing and Locating Partially Visible Objects: the Local Feature Focus Method. *The International Journal of Robotics Research*, 1:57–82, 1982.

[12] **Bourdon, O. and G. Medioni**. Object Recognition Using Geometric Hashing on the Connection Machine. In *Proceedings of the International Conference on Patern Recognition*, pages 596–600, Atlantic City, New Jersey, June 1990.

[13] **Brooks, R**. Symbolic Reasoning Around $3D$ Models and $2D$ Images. *Artificial Intelligence*, 17:285–348, 1981.

[14] **Califano, A**. Feature Recognition Using Correlated Information Contained in Multiple Neighborhoods. In *Proceedings of the Seventh AAAI*, pages 831–836, 1987.

[15] **Califano, A. and R. Mohan**. Multidimensional Indexing for Recognizing Visual Shapes. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, Maui, Hawaii, June 1991.

[16] **Carrihill, B. and R. Hummel**. Experiments with the Intensity Ratio Depth Sensors. *Computer Vision, Graphics, and Image Processing*, 32:337–358, 1985.

[17] **Charniak, E. and D. McDermott**. *Introduction to Artificial Intelligence*. Addison-Wesley, 1985.

[18] **Chen, C. and A. Kak**. A Robot Vision System for Recognizing $3D$ Objects in Low-order Polynomial Time. *IEEE Transactions on Systems Man and Cybernetics*, 19(6):1535–1563, 1989.

[19] **Costa, M., R. Haralick and L. Shapiro.** Optimal Affine Matching. In *Proceedings of the 6th Israeli Conference on Artificial Intelligence and Computer Vision*, Tel Aviv, Israel, December 1989.

[20] **Dickinson, S., A. Pentland, and A. Rosenfeld.** $3D$ Shape Recovery Using Distributed Aspect Matching. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 14(2):174–198, 1992.

[21] **Duda, R. and P. Hart.** *Pattern Classification and Scene Analysis.* John Wiley and Sons, 1973.

[22] **Fischler, M. and R. Bolles.** Random Sample Consensus: A Paradigm to Model-fitting with Application to Image Analysis and Automated Cartography. *Communications of the ACM*, 24(6):381–395, 1981.

[23] **Flynn, P. and A. Jain.** BONSAI: $3D$ Object Recognition Using Constrained Search. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1066–1075, 1991.

[24] **Flynn, P. and A. Jain.** $3D$ Object Recognition Using Invariant Feature Indexing of Interpretation Tables. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 55(2):119–129, 1992.

[25] **Forsyth, D., J. Mundy, A. Zisserman, C.Coelho, A. Heller, and C. Rothwell.** Invariant Descriptors for $3D$ Object Recognition and Pose. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):971–991, 1991.

[26] **Gavrila, D. and F. Groen.** $3D$ Object Recognition from $2D$ Images Using Geometric Hashing. *Pattern Recognition Letters*, 13(4):263–278, 1992.

[27] **Goad, C.** Special Purpose Automatic Programming for $3D$ Model-based Vision. In *Proceedings of the DARPA Image Understanding Workshop*, 1983.

[28] **Grimson, W. and D. Huttenlocher**. On the Sensitivity of Geometric Hashing. In *Proceedings of the International Conference on Computer Vision*, Osaka, December 1990.

[29] **Grimson, W. and T. Lozano-Perez**. Model-based Recognition and Localization from Sparse Range or Tactile Data. *The International Journal of Robotics Research*, 3(3):3–35, 1984.

[30] **Grimson, W. and T. Lozano-Perez**. Localizing Overlapping Parts by Searching the Interpretation Tree. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 9(4):469–482, 1987.

[31] **Guéziec, A. and N. Ayache**. Smoothing and Matching of 3*D* Space Curves. In *Proceedings of the European Conference on Computer Vision*, Santa Margherita Ligure, Italy, May 1992.

[32] **Hong, J. and H. Wolfson**. An Improved Model-based Matching Method Using Footprints. In *Proceedings of the International Conference on Patern Recognition*, Rome, Italy, November 1988.

[33] **Hough, P.** Method and Means for Recognizing Complex Patterns, 1962. U.S. Patent 3,069,654.

[34] **Hummel, R. and H. Wolfson**. Affine Invariant Matching. In *Proceedings of the DARPA Image Understanding Workshop*, April 1988.

[35] **Huttenlocher, D. and S. Ullman**. Recognizing Solid Objects by Alignment. In *Proceedings of the DARPA Image Understanding Workshop*, April 1988.

[36] **Kalvin, A., E. Schonberg, J. Schwartz, and M. Sharir**. Two-dimensional Model-based Boundary Matching Using Footprints. *The International Journal of Robotics Research*, 5(4):38–55, 1986.

[37] **Kim, W.-Y. and A. Kak**. 3*D* Object Recognition Using Bipartite Matching Embedded in Discrete Relaxation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(3):224–251, 1991.

[38] **Kishon, E. and H. Wolfson.** 3D Curve Matching. In *Proceedings of the AAAI Workshop on Spatial Reasoning and, Multisensor Fusion*, pages 250–261, St. Charles, Illinois, October 1992.

[39] **Kriegman, D. and J. Ponce.** Recognizing and Positioning Curved 3D Objects. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(12):1127–1137, 1990.

[40] **Lamdan, Y.** *Geometric Hashing.* PhD thesis, New York University, June 1989.

[41] **Lamdan, Y. and H. Wolfson.** Geometric Hashing: A General and Efficient Model-based Recognition Scheme. In *Proceedings of the International Conference on Computer Vision*, pages 238–249, 1988.

[42] **Lamdan, Y. and H. Wolfson.** Geometric Hashing: A General and Efficient Model-based Recognition Scheme. In *Proceedings of the International Conference on Computer Vision*, pages 238–249, 1988.

[43] **Lamdan, Y. and H. Wolfson.** On the Error Analysis of Geometric Hashing. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, Maui, Hawaii, June 1991.

[44] **Lamdan, Y., J. Schwartz and H. Wolfson.** Object Recognition by Affine Invariant Matching. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, pages 335–344, Ann Arbor, Michigan, June 1988.

[45] **Lamdan Y., J. Schwartz, and H. Wolfson.** On Recognition of 3D Objects from 2D Images. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 1407–1413, Philadelphia, PA, April 1988.

[46] **Linnainmaa, S, D. Harwood, and L. Davis.** Pose Determination of a Three-dimensional Object Using Triangle Pairs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 10:634–647, 1988.

[47] **Lowe, D. G.** *Perceptual Organization and Visual Recognition.* Kluwer Academic Publishers, 1985.

[48] **Mundy, J. and D. Thompson.** Model-directed Object Recognition on the Connection Machine. In *Proceedings of the DARPA Image Understanding Workshop*, pages 98–104, 1987.

[49] **Mundy, J. and D. Thompson.** Three-dimensional Model Matching from an Unconstrained Viewpoint. In *Proceedings of the IEEE International Conference on Robotics and Automation*, pages 208–220, Raleigh, NC 1987.

[50] **Richardson, D.** *The World's Major Military Jets.* Salamander Books Ltd., 1990.

[51] **Rigoutsos, I.** *Massively Parallel Bayesian Object Recognition.* PhD thesis, New York University, August 1992.

[52] **Rigoutsos, I. and R. Hummel.** Implementation of Geometric Hashing on the Connection Machine. In *Proceedings of the IEEE Workshop on Directions in Automated CAD-Based Vision*, Maui, Hawaii, June 1991.

[53] **Rigoutsos, I. and R. Hummel.** On a Scalable Parallel Implementation of Geometric Hashing on the Connection Machine. Technical Report 554, Courant Institute of Mathematical Sciences, New York University, April 1991.

[54] **Rigoutsos, I. and R. Hummel.** Robust Similarity Invariant Matching in the Presence of Noise. In *Proceedings of the 8th Israeli Conference on Artificial Intelligence and Computer Vision*, Tel Aviv, Israel, December 1991.

[55] **Rigoutsos, I. and R. Hummel.** Several Results on Affine Invariant Geometric Hashing. In *Proceedings of the 8th Israeli Conference on Artificial Intelligence and Computer Vision*, Tel Aviv, Israel, December 1991.

[56] **Rigoutsos, I. and R. Hummel.** Massively Parallel Model Matching: Geometric Hashing on the Connection Machine. *IEEE Computer: Special Issue on Parallel Processing for Computer Vision and Image Understanding*, February 1992.

[57] **Roberts, L.** *Optical and Electro-optical Information Processing*, chapter title: Machine Perception of Three-dimensional Solids. MIT Press, Cambridge, Massachusetts, 1965.

[58] **Schreiber, I. and M. Ben-Bassat.** Polygonal Object Recognition. In *Proceedings of the International Conference on Patern Recognition*, pages 852–859, Atlantic City, New Jersey, June 1990.

[59] **Schwartz, J. and H. Wolfson.** Improved Shape-signature and Matching Methods for Model-based Robotic Vision. In *Proceedings of the Workshop on Space Telerobotics*, pages 103–109, JPL, NASA, January 1987.

[60] **Shankar, R., G. Ramamoorthy and M. Suk.** Three-dimensional Object Recognition on the Connection Machine. *Pattern Recognition Letters*, 11(6):485–492, 1990.

[61] **Sossa, H. and R. Horaud.** Model Indexing: the Graph-hashing Approach. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, Urbana-Champaign, Illinois, June 1992.

[62] **Stark, L. and K. Bowyer.** Achieving Generalized Object Recognition through Reasoning about Association of Function to Structure. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1097–1104, 1991.

[63] **Stein, F. and G. Medioni.** Efficient Two-dimensional Object Recognition. In *Proceedings of the International Conference on Patern Recognition*, pages 596–600, Atlantic City, NewJersey, June 1990.

[64] **Stein, F. and G. Medioni.** Structural Hashing: Efficient Three-dimensional Object Recognition. In *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference*, pages 244–250, Maui, Hawaii, June 1991.

[65] **Stockman, G.** Object Recognition and Localization via Pose Clustering. *Computer Vision, Graphics, and Image Processing*, 40:361–387, 1987.

[66] **Strat, T. and M. Fischler.** Context-based Vision: Recognizing Objects Using Information from both $2D$ and $3D$ Imagery. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):1050–1065, 1991.

[67] **Swain, M.** *Color Indexing.* PhD thesis, University of Rochester, 1990.

[68] **Sweetman W. and R. Bonds.** *The Great Book of Modern Warplanes.* Portland House, 1987.

[69] **Ullman, S.** Aligning Pictorial Descriptions: An Approach to Object Recognition. *Cognition*, 32(3):193–254, 1989.

[70] **Ullman, S. and R. Basri.** Recognition by Linear Combination of Models. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(10):992–1006, 1991.

[71] **Vayda, A. and A. Kak.** A Robot Vision System for Recognition of Generic Shaped Objects. *Computer Vision, Graphics, and Image Processing: Image Understanding*, 54(1):1–46, 1991.

[72] **Wolfson, H., E. Schonberg, A. Kalvin and Y. Lamdan.** Solving Jigsaw Puzzles by Computer Vision. *Annals of Operations Research*, 12:51–64, 1988.