COMPUTER VISION AND IMAGE UNDERSTANDING
Vol. 62, No. 1, July, pp. 11–26, 1995

A Bayesian Approach to Model Matching with Geometric Hashing*

ISIDORE RIGOUTSOS

IBM Thomas J. Watson Research Center, P.O. Box 704, Yorktown Heights, New York 10598

AND

ROBERT HUMMEL

Courant Institute, Department of Computer Science, New York University, 251 Mercer Street, New York, New York 10012

Received May 15, 1992; accepted May 31, 1994

Geometric hashing methods provide an efficient approach to indexing from image features into a database of models. The hash functions that have typically been used involve quantization of the values, which can result in nongraceful degradation of the performance of the system in the presence of noise. Intuitively, it is desirable to replace the quantization of hash values and the resulting binning of hash entries by a method that gives increasingly less weight to a hash table entry as a hashed feature becomes more distant from the hash entry position. In this paper, we show how these intuitive notions can be translated into a well-founded Bayesian approach to object recognition and give precise formulas for the optimal weight functions that should be used in hash space. These extensions allow the geometric hashing method to be viewed as a Bayesian maximum-likelihood framework. We demonstrate the validity of the approach by performing similarity-invariant object recognition using models obtained from drawings of military aircraft and automobiles and test images from real-world grayscale images of the same aircraft and automobile types. Our experimental results represent a complete object recognition system, since the feature extraction process is automated. Our system is scalable and works rapidly and very efficiently on an 8*K*-processor *CM* − 2, and the quality of results using similarity-invariant model matching is excellent. © 1995 Academic Press, Inc.

1. INTRODUCTION

Geometric hashing is a method for organizing the search for matches in a model-based vision system. Using the title geometric hashing, the technique was first prominently introduced by Lamdan and Wolfson [11] and has since been implemented and extended by many researchers [10, 11, 13, 16–19]. The method provides only the search engine portion of an object recognition system; the representation and extraction of the features form crucial inputs to geometric hashing as with any object recognition system. Geometric hashing, however, provides a particularly efficient search strategy, by multiply encoding the models with respect to many different normalizations. The scene is then compared not only against each of the models, but also against many possible normalizations of the models. By using a hashing strategy for indexing from the scene features to the features in the database of models, it is possible to achieve parallelizable, robust, efficient object recognition.

A fundamental notion in geometric hashing is that of a basis set. A basis set is a collection of features sufficient to establish a transformation when placed in correspondence with another basis set. Thus for translation invariance, a $2D$ point forms a basis; for rotation, translation, and scale (similarity) invariance of $2D$ point sets, two points form a basis set; affine transformation requires three points to form a basis set. In this paper, we demonstrate similarity transformation invariant matching of sets of $2D$ points to databases of models consisting of $2D$ points. However, our treatment is more general, and extensions are possible. In particular, our treatment deals with patterns of $2D$ points, but a more general treatment would deal with patterns of n-dimensional features.

As the degree of invariance increases, the discriminability of an object recognition system will decrease. For example, affine invariance allows greater flexibility, and thus greater numbers of "false alarm" matches to models, especially when the number of models is large. In geometric hashing, each model is "normalized" with respect to every reasonable basis set chosen from the model, creating a database of multiply encoded instances of the model. By prenormalizing the models with respect to a discrete collection of possible transformations, the matching process is able to check all transformations for each model simultaneously. That is, a basis set is chosen in the scene, and the scene is normalized with respect to that basis set and then compared with respect to every

* This work has been supported by an IBM Graduate Research Fellowship to I. Rigoutsos; partial support to R. Hummel was provided by AFAL Contract F33615-89-C-1087 Reference 87-02-PMRE.

normalized model/basis combination. A hashing approach is used to reduce the complexity of the mapping from normalized features in the scene to normalized features in the model database. A summary of the geometric hashing method is given in the next section.

In order to deal with noise, a normalized point (or feature) from the scene must be viewed as an approximate location and thus must be allowed to match a range of locations in the normalized space of features in model/basis patterns. Because geometric hashing uses a hashing scheme, quantized bins have been used in all geometric hashing systems heretofore. Thus normalized scene points quantize their locations in order to obtain a bin location that contains entries for points in model/basis patterns. The size of the bin is a critical parameter. If the bins are too large, then there will be too many false alarms. If the bins are too small, then there is insufficient provision for noise in the input. Gavrila and Groen [7] use a compromise: they use small bins, but compose a circular pattern of multiple bins for any given hash.

Analyses of the discriminability of object recognition based upon matching using fixed-size bins in a normalization space provides pessimistic predictions, particularly for affine-invariant recognition [8]. Problems are less severe with similarity invariance, but fixed-size bins provide a crude means of dealing with possible error in feature values.

Costa *et al.* provide a more reasonable alternative [4, 5]. They observe that for Gaussian error in the position of $2D$ points, the locations of points normalized with respect to a scene basis will, to first order, distribute according to a Gaussian distribution. By adding a noise model to geometric hashing, they provide a more rational approach to matching of scene features to possible model features. That is, each normalized model can be credited with a weighted vote when normalized scene points lie nearby. Further, by making an independence assumption over the scene features, they develop a Bayesian interpretation for the computations that can be performed by weighted voting.

Our analysis in this paper is similar. We likewise study the possibility of weighted votes, and we similarly perform a Bayesian analysis. However, our assumptions are different, and thus the resulting formulas are different. The principal difference arises because we allow the error range to vary depending on the scene basis set, and also upon expected errors for particular normalized model patterns. Notably, we make use of log-probability ratios in order to avoid certain approximations and term coalescing.

The use of the Bayesian voting scheme given in Section 4 provides dramatically better discrimination ability to a geometric hashing system. Using these methods, we demonstrate a system that is able to recognize models from a database of 32 models, each with 16 feature-points, in scenes containing typically a couple hundred features. The system is able to deal with positional error in the scene that is several percent (2–3%) of the size of the objects in the scene. Performance of the system without Bayesian weighted voting depends on the size of the bins and other parameters, but essentially fails at all tasks that are demonstrated in Section 7. A related analysis of the discriminability of binning using simulation data is presented in [15], where a particular bin size is chosen and a hypothetical database of 500 models is generated. An error analysis based on binning is also given in [12]. These studies, together with those in [8], suggest that binning in conjunction with scene noises poses severe problems. We believe that the use of Bayesian weighted voting largely addresses these problems, especially for low-order transformation invariance, such as similarity invariance.

2. AN OVERVIEW OF GEOMETRIC HASHING

We begin with a brief overview of the geometric hashing approach to object recognition. In the feature extraction stage, features such as points, linear and curvilinear segments, and corners are extracted. Any such collection of features can be represented by a set of dots: each dot represents the feature's location, and associated with each dot is a list of one or more attributes (the feature's *attribute list*) that depends on the corresponding feature's type.

We can thus confine ourselves to the problem of recognition of patterns of *point* features (dot patterns), with an attribute list associated with each point.

Suppose that we perform recognition of patterns of point features with empty attribute lists, where the pattern may be translated, rotated, and scaled (similarity transformations). Two points are needed to define a basis. Figure 1 shows a model (M_1) consisting of five dots with position vectors $\mathbf{q}_1$, $\mathbf{q}_2$, $\mathbf{q}_3$, $\mathbf{q}_4$, and $\mathbf{q}_5$ respectively. We begin by scaling the model M_1 so that the magnitude of $\overrightarrow{\mathbf{q}_4\mathbf{q}_1}$ in the Oxy system is equal to 1 (Fig. 2). Suppose now that we place the midpoint between dots 4 and 1 at the origin of a coordinate system Oxy in such a way that the vector $\overrightarrow{\mathbf{q}_4\mathbf{q}_1}$ has the direction of the positive x axis. The remaining three points of M_1 will land in three locations. Let us record in a hash table, in each of the three bins where the remaining points land, the fact that model M_1 with basis (4, 1) yields an entry in this bin. This is shown graphically in Fig. 2. The bins may be infinitesimal and are needed only to facilitate fast access to regions of the "hash space."

Similarly, the hash table contains three entries of the form (M_1, (4, 2)), three entries of the form (M_1, (4, 3)), etc. Each triplet of entries is generated by first scaling the

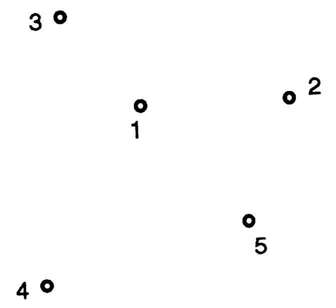


FIG. 1. Model M_1 consisting of five points.

model M_1 so that the corresponding basis has unit length in the Oxy coordinate system, and then by placing the midpoint of the basis at the origin of the hash table in such a way that the basis vector has the direction of the positive x axis. The process is repeated for each ordered basis, and each of the models in the database. As a result, the final hash table data structure will contain a list of entries of the form (*model*, *basis*) in each hash table bin.

In the recognition phase, a pair of points, $(\mathbf{p}_{\mu_1}, \mathbf{p}_{\mu_2})$, from the image is chosen as a candidate basis. This ordered basis defines a coordinate system Oxy whose center coincides with the midpoint of the pair; the direction of the basis vector $\mathbf{p}_{\mu_2} - \mathbf{p}_{\mu_1}$ coincides with that of the positive x axis. The magnitude of the basis vector defines the "unit" length for Oxy.

The coordinates of all other points are then calculated in the coordinate system defined by the basis. Each of the remaining image points is mapped to the hash table, and all entries in the corresponding hash table bin and nearby bins receive a vote. Figure 3 shows this graphically. Here, we merely make use of the fact that any point $\mathbf{p}$ in the plane can be represented in the selected basis, namely, there is a unique pair of scalars (u, v) such that

$$\mathbf{p} - \mathbf{p}_0 = u(\mathbf{p}_{\mu_2} - \mathbf{p}_{\mu_1}) + v(\mathbf{p}_{\mu_2} - \mathbf{p}_{\mu_1})^{\perp},$$

where $\mathbf{p}_0 = (\mathbf{p}_{\mu_1} + \mathbf{p}_{\mu_2})/2$ is the midpoint between $\mathbf{p}_{\mu_1}$ and $\mathbf{p}_{\mu_2}$.

If there are sufficient votes for one or more (*model, basis*) combinations, then a subsequent stage attempts to verify the presence of a model. In the case where model points are missing from the image because they are obscured, recognition is still possible, as long as there is a sufficient number of points hashing to the correct hash table bins. Recognition should occur once a basis is chosen in the scene such that all the points belong to a single model. It does not matter which points in the model correspond to the chosen basis, which accounts for the efficiency of the search. Classification or perceptual grouping of features can be used to make the search over scene features more efficient, for example, by making use of only special basis tuples.

3. WEIGHTED VOTING IN GEOMETRIC HASHING

We now generalize the formulation of the geometric hashing technique. Let us 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 $\mathbf{q}_{k,i} \in \Re^2$. 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 $\mathbf{q}_{k,l}$ are computed in the appropriate coordinate system. We typically use the hash function and coordinate system as described in the previous section. There will be a sepa-

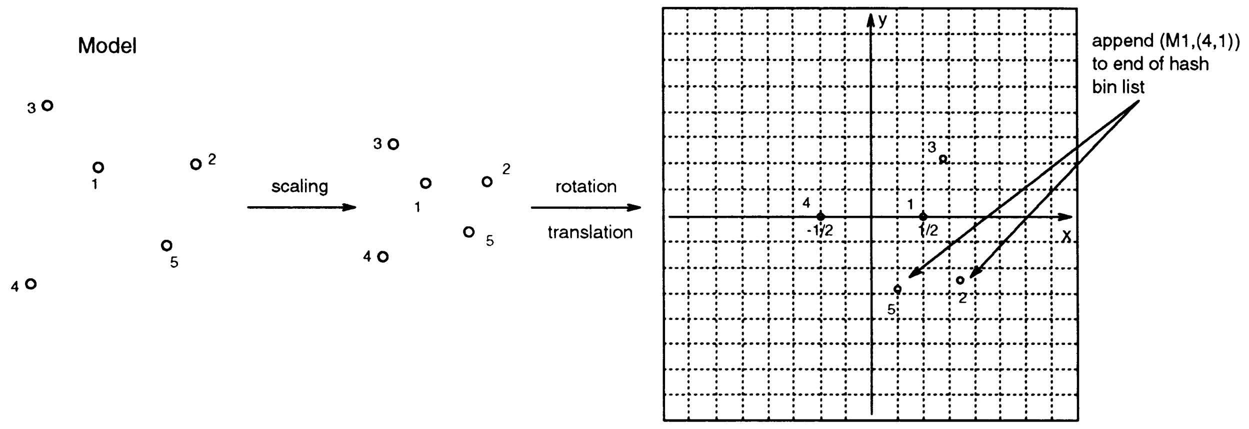


FIG. 2. Determining the hash table entries when points 4 and 1 are used to define a basis. The models are allowed to undergo rotation, translation, and scaling.

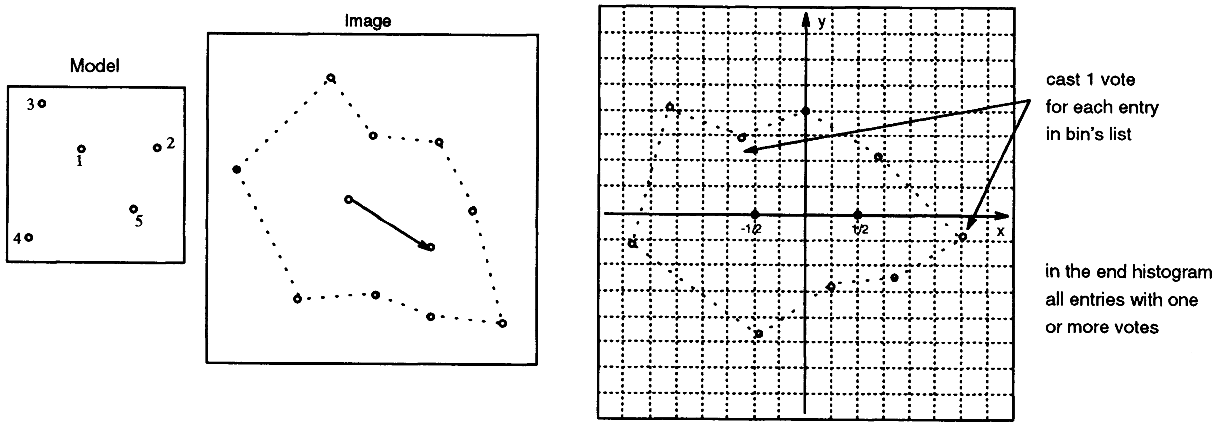

Model
Image
3
1
2
5
4
y
x
-1/2
1/2
cast 1 vote for each entry in bin's list
in the end histogram all entries with one or more votes

FIG. 3. Determining the hash table bins that are to be notified when two arbitrary image points are selected as a basis. The allowed transformation is similarity.

rate entry in the hash table for every ordered triple of distinct points (i, j, l) and every model M_k.

During the recognition phase, a collection of scene points $\mathbf{p}_1, \ldots, \mathbf{p}_S$, are observed, with each $\mathbf{p}_l \in \Re^2$. 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 0 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 in 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 $[(d_x, d_y)C^{-1}(d_x, d_y)^t$ 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 for a well-justified weighted voting approach, using an exponentially decreasing function of a scaled Mahalanobis distance.

4. A BAYESIAN FORMULATION

In this section, we present the Bayesian extension to the geometric hashing algorithm. Although we will treat only the similarity transformation case, the approach is more general and can be applied to other transformation classes. A more detailed analysis that is transformation independent appears in [15].

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, $\mathcal{S} = \{\mathbf{p}_l\}_{l=1}^{S}$. We assume that two points of $\mathcal{S}$ are chosen as a basis pair, say, $\mathcal{B} = \{\mathbf{p}_\mu, \mathbf{p}_\nu\}$. Thus, given $\mathcal{S}$, we wish to determine if a model is present with $\mathcal{B}$ as a basis.

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. 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:

• $(M_k, i, j, \mathcal{B})$ means that the 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 $\mathbf{p}_\nu$, using the basis pair $\mathcal{B} = \{\mathbf{p}_\mu, \mathbf{p}_\nu\} \subset \mathcal{S}$.
• $\mathcal{S}'$ 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 basis set $\mathcal{B}$.

Using this notation, we compute

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

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 basis set.

A maximum-likelihood approach to object recognition results if we ask to find the maximum of Eq. (1) over all possible M_k, i, j, and $\mathcal{B}$. The resulting model/basis combination is the most likely match given the collection of hash values generated from $\mathcal{S}'$. Note that the collection $(M_k, i, j, \mathcal{B})$ are not mutually exclusive. If the model M_k is present, then there will be multiple possible pairings of model features to scene basis sets. A reasonable assumption for practical purposes is the following: "If there is no match, then the probability value of even the maximum winner will not be large, whereas if there is a match, then for some choice of $\mathcal{B}$ (and most likely, multiple choices) there will be a large probability value for some $(M_k, i, j, \mathcal{B})$." If there are multiple models present, then several model/basis combinations will share a large probability. Since one can always pass the 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

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

for every possible $(M_k, i, j, \mathcal{B})$. 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. The assumptions are reasonable if the features are chosen judiciously.

With the above assumptions, and using standard probabilistic derivations based on Bayes' theorem [3], the probabilities of Eq. (1) may be rewritten as

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

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

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

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 Eq. (3), noting that the constant of proportionality becomes a constant additive factor, whence we see that we must maximize

$$\log(\Pr((M_k, i, j, \mathcal{B}))) + \sum_{\mathbf{p}_\xi \in \mathcal{S}'} \log\left(\frac{\Pr(\mathbf{p}_\xi \mid (M_k, i, j, \mathcal{B}))}{\Pr(\mathbf{p}_\xi)}\right) \tag{4}$$

over all possible model/basis combinations and basis selections.

This is in essence what geometric hashing does. We first posit a fixed 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 $\mathcal{B}$.

Note that it is not necessary to exhaustively consider all model/basis combinations and basis selections in order to come up with an answer. Due to the redundant representation, there is more than one $i, j, \mathcal{B}$ combination that induces the recovery of model M_k.

As Eq. (4) dictates, the contribution that a particular hash value based on a point $\mathbf{p}_\xi$ in the scene should lend to a model/basis combination (M_k, i, j) wil be equal to a log-probability ratio

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

The ratio compares the probability of "hashing" to the location where $\mathbf{p}_\xi$ hashes using $\mathcal{B}$, under the assumption that model M_k is present with the basis (i, j) matching the chosen basis $\mathcal{B}$, to the probability without this assumption. The probability of hashing to a given location is simply the probability density value of a given location in hash space. Hash value probability densities for a variety of feature distributions and transformations have been studied and are presented elsewhere [15].

The log-probability ratio measures the logarithm of the factor by which the probability increases or decreases due to the conditioning hypothesis. We "weight" the hashes by a quantity that depends on the actual hash value. Most of the hash values occur near the origin of the hash space. In essence, this means that hashes that occur in the less populated regions of the hash table (distant from the origin) will be given more weight.

The log-probability ratio 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, the probability density value 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 Eq. (4) that initially each combination should have a bias amount equal to the logarithm of its *a priori* probability. If every model/basis combination and basis selection is equally likely, then this term may be dropped.

5. DENSITY OF ENTRIES IN HASH SPACE

The previous section shows that geometric hashing, with the appropriate weighted voting, can be interpreted as a Bayesian maximum-likelihood object recognition system. The weighted voting scheme, which extends the binning notion of geometric hashing as presented by Lamdan, is critical to this interpretation.

To compute the weights that should be used, we must analyze 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 density functions for similarity transform hashing, using the hash function computation indicated in Section 2.

Our interest here is to compute the log-probability ratio

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

as used in Eq. (4).

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 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 depends on density functions which are 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 $n - 2$ locations 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 1, and the log-probability ratio is close to 0. 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 1 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

$$\begin{pmatrix} \nu_1 - \mu_1 & -\nu_2 + \mu_2 \\ \nu_2 - \mu_2 & \nu_1 - \mu_1 \end{pmatrix} \begin{pmatrix} u \\ v \end{pmatrix} = \begin{pmatrix} \xi_1 - (\mu_1 + \nu_1)/2 \\ \xi_2 - (\mu_2 + \nu_2)/2 \end{pmatrix}. \qquad (5)$$

If we assume Gaussian perturbations of the positions of the three points, we can show that the resulting distribution of the computed invariant (u, v) is in the first-order Gaussian centered at (x, y), and with covariance C given by

$$C = \frac{(4(x^2 + y^2) + 3) \cdot \sigma^2}{2\, \|\mathbf{p}_\mu - \mathbf{p}_\nu\|^2} \cdot I, \qquad (6)$$

where I is the identity matrix (see [17, 18]). Note that the covariance function is diagonal and depends on both the location of the hash point (x, y) in the noise-free case as well as on 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 $n - 2$ nonbasis 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:

$$f(u, v) = \sum_{q=1}^{n-2} \frac{1}{2\pi\sqrt{|C_q|}} \exp\left(-\frac{1}{2}(u - x_q, v - y_q) C_q^{-1} (u - x_q, v - y_q)^t\right). \qquad (7)$$

Here, C_q is the covariance matrix at (x_q, y_q) given by Eq. (6). 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 Eq. (5). The distribution function of the hash location (u, v) will depend on the distribution of the points in the scene. Suppose that the scene points are distributed according to a Gaussian distribution with a fixed but *unspecified* variance; the center of the distribution is assumed to coincide with the origin of the scene coordinate system. The expected distribution of hashes (u, v) in hash space, as determined by the solution to Eq. (5), *will be independent of the actual variance value in feature space* and is given by

$$g(u, v) = \frac{12}{\pi} \frac{1}{(4(u^2 + v^2) + 3)^2}. \qquad (8)$$

A derivation of this result appears in [16]. A large value for the variance will result in a distribution of points that is approximately uniform over the scene "window"; however, the distribution of hashes in hash space will still follow Eq. (8). 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

$$\log\left(\frac{(\Pr(\mathbf{p}_\xi \mid (M_k, i, j, \mathcal{B}))}{\Pr(\mathbf{p}_\xi)}\right) = \log\left(1 + \frac{f(u, v)}{S \cdot g(u, v)}\right), \qquad (9)$$

where point $\mathbf{p}_\xi$ hashes to location (u, v) using basis $\mathcal{B}$, f is given by Eq. (7), and g is given by Eq. (8). Accordingly, Eq. (9) provides the formula for the weighted accumulation amounts that contribute to the log-probabilities, as computed in Eq. (4).

The function f depends on $(M_k, i, j, \mathcal{B})$ in the sense that $f(u, v)$ requires knowledge of the location of the hash entries of model M_k using model basis $(\mathbf{q}_{k,i}, \mathbf{q}_{k,j})$ and explicitly depends on the scene basis separation (namely, $\|\mathbf{p}_\mu - \mathbf{p}_\nu\|$. Accordingly, let us denote f by $f_{M_k,i,j,\mathcal{B}}$, and the hash location of $\mathbf{p}_\xi$ under basis $\mathcal{B}$ by (u_ξ, v_ξ). We thus have that the log-probabilities $\log(\Pr((M_k, i, j, \mathcal{B}) \mid \mathcal{S}'))$, for a fixed $\mathcal{B}$, are ordered in the same order as

$$\sum_{\mathbf{p}_\xi \in \mathcal{S}'} \log\left(1 + \frac{f_{M_k,i,j,\mathcal{B}}(u_\xi, v_\xi)}{\mathcal{S} \cdot g(u_\xi, v_\xi)}\right). \qquad (10)$$

It is the function of the geometric hashing algorithm to compute approximates to Eq. (10) for a variety of $(M_k, i, j, \mathcal{B})$.

Let us now consider how the parameters of the problem are reflected in Eq. (10). For a given scene basis $\mathcal{B}$, each scene point $\mathbf{p}_\xi \in \mathcal{S}$ contributes to votes for model/basis combinations $(M_k, i, j, \mathcal{B})$. The contribution depends on the location (u_ξ, v_ξ) to which $\mathbf{p}_\xi$ hashes using the basis $\mathcal{B}$. The contribution will be close to 0 (NB: $\log(1) = 0$) as long as $f_{M_k,i,j,\mathcal{B}}(u_\xi, v_\xi)$ is small. Thus for a given hash (u_ξ, v_ξ) based on a fixed basis $\mathcal{B}$, nonzero contributions will go only to model/basis combinations (M_k, i, j) such that $f_{M_k,i,j,\mathcal{B}}(u_\xi, v_\xi)$ is large. This value is large only when (u_ξ, v_ξ) lies near an entry $(x_q, y_q, M_k, i, j, l_q)$ among the $n - 2$ entries generated by model M_k with basis $\mathbf{q}_{k,i}$, $\mathbf{q}_{k,j}$. Here, "nearness" is measured in terms of the covariance matrix C_q associated with the entry. Accordingly, for a scene point $\mathbf{p}_\xi$ and its hash location (u_ξ, v_ξ), we only need to increment accumulators for model/basis that have entries that lie near (u_ξ, v_ξ).

The amount of the increment to a particular model/basis combination is equivalent to one of the summands of Eq. (10). Assuming that the entries in hash space for model M_k with basis (i, j) are separated, then the nearest entry to (u_ξ, v_ξ) dominates the computation of $f_{M_k,i,j,\mathcal{B}}(u_\xi, v_\xi)$. That is, only one term in Eq. (7) dominates, and the closer (u_ξ, v_ξ) lies to its nearest entry (x_q, y_q), relative to C_q, the larger the contribution.

The $f(u_\xi, v_\xi)$ term is modulated by the number of scene points, $\mathcal{S}$, representing the amount of background clutter and also by the background distribution, evaluated at the hash location, $g(u_\xi, v_\xi)$.

The larger the number of clutter features ("noise"), the larger the value of the denominator, and thus the smaller the relative contribution from the imaged object's features (the "signal"). Above a certain amount of clutter, the contribution will become insignificant and the signal, will not be distinguishable from the background noise; i.e., the imaged object will not be recognizable.

The background distribution serves a similar function. Hashes to locations near the origin, which are common, contribute relatively less than hashes that are far from the origin, which are more indicative of a match to a model.

The impact of occlusion on the ability to recognize an imaged object is incorporated in the numerator of the argument of the log function. Occlusion will result in fewer *visible* model features and consequently in a smaller contribution from the $f(u_\xi, v_\xi)$ terms. In the presence of scene clutter, a *highly* occluded imaged object will not be recognized.

One can use expression (9) to produce reasonable estimates for thresholds to determine whether a hash is close to an entry. An alternative approach for determining such thresholds has been discussed in [9]: the recognition problem is modeled as a statistical occupancy problem and thresholds are derived as a function of sensor uncertainty, and the scene and model complexities.

6. THE BAYESIAN GEOMETRIC HASHING ALGORITHM

We now summarize the algorithm that allows us to extend the geometric hashing method to 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 $\tau = (4(x^2 + y^2) + 3) \cdot \sigma^2$. Of course, τ can be reconstructed from the location (x, y) but it is often useful to store the precomputed value. 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.

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 $(x, y, M_k, i, j, l, \tau)$ are accessed. For each such nearby record, we record a weighted vote for the model/basis combination (M_k, i, j). From Eq. (9), the amount of the vote should be equal to

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

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. The value z is an approximation of the quantity (Eq. 9), which is the total contribution of the hash (u, v) to the model/basis (M_k, i, j), obtained by neglecting all terms in f except for the entry at (x, y).

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 $(x, y, M_k, i, j, l, \tau)$, 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 locations (x, y) and update the recorded increment corresponding to the hash that lies closer in the sense of the covariance radius r, which amounts to using the larger of z and z'. 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 $(x, y, M_{k_0}, i_0, j_0, l, \tau)$.

When all weighted votes are tallied, model/basis combinations whose total accumulated evidence exceeds a threshold may be passed onto a verification process. If the maximum accumulated evidence for no model/basis combination exceeds the threshold, the algorithm continues by iterating through different candidate basis pairs from the image.

A major advantage of the geometric hashing approach to model-based object recognition is its inherently parallel nature. There are a number of ways that the geometric hashing algorithm can be parallelized. Two basic algorithms (based on a "connectionist" view of the method and a "hash location broadcast" model respectively) together with some variants are described in [19].

7. IMPLEMENTATION DETAILS AND RESULTS

In this section, we describe in greater detail the automatic feature extraction and basis selection mechanism and present our experimental results. Our object recognition system employs real-world images such as photographs from books, or actual street scenes. Many alternative object recognition systems have used either synthetic or controlled-environment (laboratory) scenes of real objects; moreover, in many cases, the segmentation and/or feature extraction/selection was done by hand.

The results that we present here are those from an implementation of our system on an 8K-processor Connection Machine 2. Since this implementation was done, the language Concert/C [1] was used to implement a second-generation geometric hashing object recognition system based on a distributed model of computation [20]. Both systems produced good results, although it is notable that the distributed implementation on 16 workstations runs faster than the 8*K*-processor Connection Machine 2 implementation. Here, however, we present results based on the CM-2 implementation.

The system incorporates an automatic feature extraction mechanism: the mechanism makes use of a boundary-following algorithm (eight-connectivity is assumed) applied to the output of the edge detection stage. We used the Cox–Boie edge detector [2], which gives good results and has been in common use at our lab for a number of years. Edge detection is followed by a divide-and-conquer polygonal approximation algorithm [6]. The output of the edge detection stage is typically a collection of curves, and polygonal approximations for each of those curves are determined. Curves shorter than 100 pixels are not considered. The vertices of the different approximating polygons compose our feature set.

More sophisticated approaches, such as spline fitting, could be used to determine the point feature set. We have used a simple feature extraction process and have discarded all information about the features other than the coordinate positions in order to concentrate on the search aspects of geometric hashing. In order to select among basis sets during the recognition phase, we simply chose randomly among the different bases: the two basis members were selected, with replacement, uniformly from the scene point-feature set. A basis set might be discarded if the basis separation is too small or too big; the cutoff values for our experiments were 130 and 550 pixels respectively, whereas our test inputs were 512 pixels on the side. Our scenes contained between 80 and 170 feature points, at most 16 of which belonged to a model in the database, which is a small signal-to-clutter ratio. Using the simple randomized algorithm for selecting bases requires that on average approximately 35 basis pairs be selected before recognition occurs. More elaborate basis selection mechanisms are also possible. In particular, it is possible to use grouping strategies to select likely basis pairs; alternatively, a partial randomized accumulation of weighted votes can be used to screen out poor basis selections.

The database used for our experiments contained 32 models: 14 of the models were military aircraft, whereas the remaining 18 were automobiles (six vehicles seen from three different viewpoints each). For the aircraft models, profile drawings of 14 military aircraft from [14] were scanned using a Microtek 300 color/gray-level scanner. These drawings are not photographs but are instead schematic drawings that we *assume* are drawn roughly to scale. Although our scanner is capable of a resolution of 300 dpi, we used 120 dpi resolution to digitize the drawings. For the automobile models, we obtained photographs of six different automobiles seen from three different viewpoints: the camera was at the height of the midline, with its optical axis pointing at the middle of the automobile's long side. The three viewpoints corresponded to azimuth values of roughly $-45°$, $0°$, and $+45°$. The average distance from the automobiles was approximately 15 m. The photographs were subsequently scanned at a resolution of 75 dpi. The 14 aircraft and 18 automobile models contained in our database appear in Table 1.

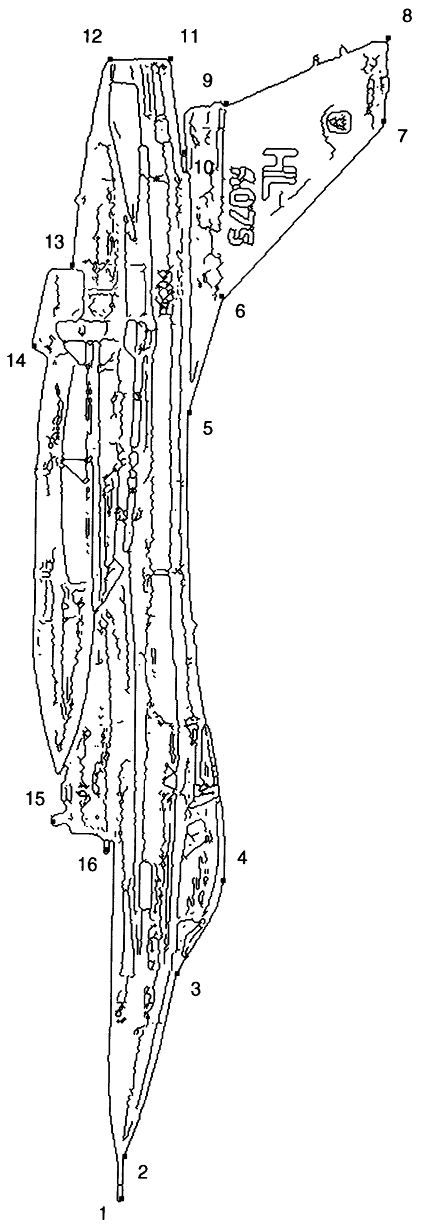
The digitized images were then processed using the Cox–Boie edge detector [2]; the value of the filter's σ was typically equal to 2.0. We call the result of this stage the *edge map*. No other filtering or preprocessing was performed. Example edge maps of the models are shown in Fig. 4.

Next, 16 points were manually subselected from the contour of each model. The points were chosen to coincide with points of discontinuity in the tangent direction of the model's contour (i.e., vertices or points of very high curvature), or with points of maximum curvature. Within a class of models (i.e., aircraft or automobiles) we made a point of selecting the *same* feature on *all* the models: the tip of the radar in all aircraft, the tip of the rudder, etc. The intention behind this choice was to make the task of the object recognition system more difficult by *decreasing* the number of distinguishing feature markers.

The model point selection is performed manually during the construction of the database and thus takes place off-line. Candidate feature points may be presented by means of the edge tracking algorithm. Figure 4 shows the points selected for three of the database models: the F-16

TABLE 1
The Models of the Database

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	
Chevrolet *Astro* (lateral)	Chevrolet *Astro* (oblique frontal)	Chevrolet *Astro* (oblique rear)
Dodge *Dart* (lateral)	Dodge *Dart* (oblique frontal)	Dodge *Dart* (oblique rear)
Ford *Econoline150* (lateral)	Ford *Econoline150* (oblique frontal)	Ford *Econoline150* (oblique rear)
Chrysler *Horizon* (lateral)	Chrysler *Horizon* (oblique frontal)	Chrysler *Horizon* (oblique rear)
Honda *Civic* (lateral)	Honda *Civic* (oblique frontal)	Honda *Civic* (oblique rear)
Volvo *S.-W.* (lateral)	Volvo *S.-W.* (oblique frontal)	Volvo *S.-W.* (oblique rear)



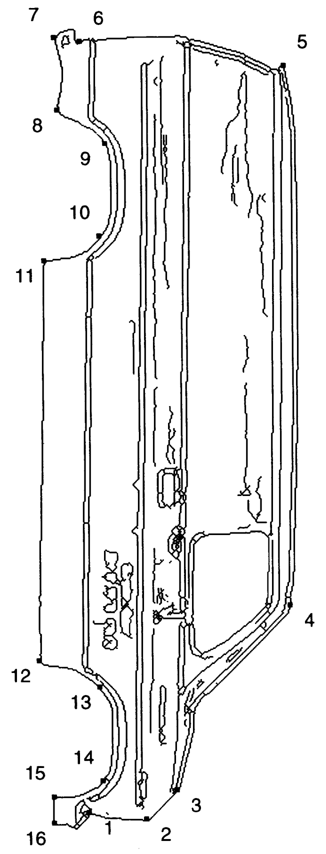


FIG. 4. The edge maps and the selected feature points for the database models of the F-16 *Falcon*, the Ford *Econoline150*, and the Sea Harrier.

Falcon, the Sea Harrier, and the Ford *Econoline150* (lateral view).

We then selected a number of photographs of the same military aircraft, but from a *different source* [21]. 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 algorithm uses 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

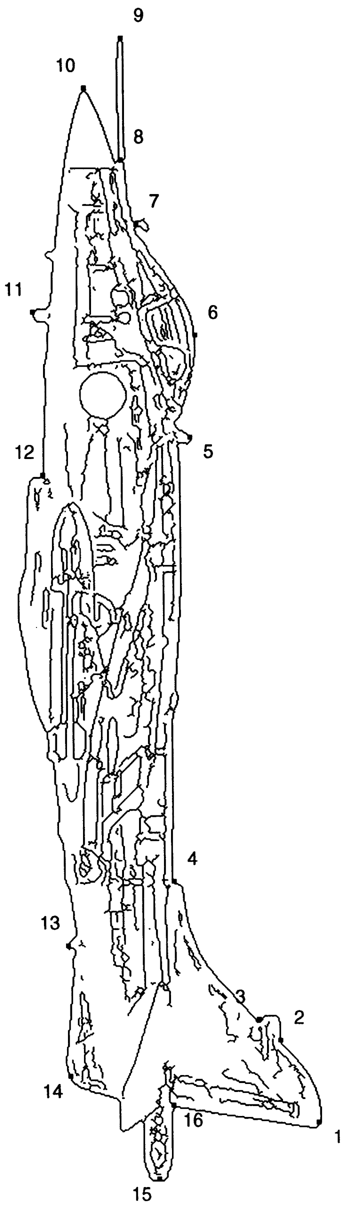


FIG. 4—*Continued.*

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.

We also obtained additional photographs of automobiles. The automobiles in our test photographs were from various locations in New York City. The only thing that the automobiles in the test photographs and those used to build our database have in common, other than the approximately similar viewpoint, is the automobile type.

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 3D 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 were performed. A polygonal approximation of the different edge maps provides the points of the feature set. Figures 5 through 7 show the digitized photographs for three of our test inputs together with the corresponding edge maps and extracted point features.

In Fig. 5, we can see that the original photograph of the F-16 was taken with the camera positioned below the airplane's midline and toward the back of the aircraft. Further, the airplane is banking to the left. A total of 80 feature points were extracted from the edge map of this test input. The extraneous-features-to-model-features ratio for this test input was 4 : 1.

The original photograph of the Sea Harrier (Fig. 6) was very small; a juxtaposed pencil helps estimate the actual size of the original image. 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. In the edge map of this test input, we identified a total of 169 feature points. This represents an extraneous-features-to-model-features ratio of 9.5 : 1.

The photograph of the Ford *Econoline150* (Fig. 7) was taken from the driver's side of the automobile, unlike that of the model which was taken from the passenger's side. Instead of augmenting our database with entries corresponding to that viewpoint, we *reversed* the test input and used its reflection around the vertical axis as a test input; this explains why the lettering on the front door of the vehicle appears backwards. In more recent work, we have endowed the recognition system with left/right reflection invariance, but that work is not presented here. The extraneous-features-to-model-features ratio for this test input was 5.3 : 1.

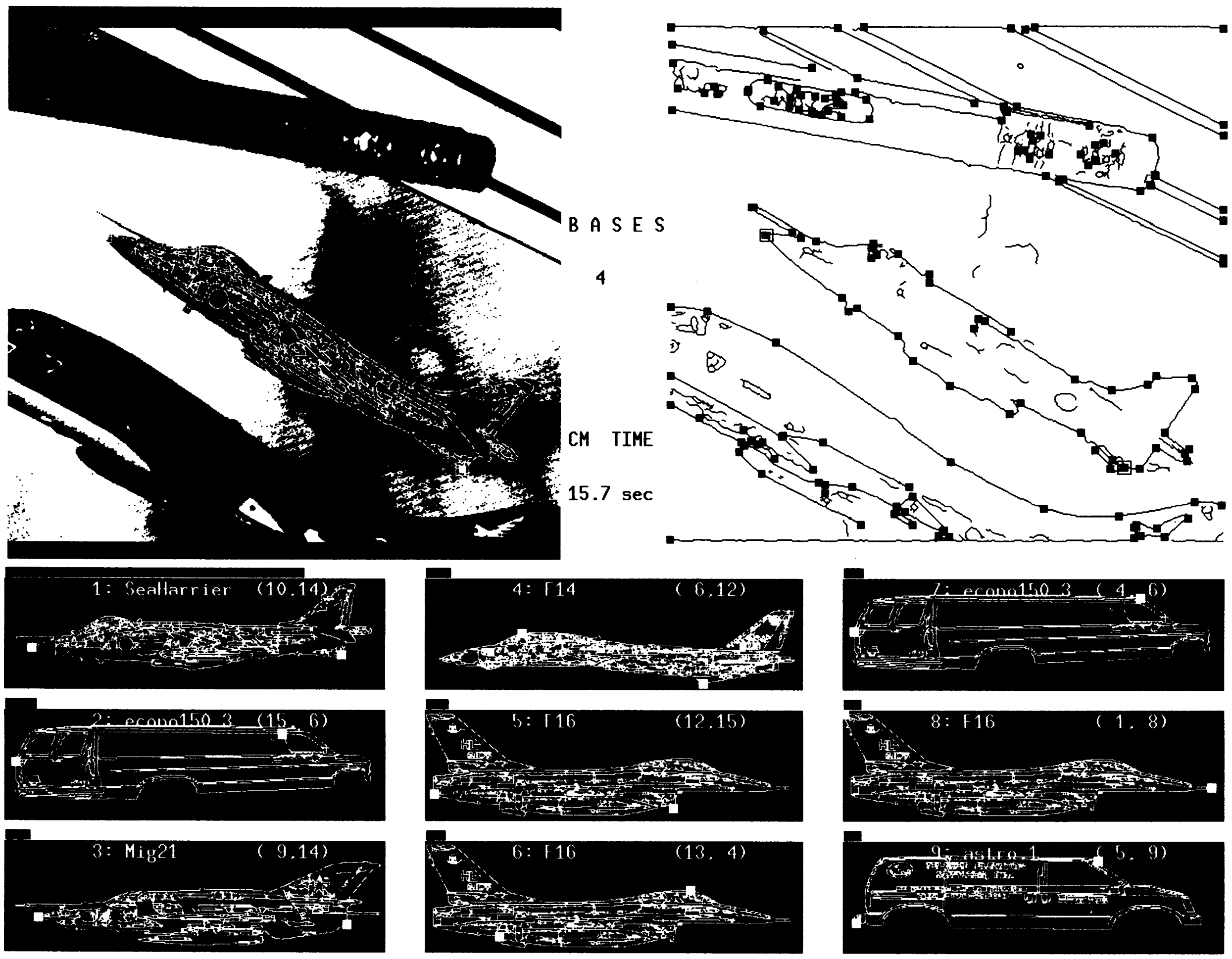


FIG. 9. The output of the implementation of our system on the Connection Machine. The test input (Sea Harrier) is shown at the top left. The edge map together with the automatically extracted point features is shown at the top right; the basis selection that led to recognition is also marked. A total of four basis selections were required, and the elapsed time was 15.7 s for the voting accumulation over all four trials on an 8K CM-2. The nine panels below show the top model/basis combinations for the 4th trial, where the basis is indicated with white square dots. The bars above each panel provide a length encoding of the total accumulated log-probability value for the model/basis combination. The top scoring database model, scaled, rotated, and translated by placing the bases in correspondence, is shown overlaid on the test input.

In Figs. 8 through 10 we show the output of our system's implementation for three test inputs. The database model that receives the maximum weighted vote is scaled, rotated, translated, and shown overlaid on the test input. In the bottom half of each screendump, the nine top ranking model/basis combinations are shown in order of decreasing accumulated evidence (column-major order). For each of the nine combinations, the model name and the corresponding retrieved basis are indicated. The point features of the retrieved basis are marked along the contour of the corresponding model. Above each model, bars providing a length-encoded representation of the evidence (which is a log-probability) are also shown. In this implementation, we perform *no verification*: whenever the maximum accumulated evidence exceeds the present threshold (see Section 5) we stop the basis pair selection and display the results for the nine top-ranking winners. 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.

In all our experiments, the true model/basis combination has been 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, a postprocessing stage could be used to verify the suggested matches: for our database of 32 models, there are 7680 possible model/basis combinations; if the generated nine top-ranking model/basis combinations are checked, we will have still achieved a considerable speedup over

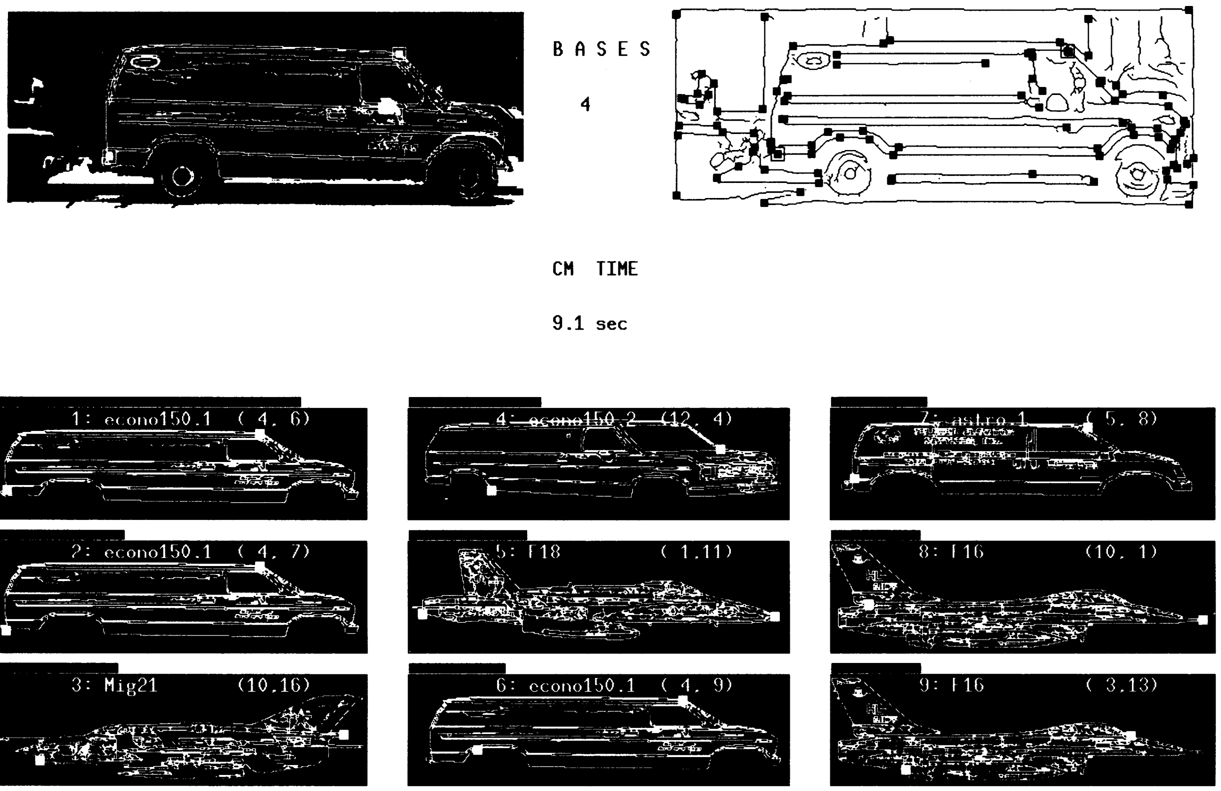


FIG. 10. The output of the implementation of our system on the Connection Machine. The test input (Ford em Econoline150) is shown at the top left. The edge map together with the automatically extracted point features is shown at the top right; the basis selection that led to recognition is also marked. A total of four basis selections were required, and the elapsed time was 9.1 s for the voting accumulation over all four trials on an 8K CM-2. The nine panels below show the top model/basis combinations for the 4th trial, where the basis is indicated with white square dots. The bars above each panel provide a length encoding of the total accumulated log-probability value for the model/basis combination. The top scoring database model, scaled, rotated, and translated by placing the bases in correspondence, is shown overlaid on the test input.

the alternative of exhaustively checking all possible matchings. Extensive experimentation has shown that the accumulated evidence for the ninth model/basis combination is considerably smaller than the evidence for the winning model/basis combination, indicating that the method is robust.

REFERENCES

1. J. Auerbach, 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. R. Boie 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.*
3. E. Charniak and D. McDermott, *Introduction to Artificial Intelligence,* Addison–Wesley, Reading, MA, 1985.
4. M. Costa, 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.*
5. M. Costa, R. Haralick, and L. Shapiro. Optimal affine-invariant matching: Performance characterization, in *Proceedings SPIE Conference on Image Storage and Retrieval, San Jose, CA, January 1992.*
6. R. Duda and P. Hart, *Pattern Classification and Scene Analysis,* Wiley, New York, 1973.
7. D. Gavrila and F. Groen, 3*D* object recognition from 2*D* images using geometric hashing, *Pattern Recognit. Lett.* **13**(4), 1992, 263–278.
8. W. Grimson and D. Huttenlocher, On the sensitivity of geometric hashing, in *Proceedings of the International Conference on Computer Vision, Osaka, December 1990.*
9. W. Grimson and D. Huttenlocher, On the verification of hypothesized matches in model-based recognition, *IEEE Trans. Pattern Anal. Mach. Intell.* **13**(12), 1991, 1201–1213.
10. R. Hummel and H. Wolfson, Affine invariant matching, in *Proceedings of the DARPA Image Understanding Workshop, April 1988.*
11. Y. Lamdan and H. Wolfson, Geometric hashing: A general and efficient model-based recognition scheme, in *Proceedings of the International Conference on Computer Vision,* pp. 238–249, Computer Society Press, Los Alamitos, CA, 1988.
12. Y. Lamdan 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.*

13. Y. Lamdan, J. Schwartz, and H. Wolfson, Object recognition by affine invariant matching, in *Proceedings of the IEEE Computer Vision and Pattern Recognition Conference, Ann Arbor, MI, June 1988,* pp. 335–344, Computer Society Press, Los Alamitos, CA.

14. D. Richardson, *The World's Major Military Jets*, Salamander Books, 1990.

15. I. Rigoutsos, Massively parallel bayesian object recognition, Ph.D. thesis, New York University, August 1992.

16. I. Rigoutsos 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, HI, June 1991,* Computer Society Press, Los Alamitos, CA.

17. I. Rigoutsos 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.*

18. I. Rigoutsos 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.*

19. I. Rigoutsos and R. Hummel, Massively parallel model matching: Geometric hashing on the connection machine, *IEEE Comput. Spec. Issue Parallel Process. Comput. Vision Image Understanding* **25**(2), 1992, 33–42.

20. I. Rigoutsos and R. Hummel, Distributed Bayesian object recognition, in *Proceedings IEEE Conference on Computer Vision and Pattern Recognition, New York City, NY, June 1993,* Computer Society Press, Los Alamitos, CA.

21. W. Sweetman and R. Bonds, *The Great Book of Modern Warplanes,* Portland House, New York, 1987.