DATABASE-RETRIEVAL ORIENTED APPROACH FOR MODEL-BASED OBJECT RECOGNITION

G. Sudhir, College of Engg., Univ. of California, Riverside, U.S.A.
Bir Bhanu, College of Engg., Univ. of California, Riverside, U.S.A.

ABSTRACT
Model-based recognition of objects when the number of model objects becomes large is a challenging problem which makes it increasingly difficult to view the object recognition problem as "find the best match" problem. We present a database-retrieval oriented approach for this case wherein the goal is to index, retrieve, rank and output a few top-ranked models, according to their similarity with an input query object. The approach consists of three stages: (1) feature-based representation of model objects and object-feature correspondence analysis; (2) clustering and indexing of the model objects in the feature space, and (3) ranking indexed models based on mutual information with query object. We believe that this approach has practical advantages for large model databases where the best-match approach can be unreliable. It also finds applications in semi-automatic object recognition tasks which involve interaction with humans. Experimental results are presented using a variety of data to demonstrate the merits of our approach.

1 INTRODUCTION
Model-based object recognition is a powerful approach which involves invariant feature domain representations of models of different objects to generate a model database, and matching these to the features of a new observation of an object, to select the best match. However, as the number of models in the model database increases, it becomes increasingly difficult to view the object recognition problem as "find the best match" problem. There are two main reasons for this: (1) Firstly, the discriminating power of a set of known features becomes increasingly insufficient for the purpose of finding the correct matching model, for all the objects. Moreover, there may be a lack of knowledge about the optimal set of features; and (2) Secondly, the sensitivity of features to changes in object pose, image formation geometry, sensor parameters, etc. adds to the problem of correct recognition.

In this context, the model-based object recognition problem shares similarity with the content-based retrieval problems where the insufficiency of features affects in capturing the notion of content. However, the database retrieval approach - wherein, the goal is index, retrieve, rank and output a few top-ranked models as the most probable matches - has proved to be useful for various practical tasks [1]. Hence, for the case of large model databases, taking a database-retrieval stance to model-based recognition is a practically useful one for various interactive applications. One such case is that of model-based recognition of targets in Synthetic Aperture Radar (SAR) images. As has been observed and quantified in our earlier research efforts [2], one of the main characteristics of SAR images of objects is the sensitivity of features (scattering centers) to environmental variation (pose variation). That is,
2 RELATED RESEARCH AND CONTRIBUTIONS

Recently, Fuss et al. [3] have used correspondence analysis (CA) for statistical structuring of pictorial databases for content-based retrieval. They represent images using aggregate features based on multiple cues, and do CA on images and attributes. However, in the factor space generated by CA, they use the feature projections only to explain the dominant factors and not as seeds for clustering objects. They propose an Ascendant Hierarchical Classification method [4, 5] for structuring the projections of the images in the factor space and report their experiments on example image databases of 54 images. For large model databases consisting of thousands of objects with sensitive and noisy features (like the SAR target database we have used), the projections of objects are spread out almost randomly in the factor space, and as a result, a tree-based classification may not be appropriate. Hence, researchers propose the combination of feature-based clustering of objects in factor spaces and a triangle-inequality-based indexing of the objects inside each cluster. We believe that it is a useful alternative to hierarchical clustering and indexing approaches.

Triangle-inequality (TI) based indexing schemes take advantage of the triangle inequality of distance measures to reduce the number of direct comparisons in a threshold search. This indexing scheme has recently gained a lot of attention in the context of content-based retrieval from image databases [6, 7]. Barros et al. [6] have applied the idea to a real image database and Berman et al. [7] have reported the performance of different algorithms for the selection of key objects as well as handling multiple distance measures on a set of image features. However, application of the TI using the distance between objects in the feature space has the limitation of not dealing with any redundancies of feature set in capturing a description of the database objects. Our approach deals with it in a systematic manner, since it considers all the distances in the factor space after object-feature correspondence analysis. An important advantage of computing distance in feature space (instead of feature space) is that, undesirable weighting bias towards a large set of redundant features is clearly avoided. This is because, CA eliminates the effect of a set of redundant features by grouping them together in the feature space. Exploiting this, our approach computes the distance of the model objects from the “feature-group” as one entity.

Berman et al. [7] have reported on ranking the retrieved model objects according to the computed lower bounds on distances, in the context of content-based retrieval of images. However, in the context of object recognition, since the feature-based representation of model objects is usually insufficient to capture all the properties of objects, the ordering based on lower bounds may not be accurate. Hence, we propose the use of feature-based

representation only to quickly index a set of model candidates for a query object which can be further ranked based on mutual information which is a robust measure of similarity between the query and model objects. Recently, mutual information (MI) has been reported to be a more robust measure than normalized cross-correlation, for its relative insensitivity with respect to variations in illumination [8]. Vishal et al. [8] use MI measure in their algorithm for alignment of a model object to a new observation, assuming that the new observation belongs to the same model object. In our context, we use the MI measure for comparing a candidate model object to a query object. Our motivation to use MI is to compare the model and query objects according to their statistical dependencies, in order to robustly rank a set of candidate models for a query object. For the empirical estimation of MI, we have used the technique reported in [9].

The main contribution of our paper is that it presents a new, systematic approach to model-based object recognition from the database-retrieval point of view. The advantages of our approach are, (i) it deals with feature redundancies using CA, (ii) it proposes feature-based clustering of model objects in factor spaces, followed by TI based indexing of objects for each cluster, as an useful alternative to standard hierarchical clustering and tree-based indexing methods, and (iii) advocates a robust approach for ranking the initial retrievals using mutual information between a query object and a candidate model object. Our approach compares the query and model objects by considering their information content first in the feature domain (as TI based indexing in factor space generated by CA), and then in the original data domain (as mutual information based ranking of initial retrievals). The former step helps in quickly pruning a large number of models from consideration for a query object and the latter step ranks the candidates in a more robust manner.

3 TECHNICAL APPROACH

Figure 1 illustrates our conceptual approach. There are two phases: (1) Database construc-

Figure 1: Overview of our database-retrieval oriented approach for object recognition.
objects and features in a common reduced-dimension factor space. These projections are further analyzed and clusters of objects are formed in the factor space. For each cluster, a set of key objects are computed to enable the best-based indexing of the objects in the factor space. Finally, the model information—the model objects, factors, clusters, key objects for each cluster, distances of model objects from key objects in each cluster—is assembled to generate the model database. In the retrieval phase, an unknown query object is processed by the invariant feature extraction module to represent it in the feature space. Then its features are input to the model database to index and retrieve a set of candidate models which are further ranked by the mutual information module to output only the top 10 ranked retrievals as the most probable match to the query input object.

In the following, we briefly describe the key modules in Fig. 1 in the two phases. More details can be found in [10].

3.1 DATABASE CONSTRUCTION PHASE

The first step in this phase is the extraction of a set of invariant features for the model objects to represent them in the feature space. In our case of SAR target databases, we have used translation-invariant features [6]. We present details about these features in Sec. 4.

The rest of this subsection assumes that features are already extracted.

3.1.1 Object-feature Correspondence Analysis: The main advantage of feature-based representation of model objects is that the dimension of the feature space is much smaller than dimension of the original model data space. However, there can be redundancy in the features for describing the model database. This causes an undesirable weighting bias towards a set of redundant features, when any distance is computed between two objects in the feature space. We propose a systematic way of dealing with this using CA of model objects and their invariant features.

Correspondence Analysis (CA) is a type of factor analysis [4, 5]. Like any factor analysis method, it provides a compact representation in a low dimension factor space of large sets of numerical data (which in our case is a matrix $D = \{d_{ij}\}$, see below). It shares the linear algebra of finding the factor axes with other factor analysis methods. Moreover, in CA, the coordinates of the data points are defined so that the usual Euclidean metric in the factor space corresponds to the $\chi^2$ distance between the points. Thus, the analysis is in terms of the independence of the data. Furthermore, unlike other factor analysis methods, CA assigns asymmetric roles to rows (objects) and columns (features). This permits simultaneous representation of both objects and features in a common factor space which not only helps interpretation of the factor space but also makes clustering of objects easier (see Sec. 3.1.2). We exploit these advantages of CA in our approach. In the following, we describe CA as relevant to the analysis of objects and features.

Let there be $M$ model objects and $N$ features for each object. For large number of model objects, $N \ll M$. The model objects and their feature representations are as the matrix $D = \{d_{ij}\}$, $1 \leq i \leq M, 1 \leq j \leq N$, where, the rows identify the model objects and the columns identify the features. For CA, $D$ is processed using the following sequence of steps [5]:

1. Compute the normalized data matrix $K = \{k_{ij}\}$, $1 \leq i \leq M, 1 \leq j \leq N$ using $k_{ij} = \frac{d_{ij}}{\sum_{j=1}^{N} d_{ij}}$, where $d_{ij} = \sum_{i=1}^{M} d_{ij}$ and $d_{ij} = \sum_{j=1}^{N} d_{ij}$. 2. Compute the $\chi^2$ matrix $H = K^T K$ where $K^T$ stands for transpose of a matrix $K$. Note that $H$ is a $N \times N$ matrix and its elements in terms of original data matrix $D$ are given by $h_{ij} = \sum_{m=1}^{M} \sum_{n=1}^{N} d_{mn}^2$, where $d_{ij} = \sum_{m=1}^{M} d_{mi} d_{mj}$ and $d_{ij} = \sum_{n=1}^{N} d_{ni} d_{nj}$. 3. Compute Singular Value Decomposition of $H$ to find the eigenvalues $\lambda_1 \leq \lambda_2 \leq \ldots \leq \lambda_N$ and eigenvectors $\nu_{1i} \leq \nu_{2i} \leq \ldots \leq \nu_{Ni}$. Let $\lambda_i \leq \lambda_j$ if $\lambda_i \geq \lambda_j, 2 \leq i \leq j \leq N$. The eigenvalues $\lambda_i$ to $\lambda_j$ determine the variance of the system and their corresponding eigenvectors ($\nu_{1i}, \nu_{2i}, \ldots, \nu_{Ni}$) determine the factor spaces and hence are called factors. Let $F_i = \nu_{1i}, 1 \leq i \leq N$ be the factors; 4. Compute the ratio $r_i = \frac{\lambda_i}{\sum_{j=i+1}^{N} \lambda_j}$. These ratios indicate the percentage of the total variance of the system that each factor $F_i$ explains. Note that only a few factors can explain up to 90% of the total variance of the system. Let $P$ denote that number, where $P \leq N - 1$; 5. Project the model objects $O_i, 1 \leq i \leq M$ along each factor axis $F_i, 1 \leq i \leq P$ using $O_i = \sum_{j=1}^{N} O_{ij} F_j$, where $Z = \sum_{j=1}^{P} \sum_{i=1}^{M} d_{ij} F_j, d_{ij} = \sum_{i=1}^{M} d_{ji}, d_{ij} = \sum_{j=1}^{N} d_{ji}$ and $F_j$ is the $j$th element of factor $F_j$. The quantity $O_i$ represents the scalar coordinate of the object $O_i$ along factor $F_j$. 6. Project the features $A_j, 1 \leq j \leq N$ along each factor axis $F_k, 1 \leq k \leq P$ using $A_k = \sum_{i=1}^{M} d_{ij} O_i F_k$, where $d_{ij} = \sum_{i=1}^{M} d_{ij}$. The quantity $A_k$ represents the scalar coordinate of the feature $A_k$ along factor $F_j$. After these steps, the objects and features are both represented in a common, reduced dimension factor space. Often, only the first two factors determine more than 70% of the total variance (it is so in all the experimental examples reported in this paper). In such a case only the first two dimensions can be considered in the following clustering and indexing phase.

3.1.2 Cluster Analysis in Factor Space: We first identify different “feature-groups” in the factor space and then use their centroids as the seeds for generating clusters of model objects. There are two advantages of this: (1) the number of features is far less compared to the number of model objects; thus, determining different groups of features in the factor space is not computationally intensive and can be easily supervised; (2) feature-groups, which act as the seeds around which objects get clustered, also explain automatically why the object cluster was formed, this can be exploited to give a meaningful annotation to a cluster as the set of model objects which are mainly described by their seed feature-group.

3.1.3 Key-object based Indexing: Since the number of feature clusters is usually small (there are just 4 to 6 clusters of features in our experiments), there can be still a large number of model objects inside each cluster. In order to efficiently index objects inside the clusters, we employ the TI based indexing scheme. The TI based indexing schemes rely on comparing a set of key objects to the database objects according to a distance metric, and storing the computed distance. The basic idea is to exploit the TI at the retrieval time to quickly compute the lower bounds on the distance of each database object from the query. The reason behind all these schemes is the fact that the distance between two objects cannot be less than the distance of their distance to any other object. Mathematically, if $O$ is a database object, $Q$ is a query object and $K$ is some key object, the inequality $d(O, K) \geq |d(O, Q) - d(K, Q)|$ always holds. Thus, by comparing the database and query

is no difference as far as the analyst objects are concerned since this metric has the same eigen-structures as the full $\chi^2$ matrix, except that the largest eigenvalue becomes trivial [9].
objects to a third key object, a lower bound on the distance between the model and query objects can be obtained. If a threshold \( T \) on the distance between a model and query object is known (or given), this lower bound can be compared to \( T \) to eliminate any further consideration all those models whose lower bound is more than \( T \). Note that, if the distances of all the model objects from the key object are stored and hence are readily available, the only distance computation that needs to be done in order to know all the lower bounds is that between the query and key objects. Since the distance computation is often much more expensive compared to simple subtraction involved in the computation of lower bounds, speed-up is achieved in indexing the model objects for each query. In our experiments, for simplicity, we have selected two key objects for each cluster. These two key objects are the two furthest apart ones in a cluster. Thus the total number of keys used to index the model database in our approach is twice the number clusters.

3.1.4 Algorithm for Clustering and Indexing: In the following, we describe the sequential steps for clustering and indexing: 1) Consider the projections of \( M \) features along the first \( F \) dominant factors. Closely projected features explain the system similarly and are redundant. Group such closely projected features into a 'feature-group' representing a single feature class. Let there be \( C \) different feature classes. Define the center of feature-group as the centroid of the feature projections of that group; 2) Generate \( C \) clusters of model objects using centers of each feature-group as the seed and using nearest-neighbor (NN) rule in the feature space; 3) For each cluster, select two model objects \( b_k \) and \( b_q \) which are further apart in the cluster. They form two key objects for the cluster; 4) For each cluster, compute the distances of model objects in the feature space with each of the key objects; 5) For each cluster, store (i) indices of the model objects, (ii) indices of the key objects and (iii) distances of the model objects to key objects, in the model database.

3.1.5 Building model database: 

![Figure 2: Contents of the Model Database.](image)

The original model data is stored inside the model database. This data is necessary when the mutual information of the query object with each of the candidate model objects needs to be computed. Also, the factors computed by the object-feature analysis are stored as part of the model database. These factors will be used in the retrieval phase. The stored clusters contain two main items: key-objects and distances of all the model objects to all the key-objects along with indices to the model objects. Note that these distances are in the factor space. Finally, the empirical entropies of all the model objects are computed a priori and stored as part of the model database. This can speed-up the computationally intensive process of empirical estimation of mutual information since model entropies are readily available.

3.2 RETRIEVAL PHASE

In this phase, a new observation is given in the form of a query object. This query object is pre-processed as follows: 1) Compute the same invariant features as done for each model object. Let \( Q = [q_1, q_2, \ldots, q_M] \) be the query features. 2) Project the \( Q \) along each factor axes \( P_k \), \( 1 \leq k \leq F \) using \( Q_k = \frac{Q \cdot P_k}{P_k \cdot P_k} \). Where \( Z, X, Y, P \) come from the stored factor details in the model database. 3) \( Z = \sum_i q_i \). The quantity \( Q \) represents the scalar coordinate of the query \( Q \) along factor \( P_k \). 4) Compute the distance (in factor space) of the query object from each of these two key objects. Using these distances and the stored distances of the other model objects from the two key objects, retrieves the model objects whose TT-based lower bounds are less than a threshold \( TH \). These initial retrievals form candidate matching objects to the query. The candidate models are further ranked using their mutual information (MI) with the query object [9].

4 EXPERIMENTAL RESULTS

In this section, we present the experimental results on several real SAR image databases. The resolution of the real SAR data is 1-foot-per-pixel. Various model databases consist of SAR images of (i) objects at a particular depression angle, or (ii) objects of a particular configuration, or (iii) objects at a particular articulation [10]. The corresponding test data consists of SAR images of (i) objects at a different depression angle, or (ii) objects of a different configuration, or (iii) objects at a different articulation. In each case, test data is an independently acquired one from which query objects are selected randomly.

The set of features we have used to represent each object in CA are listed in Table 1. The first nine features are computed on the histogram of relative distance between scatterers (see [10] for more details). Since the relative distances are used, the first nine features are translation invariant. The next three features are computed on the gray level histogram of the SAR image pixels within ROI. While we compute the features 10 to 12 based on histogram of intensity values and use them in CA, we do not consider them in clustering and indexing. This is because they are computed based on gray level values of only the top scatterers which may be unreliable. The last two features are also translation invariant. In all our experiments, we consider only 11 of the 14 features (discounting features 10 to 12) from Table 1, for clustering and indexing.

Figure 3 (left) shows the projections of features in the feature space spanned by the first two dominant factors for Database 1 (see Table 3). Note that the moon, sun, max and median fail closely in factor space and are redundant for describing the target database, whereas the model, soil and Aas fall distant to any other feature and hence form independent descriptors of the database. The other features look spread in the factor space and so we do not use

*This threshold can be a priori estimated as the average distance in factor space between pairs of similar objects, by considering a large set of similar object pairs.
Table 1: 14 features used to represent SAR targets in correspondence analysis.

<table>
<thead>
<tr>
<th>Feature</th>
<th>Feature</th>
<th>Feature</th>
<th>Feature</th>
<th>Feature</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number</td>
<td>Description</td>
<td>Number</td>
<td>Description</td>
<td>Number</td>
</tr>
<tr>
<td>1</td>
<td>mean of HRI</td>
<td>6</td>
<td>skewness of HRI</td>
<td>11</td>
</tr>
<tr>
<td>2</td>
<td>std of HRI</td>
<td>7</td>
<td>kurtosis of HRI</td>
<td>12</td>
</tr>
<tr>
<td>3</td>
<td>max of HRI</td>
<td>8</td>
<td>energy of HRI</td>
<td>13</td>
</tr>
<tr>
<td>4</td>
<td>mode of HRI</td>
<td>9</td>
<td>entropy of HRI</td>
<td>14</td>
</tr>
</tbody>
</table>

HRI = Histogram of relative distances between scatterers

HI = Histogram of intensity values

width = Diameter of the object along the width of the image

height = Diameter of the object along the height

Figure 3: Left: Projection of 11 features (features 10, 11 and 12 are not used; see text for explanation) in factor space spanned by first two dominant factors for the entire Database 1 (see Table 2). Right: Projection of both objects and feature-groups in the same factor space for Database 1. The features are marked * and the objects (targets) are marked by + in the factor space. Note that the objects appear to be almost randomly spread out in the factor space. The four feature-groups are used as the seeds for forming four clusters of objects using nearest-neighborhood approach.

In conclusion, we present a database-retrieval oriented approach to model-based object recognition, for large model databases. We have presented detailed results on several real SAR target databases to demonstrate our approach. These are difficult databases since feature invariance may be small (20% to 50%) [2]. The technique is general and has applica-

Table 2: Model databases of SAR targets.

<table>
<thead>
<tr>
<th>Model/Test Data Differences</th>
<th>Total no. of model objects in database</th>
<th>Total no. of test objects for database</th>
<th>No. of dominant factors (≥90% of total variance)</th>
<th>% of total variance explained by first 2 factors</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depression angle diff.s (Database 1)</td>
<td>1821</td>
<td>1861</td>
<td>3</td>
<td>85.31</td>
</tr>
<tr>
<td>Configuration diff.s (Database 2)</td>
<td>694</td>
<td>1621</td>
<td>4</td>
<td>70.84</td>
</tr>
<tr>
<td>Articulation diff.s (Database 3)</td>
<td>606</td>
<td>255</td>
<td>3</td>
<td>79.29</td>
</tr>
</tbody>
</table>

Table 3: Results on various model databases of SAR targets.

<table>
<thead>
<tr>
<th>Model/Test Data Differences</th>
<th>No. of random queries from the test data</th>
<th>Avg. no. of retrievals after indexing in factor space</th>
<th>No. of cases in which a target with correct ID and pose was found in top 10 retrievals</th>
</tr>
</thead>
<tbody>
<tr>
<td>Depression diff.s</td>
<td>200</td>
<td>84 (43.18%)</td>
<td>185 (92.2%)</td>
</tr>
<tr>
<td>Configuration diff.s</td>
<td>200</td>
<td>64 (32.25%)</td>
<td>170 (85%)</td>
</tr>
<tr>
<td>Articulation diff.s</td>
<td>200</td>
<td>96 (48.47%)</td>
<td>239 (90.5%)</td>
</tr>
</tbody>
</table>

tions in other interactive object recognition applications involving large model databases. It also helps interactive discovery and acquisition of new models to update database. Currently, mutual information based ranking takes more than 90% of the time during recognition (retrieval) phase. The performance of our approach could be further improved by: (a) efficient computation of mutual information between a query and a candidate model object using a coarse-to-fine strategy, (b) overlapped boundaries in nearest-neighbor clustering method, and (c) efficient methods to adapt factors to dynamically changing databases. We are currently investigating along these directions.

REFERENCES

Figure 4: Performance curves for the SAR target databases.


International Conference on Advances in Pattern Recognition

Proceedings of ICAPR 98, 23–25 November, Plymouth, UK