PERGAMON

Pattern Recognition 32 (1999) 1737–1749

PATTERN RECOGNITION
THE JOURNAL OF THE PATTERN RECOGNITION SOCIETY

# Recognition of partially occluded objects using neural network based indexing

Navin Rajpal[a], Santanu Chaudhury[b,*], Subhashis Banerjee[a]

[a]*Department of Computer Science and Engineering, IIT Delhi, New Delhi, India*
[b]*Department of Electrical Engineering, IIT Delhi, Hauz Khas, New Delhi, 110016, India*

Received 23 January 1997; received in revised form 9 November 1998; accepted 9 November 1998

**Abstract**

In this paper, a new neural network based indexing scheme has been proposed for recognition of planar shapes. Local contour segment-based-invariants have been used for indexing. Object contours have been obtained using a new algorithm which combines advantages of region growing and edge detection. Neighbourhood constraints have been applied on the results of indexing for combining hypotheses generated through the indexing scheme. Composite hypotheses have been verified using a distance transform based algorithm. Experimental results, on real images of varying complexity of a reasonably large database of objects have established the robustness of the method.


*Keywords*: Object recognition; Invariant indexing; Neural networks; Hypothesize-and-test; Contour segments

## 1. Introduction

Object recognition is finding applications in the newer areas of content-based image retrieval, video indexing, etc., in addition to classical application domains like industrial inspection and robotic manipulation. These applications necessitate development of robust recognition techniques which can work with uncalibrated views of the scene and can handle reasonable degree of occlusion of the objects. Motivated by these considerations, in this paper, we present an object recognition scheme based on local invariants. Main features of our recognition scheme are extraction of local invariant features by mapping contour segments between dominant points to a canonical frame, indexing based on these features, generation of composite hypothesis about the objects present in the scene and their pose using neighbourhood constraints on the results of indexing and subsequent verification using an algorithm based on weighted distance transform [1]. Silhouette is extracted from the image using a new pre-processing technique which combines advantages of region growing and edge detection [2,3]. Key contribution of this paper is the use of an indexing mechanism in which the capabilities of neural network have been exploited for computing association between image features and object models in a robust and efficient fashion.

The majority of techniques [1–6] which are capable of recognizing occluded objects of different sizes and orientations work well for model base containing small number of objects. For a model base containing large number of objects, invariant-based indexing schemes have been proposed in the literature [7,8]. In all these methods, an index is typically a pattern vector of invariant

*Corresponding author. fax: + 91-11-6966606.
*E-mail address:* santanu@ee.iitd.ernet.in (S. Chaudhury)

0031-3203/99/$20.00 
PII: S0031-3203(98)00164-2

measurements obtained from the image. An indexing function maps these pattern vectors to those object models which posses a set of features having similar invariant measurements. Multidimensional hashing function have been used for this purpose because it allows simultaneous indexing on all elements of measurements. However, it is clear that the indexing in this context is a classification process by which image measurements are being mapped to feature classes defined by object models. Hashing-based schemes cannot directly exploit the properties and nature of individual class definitions. The conventional hash-table-based indexing scheme used by Lamdan [7] and Zisserman et al. [8] is difficult to implement in case of multidimensional indexing. The dependence of query time on the dimension $d$ of pattern vector is generally very steep for any algorithm for approximate closest-point queries, at least $2^{\Omega(d)}$ [9].[1] This exponential dependency is due to the increase in the number of neighbours in the higher-dimensional space. Geometric hashing technique works efficiently for two- or three-dimensional feature vectors but higher-dimensional cases suffer from this problem. In order to implement an indexing scheme which is robust against imaging noise and measurements, requires limited storage (unlike inverted index created for each feature and feature combinations) and is sensitive to feature class definitions, a new classifier based indexing scheme has to be designed. On the basis of the above motivation, we have formulated an indexing scheme using neural networks and compared its performance with geometric hashing method. It is also feasible to use some statistical classifier but neural network based classifiers generally require fewer trainable parameters than conventional probability density function classifiers [10]. It has been demonstrated in experiments [11] that many neural network classifiers can accurately estimate posterior probabilities and these neural network classifiers can sometimes provide lower error rates than PDF classifiers using same number of trainable parameters. The neural network can be trained off-line for learning the indexing function and the generalization capability of the neural network can be usefully exploited for ensuring robustness against errors in measurement [12,13]. Hardware implementation also favour simpler artificial neural network methods which are more robust than statistical ones with respect to parameter tuning [10].

Most of the local invariants proposed in the literature are not always easily computable. They may not remain computable under all possible conditions of occlusion. For example, bitangents used in [8] are not applicable for all types of objects, and contour in between control points can become occluded if concavities are well separated on object boundaries. Local invariants suggested by Weiss [14] involving computation of osculating circles is computationally costly. In this paper, we make use of a simple invariant defined by the curve specified by consecutive dominant points on the contours. Measurement on the canonical representation of the curve provides the invariant vector. This invariant ensures immunity to substantial occlusion and reduces the number of possible feature combinations to be considered for invariant computation. However, since localization of dominant points is not an error-free operation, we need a robust indexing scheme to take care of errors in identification of dominant points and distortion (because of noise) of the contour. The neural network-based indexing scheme helps in overcoming this problem and has the following advantages over hash table based indexing.

1. Result of indexing generates a value between 0 and 1 signifying the degree of match between an input feature and a model feature class.
2. The input feature can match to multiple classes in model base with different match values.
3. The system can be made less sensitive to noise by training the neural network using large feature sets extracted from different orientations of model set as explained in Section 4.

These properties have further been explored as explained later in this paper to generate composite hypotheses using neighbourhood constraint criteria in order to restrict the search space in the verification step. The recognition has been tested successfully for both similarity and affine transformations for a reasonably large database of objects and can be easily extended for plane projective transformations. The recognition scheme takes care of reasonable amount of occlusion as the hypothesis about the presence of objects and their pose is generated using the features extracted only from the local contour segments.

## 2. Dominant point identification

The recognition scheme proposed in this paper works with contour-based local invariants. This requires a reliable technique for obtaining object boundaries. For extraction of object contour, a scheme which combines the advantages of region growing and edge detection has been used. The steps used are:

1. Apply averaging operator $[3 \times 3]$ on the image to reduce noisy points.
2. Apply Sobel's edge operator on the averaged image.
3. Apply a region growing algorithm on the gradient image. In the region growing process, adjacent pixels having difference between their gradient values within a defined threshold (we have used difference of 5 in gradient magnitude), are assigned common region

[1] Here $\Omega$ – indicates asymptotic lower bound. For a given function $g(n)$ we denote by $\Omega(g(n))$ the set of functions defined as $\Omega(g(n)) = \{f(n)$: there exist a positive constants $c$ and $n_o$ such that $0 \leqslant cg(n) \leqslant f(n) \forall n \geqslant n_0\}$.

label. Each region potentially represents an object contour segment.
4. Regions of size less than 20 pixels grown in the previous step are dropped because they are unlikely to represent object contours. Adjacent regions having different labels are merged together using simple connectivity considerations.
5. Outer boundaries of regions grown after steps (iii) and (iv) are extracted using Pavlidis [15] contour tracing algorithm.

This algorithm provides object contours reliably because:

1. Use of gradient image (unthresholded) eliminates the problem of encountering breaks/gaps during edge linking.
2. Contour tracing of merged boundary regions eliminates the need for edge linking using any complicated search process.

The silhouette obtained is decomposed into curve segments using dominant points which are extracted using curvature guided polygonal approximation technique [1]. The method is broadly divided into two steps.

1. Find discrete curvature at all boundary points after applying Gaussian smoothing function to the extracted contour [1,16].
2. With extrema of smoothed curvature function as initial break points extract dominant points on the contour by iterative Split and Merge technique [1,15].

The curvature-guided polygon approximation method, for extraction of dominant points is better than the other methods like finding maxima and minima [4] of curvature or using simple split–merge polygon approximation [15] because of the following reasons. Use of all maxima and minima of curvature as dominant points introduces additional points due to noise. Use of Gaussian smoothing to reduce noise [16] with high $\sigma$ value causes shift in the position of extrema. Split and merge algorithm of Pavlidis [15] with arbitrary initial starting points cannot provide invariant set of dominant points.

Examples of contour and dominant points extracted from 16 model objects and four scenes are shown in result section. The present dominant point extraction scheme provide dominant points which are invariant to similarity transformation. For affine invariant points we can use the method suggested by Lamdan [7]. However, a problem with this method is that it does not provide sufficient number of dominant points on the contour to take care of substantial occlusion.

## 3. Feature extraction

As discussed in the previous section, the contour is decomposed into smaller curve segments using dominant points. The next step in the recognition scheme is to extract similarity/affine invariant local features from these curve segments. Most of the recognition schemes based on local features [7,1,4] use only the dominant points or ratio of lengths and angle between the lines joining the dominant points as features. These features do not give a unique representation of the contour segment as the exact shape of a contour between similarly configured dominant points may be different as shown in Fig. 1. In the present scheme, the exact shape of the curve segment in addition to length ratio and angle is used for defining the local invariant. For extracting similarity/affine invariant feature, we have used a canonical frame construction scheme similar to that suggested by Zisserman et al. [8].

### 3.1. Similarity invariant features

For the case of similarity transformation, mapping of the curve segment to a line of unit length with one-to-one correspondence between their end points gives an unique invariant representation of the curve segment. One example of such a mapping is shown in Fig. 2. The curve segment between two dominant points B and C in Fig. 2a is mapped to a unit line segment B′C′ as shown in Fig. 2b, nine uniformly spaced samples along axis normal to reference axis are recorded as curve signature. A 20-dimensional similarity invariant feature vector is constructed for two adjacent curve segments between three dominant points, which include nine samples each for individual curve segments obtained after mapping to unit line segment, length ratio and angle between two lines obtained by joining end points (three dominant points) of the curve segments. This 20-dimensional feature vector is used as input for neural network-based indexing technique explained in the next section.

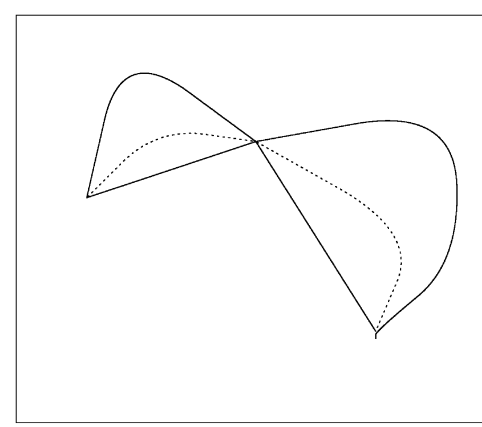

Fig. 1. Example showing two different curves having three control points with same angle and length ratio.

### 3.2. Affine invariant features

For extracting affine invariant features we need three distinguished points on the curve for canonical frame construction as explained by Lamdan [7], therefore, two adjacent curve segments between three dominant points (triplets), taken in order are mapped to canonical affine frame constructed using standard Cartesian basis. One such mapping is shown in Fig. 2. Two adjacent curve segments between points A, B and C, in Fig. 2a are mapped to Cartesian basis between points A′, B′ and C′ respectively, as shown in Fig. 2c. Eighteen uniformally spaced samples (nine along each axis) are taken as affine invariant input features as shown in Fig. 2c. The method can be further extended for extracting plane projective invariant features by mapping four dominant points on the contour taken in order to four corners of a unit square as suggested by Zisserman et al. [8].

Extraction of affine invariant or plane projective invariant features is not a difficult process, the only problem faced is in finding sufficient number of dominant points on the contour which are invariant to these transformations. A small number of dominant points leads to large curve segments between these points and the features extracted from these curves may not be strictly local and consequently the method fails for substantial occlusions.

## 4. Neural network based indexing

The feature vector extracted from a triplet in the image represent a point in 20-dimensional feature space. Because of noise and error in extracting features from different views, features extracted from a scene triplet, in general, span a region in invariant space. Regions corresponding to different triplets may overlap in invariant space because of similarity in shape after mapping on to the canonical frame. One example of such a case is shown in Fig. 3. Here the curve segments between points 0, 1 and 2 of object 1 as well object 2 have similarity in shape after mapping on to the canonical frame. Hence, the crucial problem of indexing is that of mapping a particular triplet in the image to one such region defined by the model base through an appropriate classification scheme. If a region, given a class label $C_s$ contains triplets corresponding to object models $O_x$, $O_y$ and $O_z$, then an image triplet classified into the class $C_s$ would generate hypothesis about objects $O_x$, $O_y$ and $O_z$.

Neural networks have been used for learning the class boundaries defined by the invariant features. This would provide more robust indexing scheme because any change in invariant values due to noise or local variations of the dominant points can be taken care of by the generalization capability of the neural networks. We can use large training set extracted from various images of

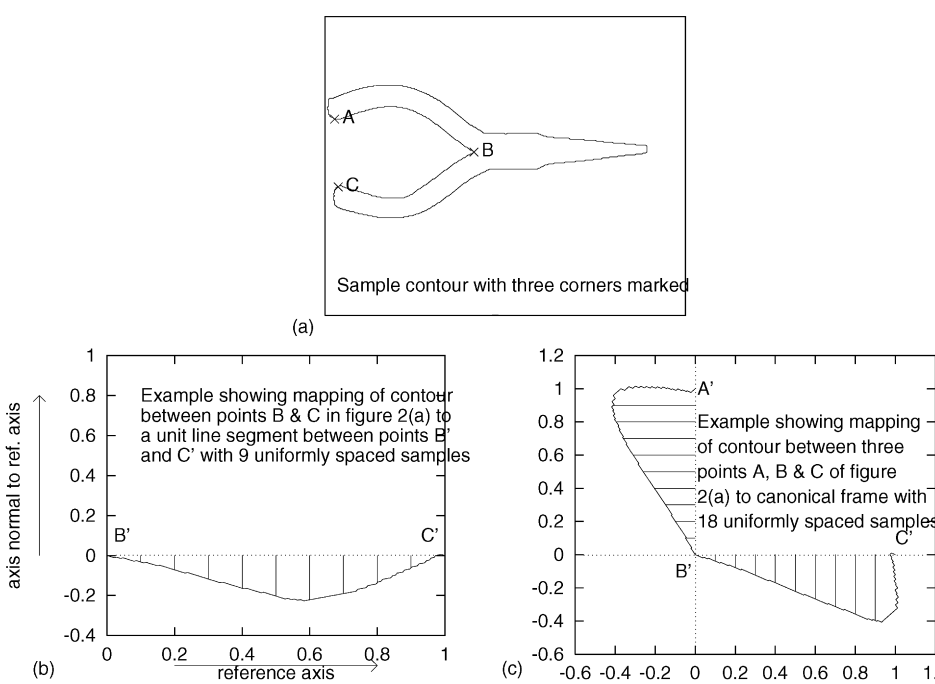


Fig. 2. Similarity and affine invariant features.

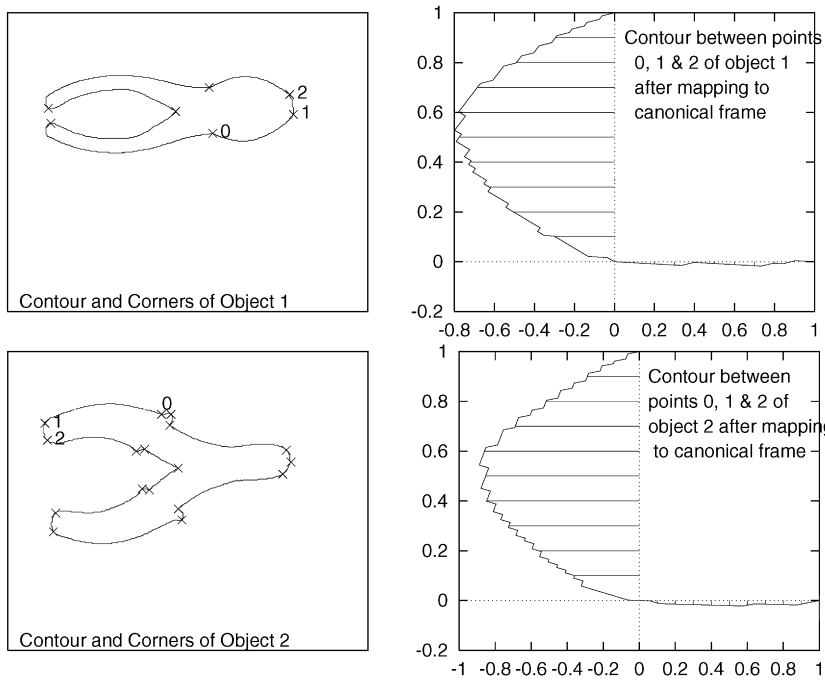


Fig. 3. Mapping of two triplets of different objects to canonical frame.

model objects captured under different lighting conditions and orientations. In this way, various kinds of noise will be automatically added to training samples and will make the system robust and less sensitive to noise. Further, the neural network can be used to identify alternative mappings for confusing cases by making use of their activation values. Appropriate definition of the activation function helps us to associate with a feature vector different degree of membership to different possible classes. In this paper, we suggest two networks for implementation of the indexing step. One is based on supervised and the other on unsupervised learning. These are discussed below.

### 4.1. Indexing using multi-layer perceptron

We use a multi-layer perceptron to learn the indexing function which maps an $N$-dimensional feature vector extracted from a scene triplet to a class representing triplet-object pairs. The architecture of the network used is shown in Fig. 4. For each class, we have a distinct network with one output element and a three node hidden layer. The input elements are common to all the networks and correspond to feature vectors. The network is trained using supervised learning with the output element corresponding to a class set to 1 when the input feature vector belongs to the class and to 0 otherwise. Consequently, during classification the unit with maximal response can be considered to indicate the correct class using a winner-take-all strategy. However, output response corresponding to classes having high degree of similarity with the input will also be non-zero and high. For this reason, any mapping indicated by networks having output response greater than 0.5 is stored in a table assuming that the feature vector may belong to any of these classes. These results are used later for generation of composite hypotheses using neighbourhood constraint criteria.

For training these networks, we use the standard back-propagation algorithm [12,13]. The training process is time consuming and is required for all the classes, but it is an off-line step and is required only once while generating the model database. The training process for this type of network is based on supervised learning, therefore, it requires pre-classification of feature vectors used for training. The curve segments corresponding to feature vectors belonging to the same class will have similar shape after mapping to canonical frame. Therefore, pre-classification of feature vectors is done by visual inspection of corresponding curve segments after mapping on to the canonical frame. Since we use seperate feed-forward network for each class, the training process for each class is de-coupled. This eliminates inter-class interferences and any error in manual grouping resulting in class definitions in the feature space does not adversely affect the overall scheme. The number of computation required for a mapping having $M$ number of classes and $N$-dimensional feature vector is of the $O(MN + M)$.

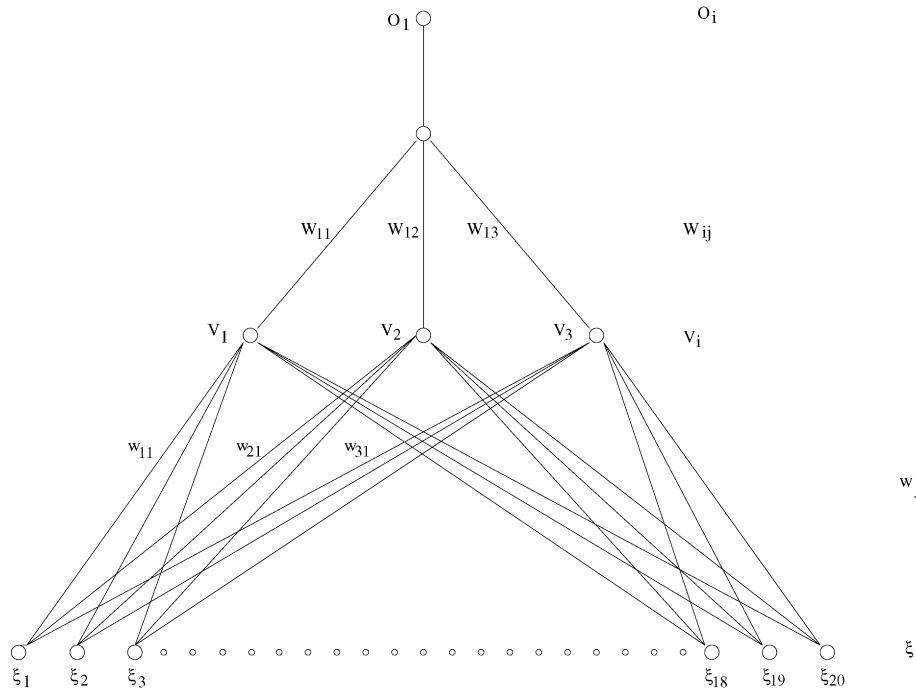


Fig. 4. The multi-layer perceptron architecture.

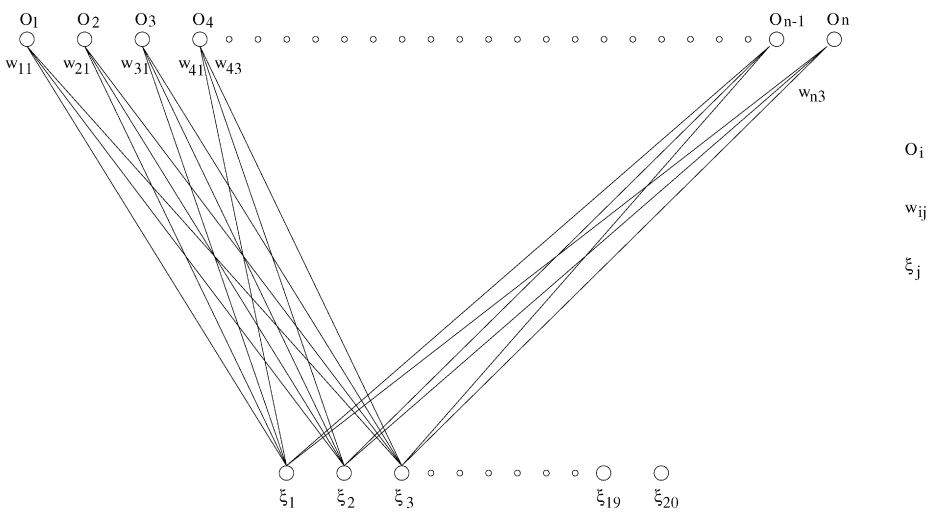


Fig. 5. The Kohonen network architecture.

### 4.2. Indexing using Kohonen network

Multi-layer back-propagation learning, suggested above is extremely slow and requires pre-classification of training feature vectors based on visual inspection. This can be avoided to some extent by an unsupervised learning approach. Various types of self-organizing nets for pattern classification like Maxnet, ART1, ART2, Kohonen's Network, etc., have been suggested in literature [12,13]. In this paper, we suggest a simple Kohonen network-based architecture as shown in Fig. 5. The characteristics of the network are given as follows.

- Twenty input elements correspond to 20-dimensional feature vector and $n$ output elements correspond to $n$ classes in invariant feature space.
- Each class as explained earlier represents set of pairs (triplet, object) and mapping of input feature vector to

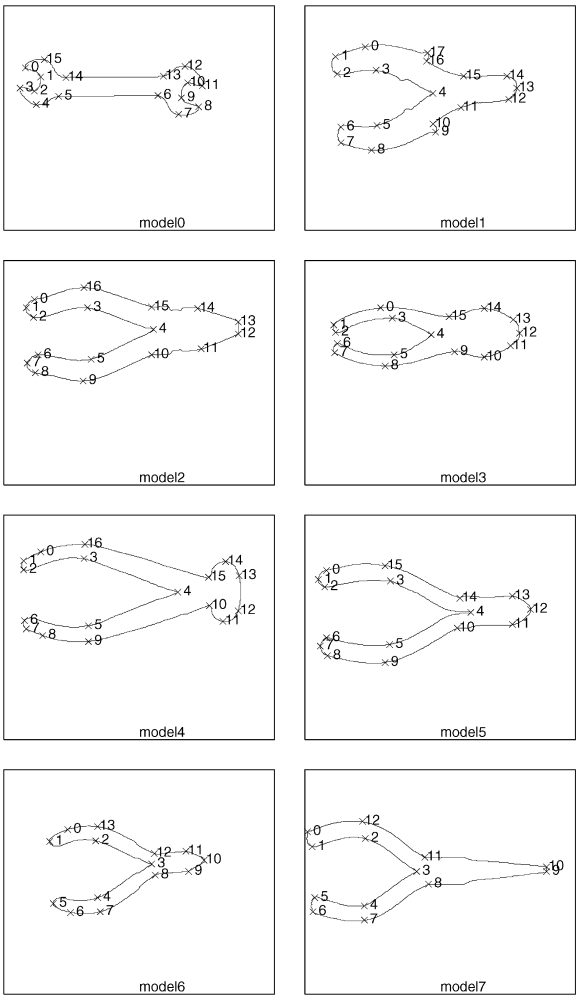


Fig. 6. Contours and corners of models 0–7.

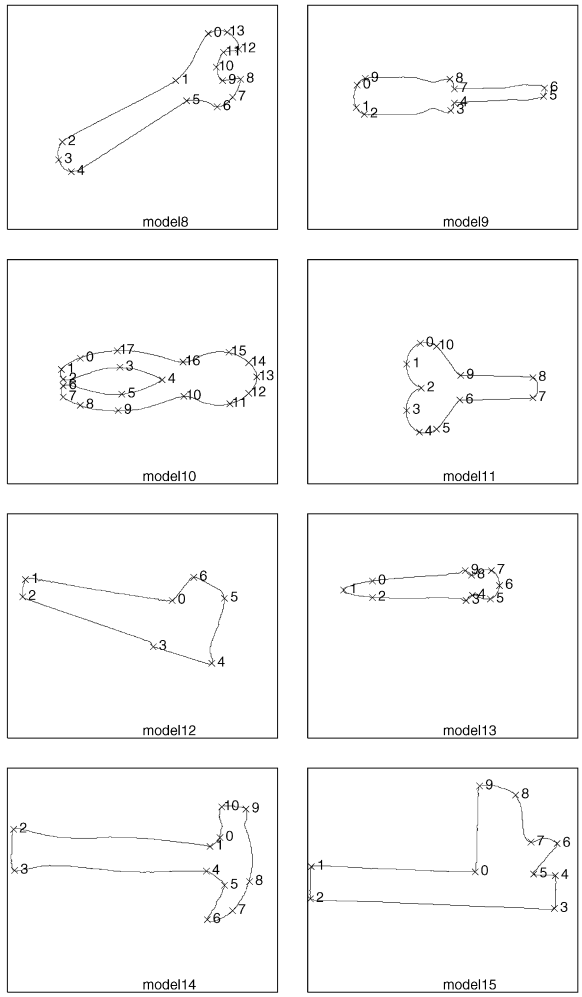


Fig. 7. Contours and conreners models 8–15.

output class is achieved autonomously by the system without external supervision, using competitive learning rule suggested by Kohonen [12,13].

- The exact number of output elements which are equal to the number of output classes depends upon the model database and the distribution of feature vectors in invariant space and is determined experimentally.
- After training of the network each output unit correspond to a class and hence set of pairs (triplet, object).

During recognition phase, the winning neuron represents the class to which a particular feature vector belongs. In order to obtain a measure of similarity, each Kohonen unit is associated with a Gaussian function $G(d, \sigma)$ for providing the output response, where $\sigma$ is width of Gaussian function, taken as the mean distance of all the feature vectors belonging to corresponding class, from the weight vector learned for that class; $d$ is the distance of input feature vector from the weight vector learned. For the Kohonen net, mapping generated by the winning neuron and two of its adjacent neighbours are stored in a table.

## 5. Composite hypotheses generation and verification

For an image triplet, indexing process will generate a number of competing hypotheses. For example, say, for a scene triplet $I$, we get output response close to one for classes $J$, $K$ and $L$. Let classes $J$, $K$ and $L$ consist of the model triplets $\{J_1, J_2, J_3, \ldots, J_{n(j)}\}$, $\{K_1, K_2, K_3, \ldots, K_{n(k)}\}$ and $\{L_1, L_2, L_3, \ldots, L_{n(l)}\}$, respectively. This requires $n(j) + n(k) + n(l)$ steps of transformation computation and verification for finding correct match for scene triplet I. To reduce the search space, we use neighbourhood

constraint. The fact used is that if a triplet in scene matches a model triplet, it is very likely that its preceding and succeeding neighbour should also match the preceding and succeeding neighbour of the same model triplet and also the transformation parameter between model and scene computed separately from these three triplets should be same. However, because of occlusion, the above expectation may be violated.

Response function $tp[i][j]$ for each scene triplet, which gives response to $j$th model triplet for $i$th scene triplet is defined to be the output of corresponding neural unit. With this value, we compute a combined response function $tpc[i][j]$ for each triplet as

$$tpc[i][j] = tp[i][j] + tp[i_p][j_p] + tp[i_s][j_s] \quad (1)$$

where $i_p$ and $i_s$ are preceding and succeeding neighbours of triplet $i$ in the scene and $j_p$ and $j_s$ are preceding and succeeding neighbours of the model triplet $j$. The value of combined response function will be high only for those triplets which satisfy neighbourhood constraint explained above. The combined response $tpc$ for the scene triplets which have values more than 1.5 are considered for verification. The steps of verification algorithm are given as follows:

1. Scene-model triplet pairs $(i, j)$ for which the value combined response function $tpc[i][j]$ is greater than 1.5 are stored in a table in decreasing order of the value of combined response function.
2. Pick a scene-model triplet pair from the table and compute six affine transformation parameters by mapping three control points of the model triplet to the scene triplet.
3. Using the six parameters computed in previous step, the entire model contour is translated to the scene contour.
4. Distance between translated model contour and the scene contour is computed using weighted distance transform technique as used by Tsang et al. [1].
5. If the value of distance between transformed model contour and the scene contour is very large (more than predecided threshold), reject this scene-model triplet match and goto step 2 for next scene-model triplet pair.
6. The small value of distance between the transformed model contour and scene contour confirms presence of the model in the scene. Remove matched portion of the scene contour as well as the corresponding scene-model triplet pairs from the table and goto step 2 for next scene-model triplet pair if left.

## 6. Experimental results

For experimental verification of the proposed scheme we considered an object set consisting of 16 types of tools like pliers, wrenches, spanners, etc. These objects were characterised by their 2D views. Model database was built from digitized images of size $300 \times 300$ and 256 grey levels. These were obtained in the laboratory using a CCD camera interfaced to MVP-AT IPS-20 system from Matrox. Scenes composed of multiple instances of these tools were considered for recognition experiments. In these scenes multiple objects partially occluded each other.

Silhouette of these objects in images were extracted using pre-processing and contour tracing steps described earlier. Two hundred and eighteen control points corresponding to 218 curve segments were extracted from these 16 objects contours using the procedure explained earlier. Contours and control points extracted from some objects are shown in Figs. 6 and 7. Twenty-dimensional similarity invariant feature vectors for 218 contour segments were also extracted by mapping 218 curve segments to unit line segment. The number of objects in the model base can increase without significant increase in the number of feature vectors because many contour segments belonging to the new objects will be similar to this set.

Comparison of neural network-based indexing with geometric hashing method and recognition results of some complex scenes consisting of multiple objects are discussed in the following subsections.

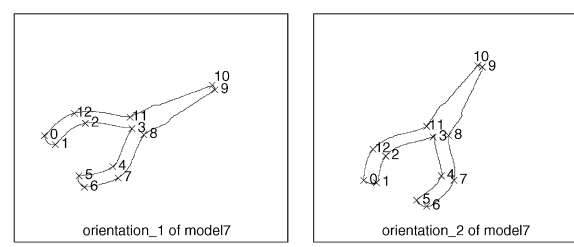


Fig. 8. Contours and cornres of two orientations of model 7.

Table 1
Results of indexing for two orientations of model 7 shown in Fig. 8

| Results | Orientation-1 | Orientation-2 |
|---|---|---|
| No. of triplets extracted | 13 | 13 |
| No. of correct mappings in case of Multi-layer perceptron | 11 | 11 |
| No. of correct mappings in Kohonen network | 11 | 11 |
| No. of mapped to neighbouring class in case of Kohonen network | 1 | 1 |
| No. of correct mappings using geometric hashing | 3 | 3 |

### 6.1. Comparison of indexing techniques

A Kohonen network consisting of 50 output units and 20 input units was trained sucessfully in 2000 iterations using initial value of $\eta = 0.99$ and width factor $\sigma = 13$. Both these factors were decreased exponentially using a value of $\alpha = 0.0005$. Using the result of Kohonen classification and by visual inspection the 218 feature vectors were divided into 50 classes for training of the Multilayer Perceptron model. Fifty, two-layer feed-forward networks were trained using momentum rate $\eta = 0.2$ and learning rate $\alpha = 0.15$. All these networks were trained successfully within 10,000 iterations.

We have also implemented geometric hashing-based indexing method for comparing it with neural network-based indexing technique. Geometric hashing was implemented by dividing 20-dimensional feature space into 20-dimensional hypercubes. Each axis in feature space is divided into ten buckets of uniform size, in this way $10^{20}$ hypercubes are generated. 218 feature vectors after appropriate quantization were stored in these hypercubes.

The performance of three indexing methods discussed above is compared by testing these on features extracted from contours and corners of different orientations of model 7 as shown in Fig. 8. Features extracted from these orientations have not been used in training phase of two

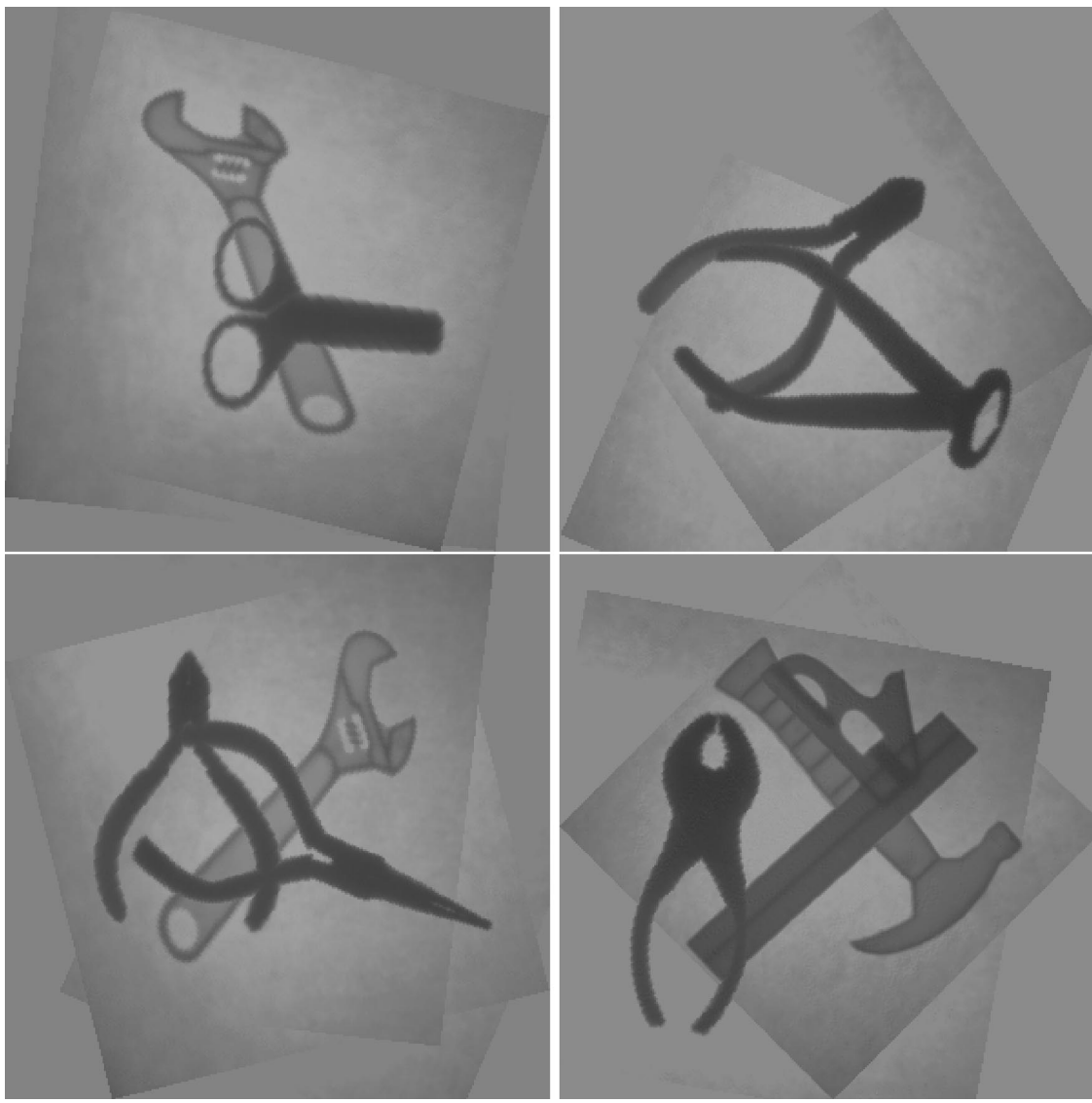

Fig. 9. Images of scenes 1–4.

types of neural networks and also not stored in the hash table. Mapping results for three indexing methods are shown in Table 1. For 11 out of 13 triplets, response was more than 0.5 for correct class using both types of neural networks and also one triplet mapped to neighbouring class in case of Kohonen network. On the other hand, in case of geometric hashing only three out of 13 triplets mapped to correct hypercubes. The reason for failure of the hashing technique is the instability of features due to shift in the position dominant points and noise. In case of neural network-based indexing these problems are taken care by the generalization capability of the network and as a consequence we obtained reasonably correct class indicators. In the hashing method, five triplets have mapped to hypercubes in the neighbourhood of the correct hypercube. We had considered neighbourhood of a hypercube having index $(i_1, \ldots, i_{20})$ as $\{(j_1, \ldots, j_{20}) \mid j_k - 1 \leqslant i_k \leqslant j_k + 1 \text{ for } k = 1, 20\}$. The size of this neighbourhood grows exponentially with the dimension $d$. Hence improving performance of the hashing method by incorporating neighbourhood search would entail exponential complexity.

For a given feature vector, indexing using Kohonen network requires $50 \times 20$ floating point multipications and additions while indexing using two layer feed-forward network requires $50 \times 63$ floating point multiplications and additions. Hashing-based indexing requires 20 divisions. Obviously, computational cost of index calculation in the hashing scheme is less. However, we have to take into account the cost of neighbourhood search (as discussed before) for estimating computational overhead of the hashing-based scheme.

Performance of the hashing-based indexing scheme can be improved by defining variable sized hypercubes using some clustering algorithm. In fact, Kohonen network generates the indexing function by exploiting natural data clusters. Since neighbourhood relation in Kohonen network is defined by the structure of the network this scheme does not require computational overhead of the nearest-neighbour search.

### 6.2. Recognition results

Recognition results of four images of complex scenes consisting of multiple objects (Fig. 9) are given here. Partially occluding silhouette of the objects and control points on the silhouette were extracted using the same procedure as for model objects. Fig. 10 shows contour and control points extracted from these complex images.

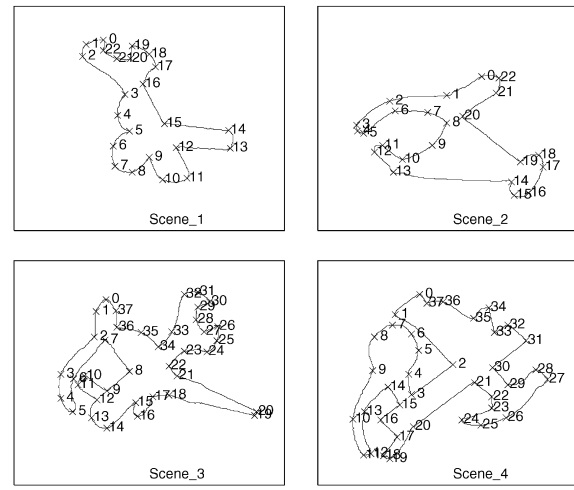


Fig. 10. Contours and corners scenes 1–4.

Table 2
Results for four scenes shown in Fig. 9

| Results | Scene-1 | Scene-2 | Scene-3 | Scene-4 |
|---|---|---|---|---|
| No. of triplets extracted | 23 | 23 | 38 | 38 |
| No. of possible mappings | $23 \times 218$ | $23 \times 218$ | $38 \times 218$ | $38 \times 218$ |
| No. of hypotheses generated using Kohonen networks | 153 | 144 | 193 | 233 |
| No. of hypotheses generated using multi-layer perceptron | 68 | 143 | 203 | 154 |
| No. of composite hypotheses generated in case of Kohonen networks | 26 | 27 | 33 | 51 |
| No. of composite hypotheses generated in case of multi-layer perceptron | 5 | 32 | 38 | 30 |
| No. of hypotheses tested in case of Kohonen networks | 4 | 5 | 3 | 8 |
| No. of hypotheses tested in case of multi-layer perceptron | 1 | 3 | 9 | 4 |
| No. of correct matches in case of Kohonen networks | 2 | 2 | 2 | 3 |
| No. of correct matches in case of multi-layer perceptron | 1 | 2 | 3 | 3 |
| No. of miss errors in case of Kohonen networks | 0 | 0 | 0 | 0 |
| No. of miss errors in case of multi-layer perceptron | 1 | 0 | 0 | 0 |
| No. of substitution errors in case of Kohonen networks | 0 | 0 | 1 | 0 |
| No. of substitution errors in case of multi-layer perceptron | 0 | 0 | 0 | 0 |

Similarity invariant feature vectors extracted from triplets of these images, were given as inputs to both types of neural networks and the output response corresponding to each model triplet was stored in the table. Then using composite hypotheses generation and verification steps recognition results were generated as explained earlier.

Results of recognition steps for four scenes are shown in the Table 2. Number of triplets extracted from four images are given in the first row. Second row shows possible number of object to model triplets mappings. Number of mappings for which response was more than 0.5 after neural networks-based indexing, for both types of networks are given in the third and fourth row of the table. Number of mappings for which response was more than 1.5 after composite hypotheses generation are given in fifth and sixth row. Seventh and eighth rows show the number of hypotheses tested using verification algorithm. Ninth and tenth rows show number of objects correctly recognized. Eleventh and twelveth rows show number of objects not recognized. Number of substitution errors, i.e. object present in the scene mapped to some other object in the model base are given in row 13 and 14.

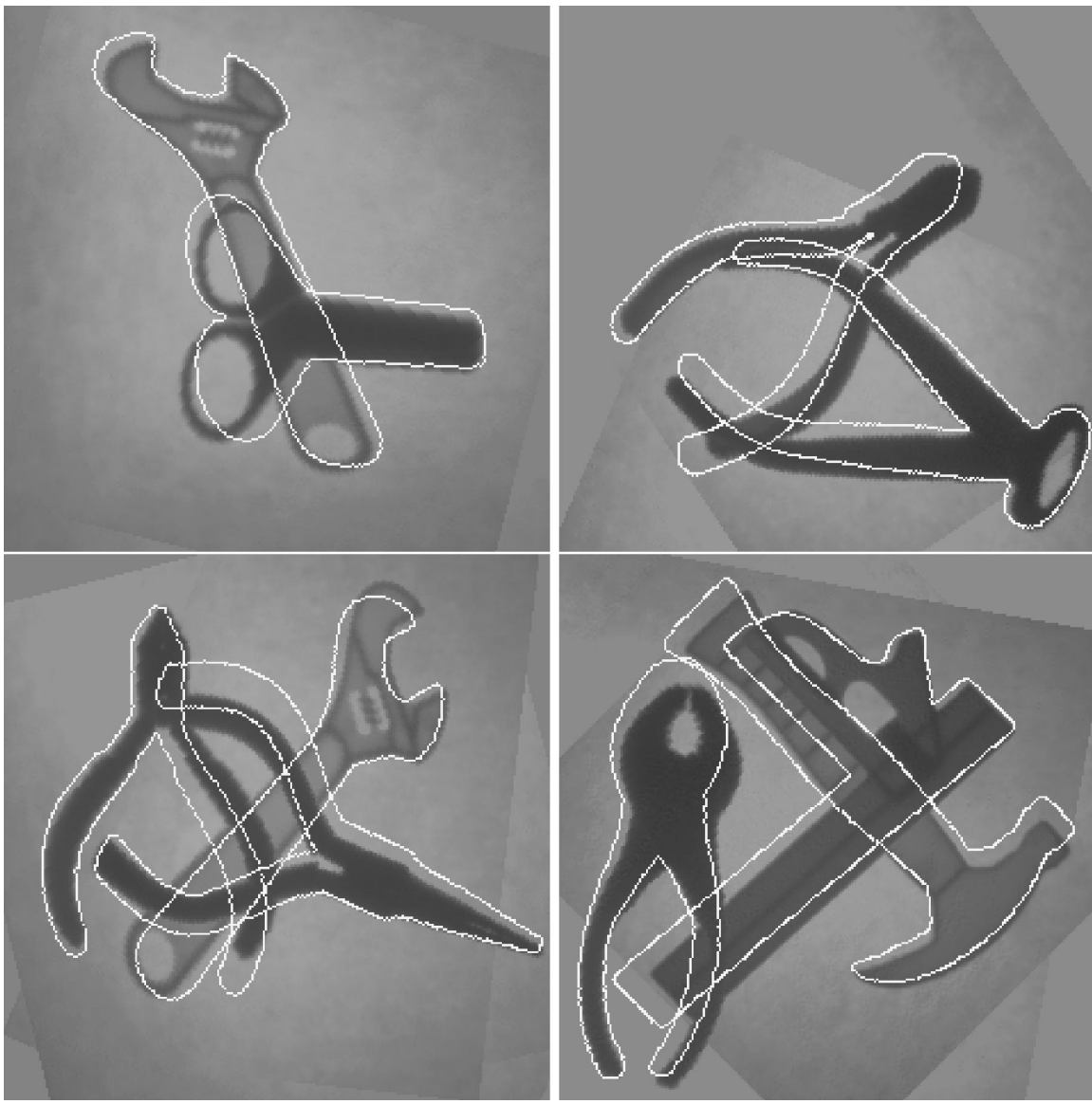

Fig. 11. Matching results of scenes 1–4.

In Scene 1, in case of Kohonen networks, Model 8 was recognized by mapping of scene triplet 19 to triplet 8 of Model 8. Model 11 was recognized by mapping of scene triplet 5 to triplet 2 of Model 11. In case of multi-layer perceptron, only one object (Model 8) in the scene was recognized, in this way, there was miss error i.e Model 11 could not be recognized. There was no substitution error. Model 8 was recognized by mapping of scene triplet 21 to triplet 10 of Model 8. Here out of five composite hypotheses generated, only one was tested because contour corresponding to other hypotheses was removed from the scene after successful verification of first hypothesis.

In Scene 2, in case of Kohonen networks, model 5 was recognized by mapping of scene triplet 0 to triplet 13 of model 5 and model 4 was recognized by mapping of scene triplet 18 to triplet 14 of model 4. Both the objects in the scene were recognized correctly, there was no miss error and substitution error. In case of multi-layer perceptron also, both the objects were recognized by mapping of the same triplets as in the case of Kohonen networks.

In Scene 3, in case of Kohonen networks, Model 6 was recognized by mapping of scene triplet 2 to triplet 12 of Model 6. Model 8 was recognized by mapping of scene triplet 31 to triplet 13 of Model 8. Model 7 could not be recognized in this case, instead of this, Model 14 was picked by mapping of scene triplet 21 to triplet 4 of Model 14. In case of multi-layer perceptron, all the three objects were recognized correctly. Model 7 was recognized by mapping of scene triplet 19 to triplet 9 of Model 7. Model 6 was recognized by mapping of scene triplet 7 to triplet 3 of Model 6. Model 8 was recognized by mapping of scene triplet 28 to triplet 10 of Model 8. There was no miss error and substitution error in this case.

In Scene 4, in case of Kohonen networks, Model 14 was recognized by mapping of scene triplet 24 to triplet 6 of Model 14. Model 3 was recognized by mapping of scene triplet 11 to triplet 1 of Model 3. Model 15 was recognized by mapping of scene triplet 33 to triplet 5 of Model 15. In case of multi-layer perceptron, Model 3 was recognized by mapping of scene triplet 13 to triplet 3 of Model 3. Model 14 was recognized by mapping of scene triplet 25 to triplet 7 of Model 14. Model 15 was recognized by mapping of scene triplet 33 to triplet 5 of Model 15.

Fig. 11 shows mapped model contours on scenes after recognition.

## 7. Conclusion

In this paper, a new neural network-based indexing scheme has been presented. It has been shown that the crucial problem of indexing is that of mapping a particular triplet in the image to a class of triplets defined by model base through an appropriate classification scheme. The indexing has been efficiently implemented using neural networks. Out of two types of neural networks discussed in this paper, it has been observed that indexing scheme using Kohonen network is better compared to multi-layer perceptron model because it does not require manual pre-classification of feature vectors and unsupervised learning is faster. From the recognition results, it is also observed that the error tolerance is more in case of Kohonen network. The complexity of the indexing scheme is expected to show sub-linear growth with increase in the number of model objects because existing feature classes would have the ability to encode contour of new objects. This scheme, therefore, can be applied for model bases containing larger number of objects.

The algorithm suggested for extraction of object contour from grey level images combines advantages of region growing and edge detection and has been tested successfully on various types of images. Neighbourhood constraint criteria used for combining hypotheses generated through indexing, reduces the number of hypotheses to almost one third or even less as shown in the results. Picking of scene-model triplet pairs for verification, using weighted distance transform, in decreasing order of combined response function value leaves very few triplets to be tested. This shows the strength of the neural network-based indexing scheme. Feature extraction, indexing, composite hypothesis generation and verification steps can be used for plane projective transformation cases with slight modifications. The only problem faced in these cases is in extraction of sufficient number of dominant points on the contour.

## References

[1] P.W.M. Tsang, P.C. Yuen, F.K. Lam, Classification of partially occluded objects using 3-point matching and distance transformation, Pattern Recognition 27 (27) (1994) 40.

[2] N. Rajpal, S. Banerjee, R. Bahl, Silhoutte-based object recognition using random associative search, Technical Report 91–2, IIT Delhi, CARE, 1991.

[3] T. Pavlidis, Y.T. Liow, Integrating region growing and edge detection, IEEE Trans. Patt. Anal. Mach. Intell. 12 (225) (1990) 233.

[4] M.H. Han, D. Jang, The use of maximum curvature points for the recognition of partially occluded objects, Pattern Recognition, 23 (21) (1990) 33.

[5] J.L. Turney, T.N. Mudge, R.A. Volz, Recognizing partially occluded parts, IEEE Trans. Patt. Anal. Mach. Intell. 7 (410) (1985) 421.

[6] H.C. Liu, M.D. Srinath, Partial shape classification using contour matching in distance transformation, IEEE Trans. Patt. Anal. Mach. Intell. 12 (1072) (1990) 1079.

[7] Y. Lamdan, J. Schwartz, H. Wolfson, Affine invariant model-based object recognition. IEEE Trans. Robotics Automat. 6 (578) (1990) 579.

[8] A. Zisserman, D.A. Forsyth, J.L. Mundy, C. Rothwell, Recognizing general curved objects efficiently, in: Geometric

Invariance in Computer Vision, MIT Press, Cambridge, MA, 1992, pp. 232–254.

[9] K.L. Clarkson, An algorithm for approximate closest-point queries, in: Proc. 10th Computational Geometry 94–6/94, Stony Brook, NY, USA, 1994, pp. 160–164.

[10] V. Cherkassky, J. H. Friedman, H. Wechsler, From Statistics to Neural Networks Theory and Pattern Recognition Applications, Springer, Berlin, 1994.

[11] R.P. Lippman, Neural networks, bayesian a posteriori probabilities, and pattern classification, in: From Statistics to Neural Networks Theory and Pattern Recognition Applications, Springer, Berlin, 1994, pp. 83–104.

[12] J. Hertz, A. Krog, R.G. Palmer, Introduction to Theory of Neural Computation, Addison-Wesley, Reading, MA, 1991.

[13] Y.H. Pao, Adaptive Pattern Recognition and Neural Networks, Addison-Wesley, Reading, MA, 1989.

[14] E. Rivlin, I. Weiss, Local invariant for recognition, IEEE Trans, Pattern Anal. Mach. Intell. 17 (226) (1995) 238.

[15] T. Pavlidis, Algorithms for Graphics and Image Processing, Springer, Berlin, 1982.

[16] D.M. Wuescher, K.L. Boyer, Robust contour decomposition using a constant curvature criterion, IEEE Trans. Pattern Anal. Mach. Intell. 13 (41) (1991) 51.