Proceedings of the IASTED International Conference
Signal and Image Processing
October 27-31, 1998, Las Vegas, Nevada - USA

Recognition Using Curvature Scale Space and Artificial Neural Networks

George Bebis[1], George Papadourakis[2] and Stelios Orphanoudakis[3]

Department of Computer Science, University of Nevada, Reno, USA[1]
Department of Science, Technological Educational Institute, Heraklion, Crete, Greece[2]
Department of Computer Science, University of Crete, Heraklion, Crete, Greece[3]

Abstract

We address the problem of recognizing real flat objects from two-dimensional images. A new method is proposed which has mainly been designed to handle complex objects and performs under occlusion and similarity transformations. Matching operates hierarchically, guided by a Curvature Scale Space (CSS) segmentation scheme, and takes advantage of important object features first, that is, features which distinguish an object from other objects. The model database is implemented using a set of Artificial Neural Networks (ANNs) which provide the essential mechanism not only for establishing correct associations between groups of segments and models but also for enabling efficient searching and robust retrieval, especially when noisy or distorted objects are considered.

1. Introduction

Object recognition is an essential part of any high-level computer vision system. The most successful approach is probably in the context of *model-based* object recognition [1], where the environment is rather constrained and recognition relies upon the existence of a set of predefined model objects. There are two separate phases of operation: *training* and *recognition*. During the training phase, a model database is built by establishing proper associations between features and models. During the recognition phase, scene features are used to retrieve appropriate associations stored in the model database. Indexing is a popular approach which has received a lot of attention lately [2]-[6]. It is based on the idea of using *a-priori* stored information about the models in order to quickly eliminate non-compatible matches during recognition. Efficient indexing schemes are used to store and retrieve information to and from a table.

In this paper, a new indexing-based object recognition method is proposed. It is assumed that the object are described by their boundaries which do not cross over themselves and have a number of "interest" points. Objects appearing in the scene may have undergone similarity transformations, that is, rotation, translation and scaling. The method has the following key characteristics: *First*, emphasis is given on the detection and utilization of important object features, that is, features which can distinguish a model object from other model objects. These features are obtained by employing a multiscale segmentation scheme based on the Curvature Scale Space (RCSS) of the object contour [7]. *Second*, matching operates hierarchically, guided by the multiscale segmentation scheme, and takes advantage of important object differences first. *Finally*, the organization of the model database is based on Artificial Neural Networks (ANNs) which provide a mechanism for efficient storage and robust retrieval [8].

2. Motivations

An important issue in the implementation of a model based object recognition system is how to extract important object features and how to utilize them efficiently during matching. A common approach is to split the object contour into a number of segments and then use them for matching [2]-[6]. However, because a large number of segments is usually obtained, which are usually too local to be appropriate for matching, recognition can become quite slow. Several methods [5][6] try to alleviate this problem by grouping together a number of adjacent segments in order to create more descriptive segments. This approach, however, requires the *a-priori* choice of a parameter which determines the number of segments to be grouped together, a parameter which is rather data dependent. Our approach for obtaining descriptive segments is based on the use of a multiscale segmentation scheme and does not require such a parameter. Segments located at relatively high scales are usually very descriptive and distinct. This makes them quite attractive for matching. Moreover, they are often very long which allow us to localize the objects in the scene more accurately. Unfortunately, their main drawback is that they are very sensitive to occlusions. On the other hand, segments detected at lower scales are less descriptive and distinct, but are also more resistant to occlusions since they are shorter. The approach we have taken is to consider both kinds of segments by using a hierarchic matching procedure, driven by the multiscale segmentation scheme.

Another important issue is how to organize and search the model database. Most of the current approaches use efficient indexing schemes in order to store and retrieve information to and from the database. Hashing is a popular indexing scheme. In general, it is important for the hash function to distribute the data randomly over the table. In object recognition, however, it is important for the hash function to be proximity preserving, that is, to map similar data to close by table locations. In addition, it should be noise tolerant. Finding hash functions which satisfy these properties is not an easy task. The multilayer ANN poses a number of properties which make it an attractive choice. Implementing the model database using ANNs requires that appropriate associations between features and models have first been established. Then, the mapping (hash function) required to implement these associations can be found through training. Assuming that the hash keys are unique, the computed hash function will be collision free. Also, some noise in the data can be handled since ANNs demonstrate good tolerance to noise. Finally, proximity can be enforced by choosing a proximity preserving representation scheme for the outputs of the ANNs.

3. Curvature scale space

The curvature scale space (CSS) approach was introduced in [7] as a shape representation for planar curves. Object boundaries are usually represented as planar curves that do not cross over themselves. This method is based on finding points of inflection (i.e, curvature zero-crossings) on the curve at varying levels of detail. The curvature k of a planar curve, at a point on the curve, is defined as the instantaneous rate of change of the slope of the tangent at that point with respect to arc length, and it can be expressed as follows:

$$k(t) = \frac{\dot{x}(t)\ddot{y}(t) - \dot{y}(t)\ddot{x}(t)}{(\dot{x}(t)^2 + \dot{y}(t)^2)^{3/2}},$$

where $\dot{x}(t)$, $\ddot{x}(t)$, $\dot{y}(t)$, and $\ddot{y}(t)$ are the first and second derivatives of $x(t)$ and $y(t)$ respectively, and $(x(t), y(t))$ is the parametric representation of the curve.

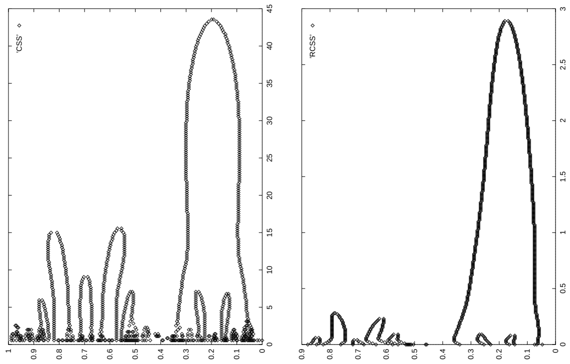

Figure 1. (a) The CSS, (b) the RCSS.

In order to compute the curvature of the curve at varying levels of detail, $x(t)$ and $y(t)$ are convolved with a Gaussian function g(t,σ). Defining $X(t,\sigma)$ and $Y(t,\sigma)$ as the convolution of $x(t)$ and $y(t)$ respectively with the Gaussian function, the smoothed curve curvature $k_\sigma(t)$ can be expressed as follows:

$$k(t,\sigma) = \frac{X_t(t,\sigma)Y_{tt}(t,\sigma) - X_{tt}(t,\sigma)Y_t(t,\sigma)}{(X_t(t,\sigma)^2 + Y_t(t,\sigma)^2)^{3/2}},$$

where $X_t(t,\sigma)$ and $X_{tt}(t,\sigma)$ correspond to the first and second derivatives of $x(t)$. $Y_t(t,\sigma)$ and $Y_{tt}(t,\sigma)$ are defined in a similar manner. The function defined implicitly by $k(t,\sigma) = 0$ is the CSS image of the curve. To demonstrate the CSS approach, the object contour shown in Figure 1(a) was convolved with a Gaussian filter, using successively doubled values of σ. The locations at which $k = 0$ are marked on each curve. The CSS representation of the an object contour (Figure 3) is shown in Figure 1(left). In this paper, we use a variation of the CSS which is called the Resampled Curvature Scale Space (RCSS) representation. The RCSS was introduced in [9], and was shown to be more suitable when there is a high-intensity non-uniform noise or local shape differences exist (see [9][10] for some examples). Also, it can be computed for open curves. More details as well as a simple way to compute RCSS can be found in [9]. Figure 1(right) shows the RCSS of the same object contour.

4. The preprocessing step

The role of the preprocessing step is to compute a hierarchy of segmentations for each model contour and to encode the contour segments. Figure 2 illustrates the steps.

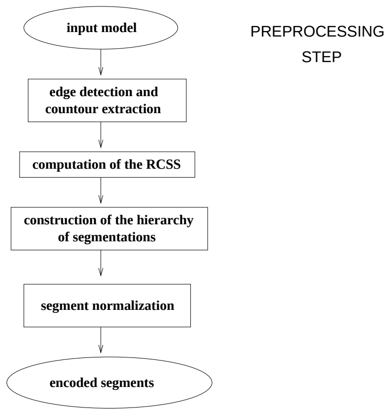

Figure 2. The preprocessing stage

4.1. The hierarchy of contour segmentations

First,the RCSS image must be computed. Then, the hierarchy of segmentations is obtained by concentrating at each scale separately and breaking the original contour at points which correspond to the curvature zero-crossing points detected at this scale. The reason we use curvature zero-crossings is because they are small in number and robust to viewpoint changes. However, curvature extrema can also be used. Initially, the RCSS of the object contour is computed by convolving the object contour repeatedly, until no curvature zero-crossing points exist any more. Then, the coarsest apex points (i.e., maxima of the RCSS contours) are detected. Apex points introduce new pairs of zero-crossings which can be used to split the object contour. To determine the endpoints of the segments accurately, the zero-crossings are localized down to the original scale or to a slightly higher scale for noise tolerance). Figure 3 shows segmentation results at different scales and localization down to the original scale.

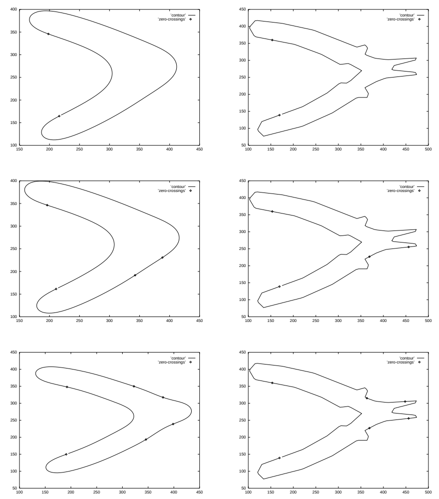

Figure 3. Segmentation and localization.

4.2. Encoding the contour segments

To encode the segments, the first four moments of their centroidal profile is used [11][12]:

$$(a)\ \frac{\frac{1}{N}[\sum_{i=1}^{N}[d(i)-m_1]^2]^{1/2}}{m_1} \qquad (b)\ \frac{\frac{1}{N}\sum_{i=1}^{N}[d(i)-m_1]^3}{\frac{1}{N}[\sum_{i=1}^{N}[d(i)-m_1]^2]^{3/2}}$$

$$(c)\ \frac{\frac{1}{N}\sum_{i=1}^{N}[d(i)-m_1]^4}{\frac{1}{N}[\sum_{i=1}^{N}[d(i)-m_1]^2]^2} \qquad (d)\ \frac{\frac{1}{N}\sum_{i=1}^{N}[d(i)-m_1]^5}{\frac{1}{N}[\sum_{i=1}^{N}[d(i)-m_1]^2]^{5/2}}$$

where $m_1 = 1/N\sum_{i=1}^{N} d(i)$ and *d(i)* is the centroidal profile representation of a segment with *N* points, *i=1,2, ..., N*. The advantages of using moments to represent the centroidal profile of a segment are: (i) they are invariant to the starting point used to compute the centroidal profile, (ii) they represent the centroidal profile economically, and (iii) they are quite robust to noise and distortions.

5. The training phase

During the training phase, the model database is built. The model database has been partitioned into two parts. The first part is implemented using a single ANN called, *segment classifier*. The goal of the segment classifier is to assign segments extracted from the hierarchy of segmentations into appropriate classes. The second part is implemented using a set of hierarchically structured ANNs called, *object recognizers*. Their goal is to recognize which models are present in the scene. Each object recognizer uses information about groups of segments which have been previously classified by the segment classifier.

5.1. The segment classifier

The purpose of the segment classifier is to assign encoded segments to appropriate segment classes. The way segment classes are determined is by applying a clustering procedure to the entire set of encoded segments. The clustering algorithm used is a variation of a simple cluster seeking algorithm (see [10]). The segment classifier is implemented by a multilayer ANN. Figure 4 shows the steps involved in the training of the segment classifier.

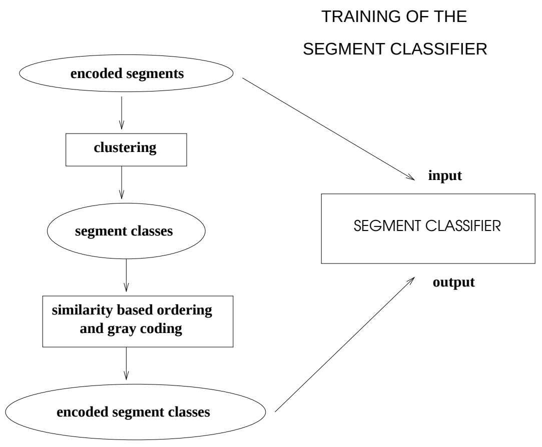

Figure 4. The training of the segment classifier.

5.2. Training the segment classifier

The training of the segment classifier is performed using training examples of the form *(segment, segment class)*. The segment classes are determined using a simple cluster seeking algorithm (see [10]). The inputs of the network are simply the four moments of the segment's centroidal profile. The outputs of the network must be an appropriate representation of the segment class. Choosing a proper encoding scheme for representing the outputs of the segment classifier is very important. In our case, the scheme should satisfy two requirements: first, it should be economical and second, it should be proximate. The proximity constraint assures that elements of this scheme, representing similar classes, will be similar. This is very desirable for increasing the robustness of the object recognizers which use the outputs of the segment classifier in their decisions. An appropriate scheme, satisfying both of the requirements, is the Gray code scheme.

5.3. The object recognizers

The purpose of the object recognizers is to perform recognition based on segment classifications made by the segment classifier. Each object recognizer is trained to learn associations between important groups of segments and models. For each model, important groups of segments are extracted by applying a *selective search* procedure on the hierarchy of contour segmentations associated with the model. The segments obtained are classified by the segment classifier and the obtained segment classifications are used to form the inputs to the object recognizers. Each object recognizer has different capabilities which depend on the size of groups (e.g., number of segments in the group) it utilizes for recognition. Figure 5 shows the steps involved in the training of the object recognizers.

5.4. Finding important groups of segments

The objective of the selective search procedure is to identify groups of segments having high discrimination power (e.g., they can distinguish a model from other models). These groups are referred to as *important groups*. First, the selective procedure looks for important groups of size one, that is, single segments which characterize a model uniquely. Such segments are usually found at the higher levels of the segmentation hierarchy. Initially, one of the hierarchies of contour segmentations is randomly chosen and designated as the "reference hierarchy". Then, we concentrate at each level of the reference hierarchy, starting from the highest possible level, comparing segments at this level with segments coming from any level of the other hierarchies. Then, the procedure continues at the next lower level of the reference hierarchy, until all the levels have been considered. When all the levels of the reference hierarchy have been processed, one of the remaining hierarchies is randomly chosen and becomes a new reference hierarchy. Then, the same procedure is repeated until all the hierarchies have been treated as a reference hierarchy. Except from identifying important segments, the selective search procedure identifies important groups of segments as well. An important group of segments consists of a number of adjacent segments which may or may not be important by themselves, however, their identity along with their order in the group makes them important (i.e., they can characterize a model uniquely). In our implementation, the search procedure extracts important groups of size two, three, and four.

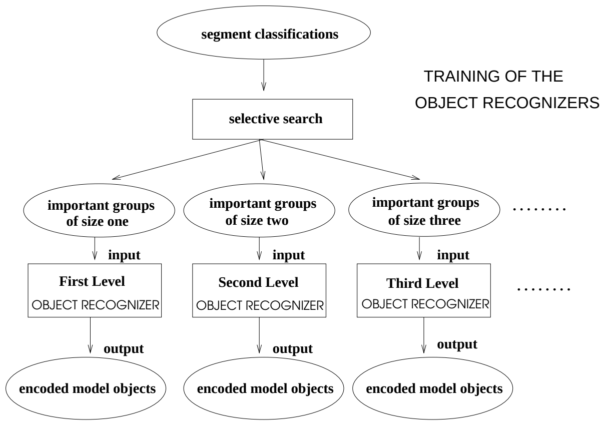
segment classifications
TRAINING OF THE OBJECT RECOGNIZERS
selective search
important groups of size one
important groups of size two
important groups of size three
....
input
input
input
First Level OBJECT RECOGNIZER
Second Level OBJECT RECOGNIZER
Third Level OBJECT RECOGNIZER
....
output
output
output
encoded model objects
encoded model objects
encoded model objects

Figure 5. The training of the object recognizers.

5.5. Training the object recognizers

During training, the object recognizers learn to associate important groups of segments with the models uniquely characterized by these groups. The object recognizers have been structured hierarchically, according to the size of groups they utilize for recognition. The object recognizer located at the top of the hierarchy is trained using important segments only. The object recognizer located at the next level is trained using important groups of size two. In a similar fashion, the object recognizer located at the bottom the hierarchy is trained to recognize model objects using important groups of a maximum size (four in our case).

The inputs to the object recognizers are simply the encoded segment classifications associated with the important groups. The output of the object recognizers is an appropriate representation of the model characterized by the important group of segments. The local coding scheme has been used. According to this scheme, the number of output nodes is equal to the number of model objects, with each node representing a different model. This is very suitable for our task because it allows us to distinguish between *ambiguous* and *unambiguous* decisions. In the case of an unambiguous recognition, the output of an object recognizer contains only one 1 at a certain location and 0's everywhere else. Outputs with more that one 1's are considered to be ambiguous because they do not correspond to a particular model object representation. Ambiguous recognition results are rejected immediately while unambiguous recognition results are considered for further verification. It should be mentioned that the training set of an object recognizer includes two kinds of examples: *positive* and *negative*. The negative training examples correspond to non-important groups of segments, that is, groups that do not characterize any model uniquely. Negative examples are important so that the networks respond in a predictable way when a decision can not be made.

6. Recognition and verification

Initially, the preprocessing step is applied to every contour extracted from the scene. Matching is performed from higher to lower scales and in a local to global basis. First, recognition is attempted using segments located at a high scale. Each segment is classified by the segment classifier and the obtained encoded segment classifications are fed, one at a time, to the top level object recognizer. If this object recognizer fails to yield an unambiguous recognition, pairs of encoded segment classifications, corresponding to adjacent segments, are fed, one at a time, to the next level object recognizer. If this object recognizer is also unable to perform an unambiguous recognition, encoded segment classifications corresponding to triplets of adjacent segments are chosen and fed to the object recognizer located at the next lower level. This procedure continues until all the levels of the hierarchy of segmentations have been considered or all the objects present in the scene have been recognized. In case of an unambiguous recognition, verification takes place to evaluate its validity. During verification, we find model-scene groups which are in correspondence (see [10]), and we compute a transformation which aligns them. This transformation is then used to back-project the model onto the scene.

7. Simulations

The performance of the recognition system was tested using a set of 20 real objects (see [10]). A Laplacian edge detector separated the objects from the background and a boundary following routine extracted their boundaries. All boundaries were approximated by 256 contour pixels.

7.1. Applying the preprocessing step

Once the boundaries of the object models were extracted, their RCSS images were computed. Thus, each contour was convolved with a Gaussian filter for a continuous range of scales. The standard deviation σ of the Gaussian was set equal to 1 and the filter's kernel size was chosen to be 5σ. Then, we computed the hierarchy of segmentations for each model, as described in section 5.1. The total number of segments obtained was 351 (segments which appeared at multiple levels were counted only once).

7.2. Training the segment classifier

First, the segment clusters were found, then they were ordered based on their similarity and finally, they were encoded using gray codes. The total number of clusters found was 159 which required 8-bits to encode them. This determined also the number of output nodes of the segment classifier (8 nodes). The network used was a two layer network (one hidden layer and one output layer), with 80 nodes in the hidden layer. The back-propagation algorithm with momentum was used for training.

7.3. Training of the object recognizers

Two layer ANNs were used for implementing the object recognizers. The object recognizer located at the top of the hierarchy was trained to recognize model objects using important segments. Thus, the number of input nodes for this network was set to 8. The number of positive training examples used to train this object recognizer was 107 while the number of negative examples was 14. The ANN located at the next lower level of the hierarchy was trained to recognize model objects using important groups of size two. Thus, the number of its input nodes was set to 16. Its training set consisted of 150 positive training examples and 8 negative training examples. In a similar manner, the number of input nodes and training examples of the rest networks were determined (see [10]). For each network, the number of output nodes was set to 20. Back-propagation with momentum was used for training.

7.4. Experiments and results

First, we performed experiments to test the robustness of the method in presence of noise and distortion. No occlusion was allowed at this time. For each model object, 100 test objects were generated. These test objects consisted of samples of different sizes of the model objects in various translational and rotational positions, with noise and distortion. The objects were rotated over the range from 0 to 2π radians and translated to random positions. The size of each object was varied from 0.5 to 1.5 times the size of the original object. Noise and distortion effects were introduced by adding random noise to the boundary points using the approach of You and Jain [13]. The accuracy of the proposed was tested using objects corrupted with various amounts of noise (0% to 50% noise corruption). Figure 6 shows a recognition example.

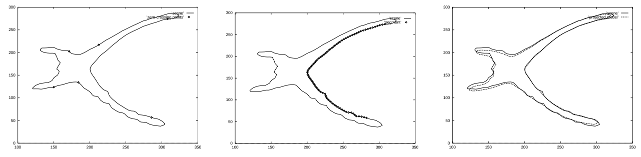

Figure 6. Recognition results (proposed method)

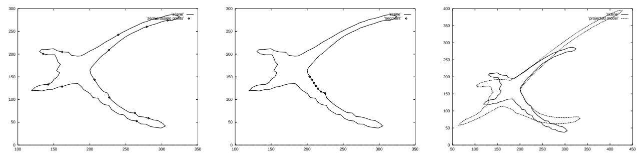

Figure 7. Recognition results (variation)

For comparison purposes, we also implemented a variation of [3]. In [3], the segments are represented by the Fourier Descriptors of the object's contour segments. Here, we are using the moments of the centroidal profiles of the segments, as is the case with our approach. Also, the procedure for extracting the contour segments in [3] is different than the one used here (the segments are determined by the curvature zero-crossings of the object's contour using a Gaussian with σ=1). The variation yields much less segments compared to the original approach ([3]) since we do not consider overlapping windows. However, our objective here is to demonstrate the problems associated with traditional indexing schemes when noise and distortion are present. Also, we want to demonstrate the advantages of using segments from higher scales first for better recognition and localization. Figure 7 shows a recognition example in the case of the variation. Both methods were able to recognize the instance of this object, however, the proposed method has localized the object more accurately. In addition, the proposed method verified only 2 hypotheses while the variation verified 46. Figure 8 (left) shows the overall recognition accuracy of the two methods (the solid line corresponds to our method), assuming various levels of noise (0% to 50%). The average number of hypotheses verified by the two methods are also shown in Figure 8 (right).

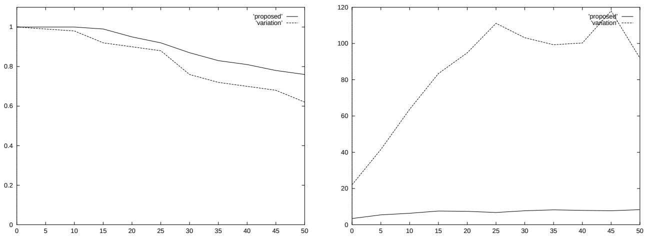

Figure 8. Comparison of the methods.

Next, we tested the proposed approach using objects corrupted by occlusion. Occlusion was introduced in a random way. The amount of occlusion in an object contour was determined by the percentage of absent contour points. The test objects used in this experiment were also rotated, translated, and scaled. Our results show that the recognition accuracy and number of hypotheses were not significantly different with up to 50% occlusion. With more than 50% occlusion, however, we observed that the recognition accuracy started to deteriorate rather fast and the number of hypotheses to increase considerably.

Finally, we tested the proposed approach using a number of real scenes containing overlapping objects. Figure 9 shows one of the scenes used in our experiments. The figure shows the segments chosen by the method during recognition. As can be observed, the method chose some composite segments first yielding unsuccessful recognition. In most of these cases, ambiguous recognition results were reported by the object classifiers while in some other cases, verification required to reject these matches. The total number of hypotheses tested for the recognition of both models was 8.

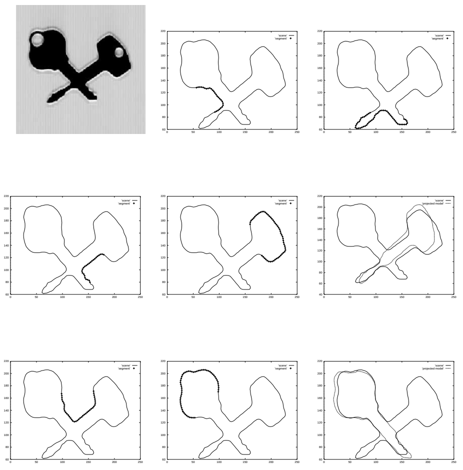

Figure 9. Recognition results.

8. Conclusions

We have presented a new method for the recognition of two dimensional objects from intensity images. The method is based on a hierarchical matching approach and uses ANNs to organize the model database. The proposed method has been tested using both artificial and real data illustrating good performance. Some comments that can be made are the following: first, although the hierarchical segmentation scheme is quite effective, it is also quite time consuming. In the current implementation, we have not tried to optimize it. However, faster implementations are possible using a number of heuristics [7]. Second, maintaining the model database is quite important. Assuming that the model database needs to be expanded by adding new models, our present implementation requires retraining all the ANNs. This is quite inefficient. However, there exist ANNs which have the ability to store new knowledge without significantly destroying previous one. This problem is known as the "stability plasticity" problem and some architectures which can deal with this problem are the Fuzzy-ARTMAP [14], and Cascade Correlation [15] architectures.

Acknowledgement: This work was supported by a grant from the Research Advisory Board at the University of Nevada, Reno (JFA).

References

[1] R. Chin and C. Dyer, "Model-based recognition in robot vision", *Computing Surveys*, vol. 18, no. 1, pp. 67-108, 1986.

[2] A. Kalvin, E. Schonberg, J. Schwartz and M. Sharir, "Two dimensional model based boundary matching using footprints", *International Journal of Robotics Research*, vol. 5, no. 4, pp. 38-55, 1986.

[3] T. Knoll and R. Jain, "Recognizing partially visible objects using feature indexed hypotheses", *IEEE Journal of Robotics and Automation*, vol. RA-2, pp. 3-13, 1986.

[4] R. Mehrotra and W. Grosky, "Shape matching utilizing indexed hypotheses generation and testing", *IEEE Trans. on Robotics and Automation*, vol. 5, no. 1, 1989.

[5] F. Stein and G. Medioni, "Structural indexing: efficient 2-D object recognition", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 12, pp. 1198-1204, 1992.

[6] I. Sethi and N. Ramesh, "Local association based recognition of two-dimensional objects", Machine Vision and Applications, vol. 5, pp. 265-276, 1992.

[7] F. Mokhtarian and A. Mackworth, "Scale-based description and recognition of planar curves and two-dimensional shapes", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 8, no. 1, pp. 34-43, 1986.

[8] J. Hertz, A. Krogh and R. Palmer, "Introduction to the theory of neural computation", *Addison Wesley*, 1991.

[9] F. Mokhtarian and A. Mackworth, "A theory of multiscale, curvature-based shape representation for planar curves", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 14, no. 8, pp. 789-805, 1992.

[10] G. Bebis, G. Papadourakis, and S. Orphanoudakis, "Curvature scale space driven object recognition with an indexing scheme based on artificial neural networks", accepted *Pattern Recognition* (also available from http://www.cs.unr.edu/~bebis).

[11] H. Freeman, "Shape Description via the use of Critical points", *Pattern Recognition*, vol. 10, pp. 159-166, 1978.

[12] L. Gupta and M. Srinath, "Contour sequence moments for the classification of closed planar shapes", *Pattern Recognition*, vol. 20, no. 3, pp. 267-272, 1987.

[13] Z. You and A. Jain, "Performance evaluation of shape matching via chord length distribution", *Computer Vision, Graphics and Image Processing*, vol. 28, pp. 129-142, 1984.

[14] G. Carpenter, S. Grossberg, N. Markuzon, J. Reynolds and B. Rosen, "Fuzzy ARTMAP: A neural network architecture for incremental supervised learning of analog multidimensional maps", *IEEE Transactions on Neural Networks*, vol. 3, no. 5, pp. 698-713, 1992.

[15] S. Fahlman and C. Lebiere, "The cascade-correlation learning architecture", *In Advances in Neural Information Processing Systems II*, D. Touretzky (Ed.), pp. 524-532, San Mateo: Morgan Kaufmann, 1990.