Recognition of Leaf Images Based on Shape Features Using a Hypersphere Classifier*

Xiao-Feng Wang[1,2], Ji-Xiang Du[1], and Guo-Jun Zhang[1]

[1] Institute of Intelligent Machines, Chinese Academy of Sciences, Hefei, 230031
[2] Department of Computer Science and Technology, Hefei University, Hefei, 230022
xfwang@iim.ac.cn

Abstract. Recognizing plant leaves has so far been an important and difficult task. This paper introduces a method of recognizing leaf images based on shape features using a hypersphere classifier. Firstly, we apply image segmentation to the leaf images. Then we extract eight geometric features including rectangularity, circularity, eccentricity, etc, and seven moment invariants for classification. Finally we propose using a moving center hypersphere classifier to address these shape features. As a result there are more than 20 classes of plant leaves successfully classified. The average correct recognition rate is up to 92.2 percent.

1 Introduction

Plant is one of the most important forms of life on earth. Plants maintain the balance of oxygen and carbon dioxide of earth's atmosphere. The relations between plants and human beings are also very close. In addition, plants are important means of livelihood and production of human beings. But in recent years people have been seriously destroying the natural environments, so that many plants constantly die and even die out every year. Fortunately, people now have realized that is a terrible mistake and are beginning to take steps to protect plants.

The first step of protecting plants is to automatically recognize or classify them, i.e., understand what they are and where they come from. But it is very difficult for ones to recognize a plant in hand correctly and immediately because there are so many kinds of plants unknown to us on earth. So we wish to use image processing and pattern recognition techniques to make up the deficiency of our recognition ability. This point can be performed just through computers and other image acquiring facilities. According to theory of plant taxonomy, it can be inferred that plant leaves are most useful and direct basis for distinguishing a plant from the others, and moreover, leaves can be very easily found and collected everywhere. By computing some efficient features of leaves and using a suitable pattern classifier it is possible for us to recognize different plants quickly.

Some recent work has focused on leaf feature extraction for recognition of plant. Im et al. [4] used a hierarchical polygon approximation representation of leaf shape to recognize the Acer family variety. Wang et al. [5] gave a method which combines different features based on centroid-contour distance curve, and

* This work was supported by the NSF of China (Nos.60472111 and 60405002).

D.S. Huang, X.-P. Zhang, G.-B. Huang (Eds.): ICIC 2005, Part I, LNCS 3644, pp. 87–96, 2005.

adopted fuzzy integral for leaf image retrieval. Moreover, Saitoh et al. [6] required two images, a frontal flower image and a leaf image to recognize the plant.

In this paper we propose a method of recognizing leaf images based on shape features using a hypershphere classifier. 15 features are extracted from pre-processed leaf images, which include eight ratios of geometric features and seven Hu moment invariants. In addition, a moving center hypersphere (MCH) classifier is proposed for classifying a large number of leaves.

This paper is organized as follows: Section 2 introduces the leaf image segmentation and feature extraction methods. The moving center hypersphere classifier is described in Section 3. Section 4 gives some experimental results about the performances of the hypersphere classifier by some practical plant leaves. Finally, conclusions are in section 5.

2 Image Segmentation and Feature Extraction

2.1 Image Segmentation

The purpose of image segmentation is to separate leaf objects from background so that we can extract leaves' shape features exactly in the later procedures, and the output of image segmentation is a binary image in which the leaf objects are numerically displayed with 1 and the background is with 0. There are two kinds of background in the leaf images that we have collected, one is simple (as shown in Fig.1-a), another kind is complicated (as shown in Fig.1-d). In this paper we choose iterative threshold selection segmentation method to address leaf images with simple background. Marker-controlled watershed segmentation method is selected for those leaf images with complicated background.

(1) Iterative Threshold Selection Segmentation: It can be seen in the leaf images with simple background that the gray level of pixels within leaf objects is distinctly different from that of pixels within the background. In this case, there are some distinct peaks corresponding to leaf objects and background in the gray level histogram. Therefore, we could find a threshold to transform a gray leaf image into a binary leaf image. Here we use the iterative threshold selection segmentation method [7] to compute the threshold, the method is summarized as follows:

Step 1. Compute the maximum value G_{max} and the minimum value G_{min} of all gray level in gray leaf image. Then set threshold $T_k = (G_{max} + G_{min})/2, k = 0$.

Step 2. At iterative step k , segment image into objects and background using threshold T_k . Compute G_O and G_B as the mean gray level of objects and background.

Step 3. Set new threshold $T_{k+1} = (G_O + G_B)/2$.

Step 4. If $T_{k+1} = T_k$, output threshold $T = T_k$, iteration stop; otherwise,$k = k + 1$, return to step 2.

(2) Marker-Controlled Watershed Segmentation: In the leaf images with complicated background we could find that target leaves are touching or covering some background leaves. It's difficult to separate out the target leaves from the

background using the traditional thresholding methods. The watershed segmentation is a popular segmentation method coming from the field of mathematical morphology, which can separate touching objects in an image. So we consider using marker-controlled watershed segmentation method [8] to get the target leaf objects within the complicated background, this procedure can be summarized as follows:

Step 1. Transform a leaf image into a gray image and compute its gradient image.
Step 2. Mark the target leaf objects and background.
Step 3. Applying watershed segmentation to the gradient image.

After segmentation we can locate the leaf objects in binary images. Notice that there exist some variance on length and curvature of leafstalks. To keep the precision of shape features extraction these leafstalks should be removed. Therefore, we consider applying opening operation of mathematical morphology to binary images, which is defined as an erosion operation followed by a dilation operation using a same structuring element. By performing opening operation several times, we can successfully remove the leafstalks while preserving the main shape characteristics of leaf objects. The results of segmentation and removing leafstalks for leaf images with simple and complicated background are illustrated in Fig.1 (a) - Fig.1 (g).

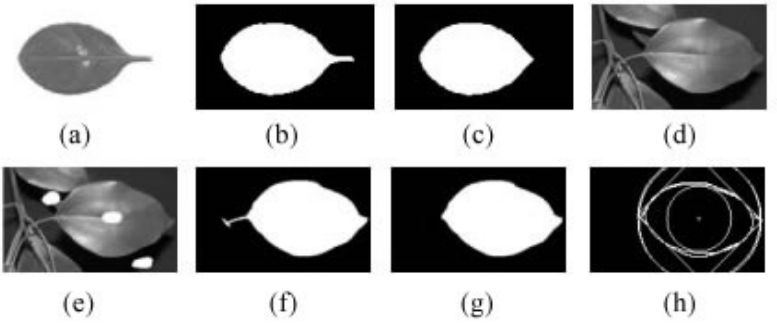

Fig. 1. (a) Leaf image with simple background. (b) Iterative threshold selection segmentation result. (c) Removing leafstalk result. (d) Leaf image with complicated background. (e) Markers of target leaf and background. (f) Watershed segmentation result. (g) Removing leafstalk result. (h) Some shape descriptors used to compute eight geometric features of Fig.1 (g).

2.2 Features Extraction

There are many kinds of image features such as shape features, color features and texture features, etc, which can be used for leaf images classification. According to theory of plant taxonomy, it can be inferred that the shape features are the

most important and effective ones. Consequently, we consider using shape features for classification. Notice that there exist greater morphological differences in different kinds of leaves; even in the same kind of leaves there also exist some variance on scale. In this case, we should use ratios instead of those variable values such as area and perimeter, etc.

From the binary image we could get several shape descriptors that include bounding box (a rectangle that circumscribes a object), convex hull (the smallest convex shape that contains the object), inscribed circle (the largest possible circle that can be drawn interior to the object), circumscribed circle (a circle that passes through all vertices of a object and contains the entire object in its interior), centroid and contour. Fig.1 (h) illustrates these shape descriptors obtained from Fig.1 (g). Using these shape descriptors we could further compute eight ratios of geometric features, which include:

$$Aspect\ Ratio = \frac{length_{bounding\ box}}{width_{bounding\ box}} \ . \tag{1}$$

$$Rectangularity = \frac{Area_{object}}{Area_{bounding\ box}} \ . \tag{2}$$

$$Area\ Convexity = \frac{Area_{object}}{Area_{convex\ hull}} \ . \tag{3}$$

$$Perimeter\ Convexity = \frac{Perimeter_{object}}{Perimeter_{convex\ hull}} \ . \tag{4}$$

$$Sphericity = \frac{R_{inscribed\ circle}}{R_{circumscribed\ circle}} \ . \tag{5}$$

$$Circularity = \frac{4\pi \times Area_{object}}{Perimeter^2_{convex\ hull}} \ . \tag{6}$$

$$Eccentricity = \frac{Axis\ Length_{long}}{Axis\ Length_{short}} \ . \tag{7}$$

$$Form\ Factor = \frac{4\pi \times Area_{object}}{Perimeter^2_{object}} \ . \tag{8}$$

These ratios are naturally invariant to translation, rotation and scaling; it's very important and useful to leaf images classification. Besides the geometric features, the moments are also widely used as shape features for image processing and classification, which provide a more geometric and intuitive meaning than the geometric features. It was M K Hu [9] that first set out the mathematical foundation for two-dimensional moment invariants. Hu has also defined seven of these moment invariants computed from central moments through order three that are invariant under object translation, scaling and rotation. Accordingly, we consider using these Hu moment invariants as classification features in this paper. The values of them can be calculated from contours using Chen's improved moments [10] as follows: The Chen's improved geometrical moments of order (p + q) are defined as:

The Chen's improved geometrical moments of order $(p+q)$ are defined as :

$$M_{pq} = \int_c x^p y^q ds \ . \tag{9}$$

where $p, q = 0, 1, 2, \ldots$, $\int_c$ is the line integral along a closed contour C and $ds = \sqrt{(dx)^2 + (dy)^2}$.

For practical implementation M_{pq} could be computed in their discrete form:

$$M_{pq} = \sum_{(x,y)\in C} x^p y^q \ . \tag{10}$$

Then the contour central moments can be calculated as follows:

$$\mu_{pq} = \int_c (x - \bar{x})^p (y - \bar{y})^q ds \ . \tag{11}$$

$$\bar{x} = \frac{M_{10}}{M_{00}} \quad \bar{y} = \frac{M_{01}}{M_{00}} \ . \tag{12}$$

In the discrete case μ_{pq} above becomes:

$$\mu_{pq} = \sum_{(x,y)\in\ C} (x - \bar{x})^p (y - \bar{y})^q \ . \tag{13}$$

These new central moments are further normalized using the following formula:

$$\eta_{pq} = \frac{\mu_{pq}}{\mu_{00}^{\gamma}} \ . \tag{14}$$

where the normalization factor is $\gamma = p + q + 1$.The seven moment invariant values can then be calculated from the normalized central moments as follows:

$$\begin{aligned}
\phi_1 &= \eta_{20} + \eta_{02} \\
\phi_2 &= (\eta_{20} - \eta_{02})^2 + 4\eta_{11}^2 \\
\phi_3 &= (\eta_{30} - 3\eta_{12})^2 + (\eta_{03} - 3\eta_{21})^2 \\
\phi_4 &= (\eta_{30} + \eta_{12})^2 + (\eta_{03} + \eta_{21})^2 \\
\phi_5 &= (3\eta_{30} - 3\eta_{12})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{03} + \eta_{21})^2] + (3\eta_{21} - \eta_{03})(\eta_{21} \\
&\quad +\eta_{03}) \times [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \\
\phi_6 &= (\eta_{20} - \eta_{02})[(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] + 4\eta_{11}(\eta_{30} + \eta_{12})(\eta_{21} + \eta_{03}) \\
\phi_7 &= (3\eta_{21} - \eta_{03})(\eta_{30} + \eta_{12})[(\eta_{30} + \eta_{12})^2 - 3(\eta_{21} + \eta_{03})^2] + (3\eta_{12} - \eta_{30})(\eta_{21} \\
&\quad +\eta_{03}) \times [3(\eta_{30} + \eta_{12})^2 - (\eta_{21} + \eta_{03})^2] \ .
\end{aligned} \tag{15}$$

Table 1 shows the values of eight geometric features and seven Hu moment invariants of leaf in Fig.1 (g).From Table 1 it could be seen that values of geometric features and moment invariants are different greatly in order of magnitude, therefore, be-fore classification these features need to be normalized as follows:

$$F = \frac{F - F_{min}}{F_{max} - F_{min}} \ . \tag{16}$$

where F denote one feature, F_{max} is the maximum value of all features in the same class with F , F_{min} is the minimum one.

Table 1. Values of eight geometric features and seven Hu moment invariants corresponding to leaf in Fig.1 (g)

Aspect Ratio	1.098	ϕ_1	1.768187e-001
Rectangularity	0.751	ϕ_2	5.537947e-003
Area Convexity	0.977	ϕ_3	3.432757e-005
Perimeter Convexity	1.052	ϕ_4	2.816526e-006
Sphericity	0.540	ϕ_5	2.603669e-011
Circularity	0.921	ϕ_6	1.662384e-007
Eccentricity	1.566	ϕ_7	9.437735e-012
Form Factor	0.768		

3 Moving Center Hyperesphere Classifier

In this paper we regard one feature vector containing eight geometric features and seven moment invariants as a pattern. Considering that the number of patterns and the dimension of pattern space are both very large, if we use conventional classifier like K-NN or Neural Network, the corresponding classification process would be quite time-consuming and space-consuming. Therefore, we propose using a moving center hypersphere(MCH) classification method to perform the plant leaves classification, which fundamental idea is that we regard each class of patterns as a series of "hyper-spheres", while in conventional approaches these patterns from one class are all treated as a set of "points". The first step of this method is to compute the multidimentional median of the points of the considered class, and set the initial center as the closest point from that class to that median. Then we find the maximum radius that can encompass the points of the class. Through a certain iteration we remove the center of the hypersphere around in a way that would enlarge the hypersphere and have it encompass as many points as possible. This is performed by having the center "hop" from one data point to a neighboring point. Once we find the largest possible hypersphere, the points inside this hypersphere are removed, and the whole procedure is repeated for the remaining points of the class. We continue until all points of that class are covered by some hyperspheres. At that point, we tackle the points of the next class in a similar manner. Here, we take one class for example to summarize the whole iterating training procedure of the hypersphere classifier as follows:

Step 1. Put all the training data points into set S, set hypersphere index $k = 0$.
Step 2. Select the closest point to the median of points in S as the initial center of the hypersphere k.

Step 3. Find the nearest point to the center from all other classes, and denote the distance as $d1$.

Step 4. Find the farthest point of the same class inside the hypersphere k with radius $d1$ to the center. Let $d2$ denote the distance from the center to that farthest point.

Step 5. Set the radius of the hypersphere k as $(d1 + d2)/2$.

Step 6. Select the point in the most negative direction of the center to the nearest point of the other classes among the nearest m points in this class.The purpose is to move the center to the new point to enlarge the hypersphere.If point exists then set the point as new center of the hypersphere k, return to step 4;otherwise continue.

Step 7. Remove those points covered by the hypersphere k from the set S. If S is still not empty then $k = k+1$, return to Step 2; otherwise remove the redundant hyperspheres that are totally enclosed by larger hyperspheres of this class, training finishes.

After the training is finished, MCH system is required to be able to classify any given input data point. The perpendicular distances from the input data point to the outside surface of all hyperspheres (with that distance counting as negative if the point is inside the hypersphere) are selected as the classification criterion. Assume that there are totally hyperspheres after training, each of which has the radius of $r_i(i = 1, 2, \cdots, H)$, and let d_i denote the distance between the data point and the center of hypersphere h_i, then we can define the decision rule as follows:

$$I = arg\ min(d_i - r_i), i \in \{1, 2, \cdots, H\} \ . \tag{17}$$

where I means index for nearest neighbor hypersphere.

4 Experimental Results

To verify the MCH classifier we have taken 800 leaf samples corresponding to 20 classes of plants collected by ourselves such as ginkgo, seatung, maple, etc (as shown in Fig.2). Each class includes 40 leaf samples, of which 25 samples are selected randomly as training samples and the remaining is used for testing samples. In our implementation we used Visual C++ 6.0 on Windows XP operating system running on Intel PC 2.4GHZ 512MB RAM.

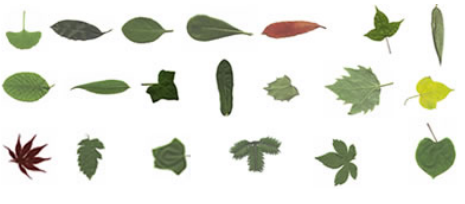

Fig. 2. Twenty classes of plant leaves used for recognition

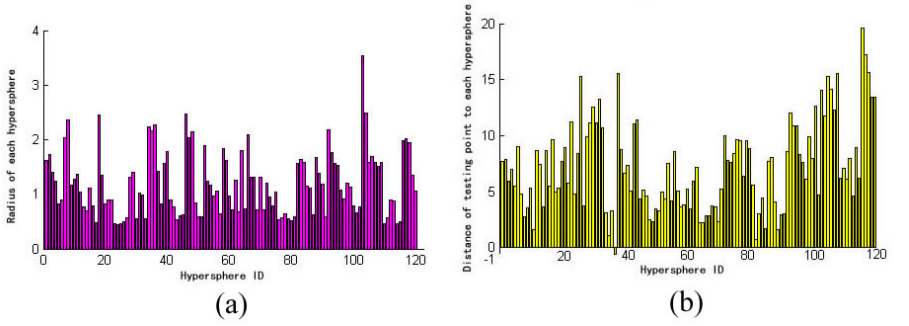

Fig. 3. (a) The Histogram for each hyperspheres' radius after training 500 samples corresponding to 20 classes. (b) The histograms for the distances of one Ginkgo testing sample to outside surface of each hypersphere.

Table 2. Class that each hypersphere belongs to

class ID	hypersphere ID	class ID	hypersphere ID	class ID	hypersphere ID
1	1 - 6	8	52 - 57	15	88 - 91
2	7 - 17	9	58 - 61	16	92 - 96
3	18 - 27	10	62 - 63	17	97 - 10
4	28 - 33	11	64 - 65	18	103 - 108
5	34 - 38	12	66 - 71	19	109 - 115
6	39 - 45	13	72 - 81	20	116 - 120
7	46 - 51	14	82 - 87		

* class ID explanation: 1 Sweet Osmanther 2 Seatung 3 Chinese Box 4 Photinia 5 Gingkgo 6 Tuliptree 7 London Planetree 8 Red Maple 9 Donglas Fir 10 SevenAngle 11 Beautiful Sweetgum 12 Panicled Goldraintree 13 China Floweringquince 14 Rose bush 15 Chrysanthemum 16 Sunflower 17 China Redbud 18 Bamboo 19 Plum 20 Willow.

Fig.3 (a) shows the histogram for each hyperspheres' radius after training 500 samples. Class that each hypersphere belongs to is listed in Table 2. It can be seen from Fig.3 (a) and Table 2 that there are total 120 hyperspheres obtained after training finished. The histogram for the distance of one ginkgo testing sample point to the outside surface of each hypersphere is shown in Fig.3 (b), in which the distance from the input data point to the outside surface of 37rd hypersphere is negative. It can be inferred that this point is just inside the 37rd hypersphere. From Table 2 it can be queried that 37rd hypersphere just encompasses part of ginkgo training sample points.

Here we also used the same data to verify 1-NN, 4-NN and BPNN (15 input nodes, 20 output nodes and one hidden layer with 12 nodes). Performance comparisons of MCH classifier with the other three classifiers are shown in Table 3 and Fig.4.

Table 3. Performance comparisons of four classifiers

	Training time(ms)	Classifying time(ms)	Storage vectors	Average correct rate(%)
1-NN	/	17.2	500	92.6
4-NN	/	42.7	500	92.3
BPNN	3720	7.6	/	92.4
H-S	36.2	9.8	120	92.2

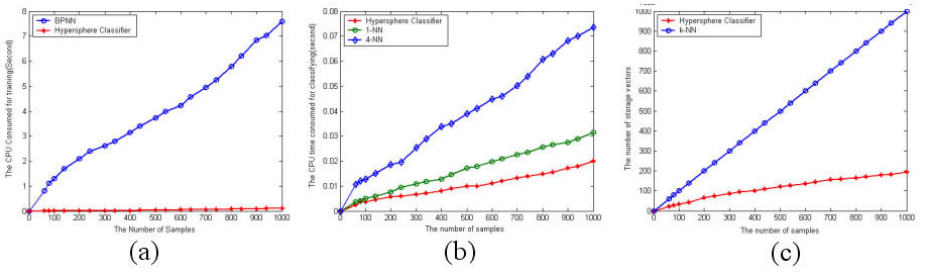

Fig. 4. (a) The CPU time consumed for training vs the number of samples (b) The CPU time consumed for classifying vs the number of samples (c) The numbers of stored vectors vs the number of samples

From Table 3 it can be seen that classifying time for the MCH classifier is shorter than the ones for 1-NN and 4-NN classifiers, and the storage vector number for the former is smaller than the ones for the latter two, and the correct recognition rate for the MCH classifier is similar to the ones for the latter two, and the MCH classifier needs to be trained while the latter two do not need at all. Here BPNN is also used to compare with MCH classifier. For the trained BPNN we only need to store its weight matrix so that storage vectors of BPNN can be ignored. However, there are two main drawbacks with BPNN, one is its slow convergence rate, and the other is that it sometimes falls into local minimum instead of global minimum. The slow convergence rate will pose a computational problem, and local minima will degrade the classification accuracy. In our experiment we found that if the number of training samples was greater than 5000, the convergence speed of BPNN would be very slow. As we know the leaf images classification is a large data set problem which has a serious speed issue and need to be made much more efficient. Fig.4 also shows that the larger the data set, the more improvement in speed and storage space which MCH classifier can make. So it can be concluded that the MCH classifier is a more preferred candidate for the leaf images classification than k-NN and BPNN because it can not only save the storage space but also reduce the time consumed without sacrificing the classification accuracy.

5 Conclusions

In this paper, we proposed using a moving center hypersphere(MCH) classifier to address shape features extracted from pre-processed images. Experimental results show that 20 classes of practical plant leaves are successfully recognized, and the average recognition rate is up to 92.2 percent. The performance of MCH classifier is satisfying while compared to nearest neighbor method and BPNN. Our future research works will include how to classify the leaves with deficiencies and combine adaptive neural networks with hypersphere technique to increase the correct recognition rate.

References

1. Milan Sonka, Vaclav Hlavac and Roger Boyle: Image Processing,Analysis and Machine Vision, Second Edition. Posts & Telecom Press, Beijing (2002)
2. Huang, D.S.: Systematic Theory of Neural Networks for Pattern Recognition. Publishing House of Electronic Industry of China, Beijing (1996)
3. Zhaoqi Bian and Xuegong Zhang: Pattern Recognition. TsingHua University Press, Beijing (1999)
4. Im, C., Nishida, H., Kunii, T.L.: Recognizing Plant Species by Leaf Shapes-a Case Study of the Acer Family. Proc. Pattern Recognition. **2** (1998) 1171–1173
5. Wang, Z., Chi, Z., Feng, D.: Fuzzy Integral for Leaf Image Retrieval. Proc. Fuzzy Systems. **1** (2002) 372–377
6. Saitoh, T., Kaneko, T.: Automatic Recognition of Wild Flowers. Proc. Pattern Recognition. **2** (2000) 507–510
7. Ridler, T.W. and Calvard, S.: Picture Thresholding Using An Iterative Selection Method. IEEE Transaction on System, Man and Cybernetics. **8, 8** (1978) 630–632
8. Vincent, L. and Soille, P.: Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations. IEEE Transactions on Pattern Analysis and Machine Intelligence. **13, 6** (1991) 583–598
9. Hu, M.K.: Visual Pattern Recognition by Moment Invariants. IRE Transaction Information Theory. **8, 2** (1962) 179–187
10. Chaur-Chin Chen: Improved Moment Invariants for Shape Discrimination. Pattern Rec-ognition. **26, 5** (1993) 683–686
11. Hart, P.E.: The Condensed Nearest Neighbor Rule. IEEE Transactions on Information Theory. **14, 3** (1967) 515–516
12. Huang, D.S.: Radial Basis Probabilistic Neural Networks: Model and Application. International Journal of Pattern Recognition and Artificial Intelli-gence. **13, 7** (1999) 1083–1101
13. Huang, D.S., Horace, H.S.Ip, Law Ken C.K. and Zheru Chi: Zeroing Polynomials Using Modified Constrained Neural Network Approach. IEEE Trans. On Neural Networks. **16, 3** (2005) 721–732
14. Huang, D.S., Horace, H.S.Ip and Zheru Chi: A Neural Root Finder of Polynomials Based on Root Moments. Neural Computation. **16, 8** (2004) 1721–1762
15. Huang, D.S.: Application of Generalized Radial Basis Function Networks to Recognition of Radar Targets. International Journal of Pattern Recognition and Artificial Intelligence. **13, 6** (1999) 945–962