한국퍼지 및 지능 시스템학회 논문지 1998, Vol. 8, No. 4. (1998) 18~22

A Fuzzy C Elliptic Shells Clustering

Daijin Kim

Department of Computer Engineering, DongA University

ABSTRACT

This paper presents a fuzzy c elliptic shells algorithm that detects clusters that can be expressed by hyperellipsoidal shells. The algorithm is computationally efficient since the prototypes of shell clusters are determined by a simple matrix inversion instead of by solving several nonlinear equations. The algorithm also works when the detected shells are partial the optimal number of clusters is unknown initially. A set of simulation results validates the proposed clustering method.

1. Introduction

Many fuzzy clustering algorithms suggested are based on an iterative method that minimizes an object function (based on an distance measure) iteratively in order to derive an successful partition. The shapes of partitions depend on either the norm used to define the distances or the kind of cluster prototypes used. For example, the k-means algorithm finds clusters of hyperspherical shapes using the Euclidean distance. However, the proposed algorithms are not suitable for detecting clusters with hollow interiors.

Recently, Dave[1] suggested a fuzzy c-shells (FCS) algorithm that detects clusters of circular arcs. The algorithm has also been generalized to the case of elliptical shells. However, the FCS algorithm requires an extensive amount of computation since it uses the Newton's method to solve two coupled nonlinear equations for the center and radius of each cluster in each iteration. Krishnapuram *et al.*[2] proposed an fuzzy c spherical shells (FCSS) algorithm that overcomes the computational burden arising from Newton's method by computing the prototypes from a simple matrix inversion. However, the algorithm can detect only the clusters of circular arcs. We extend the FCSS algorithm to detect clusters of elliptic arcs (including circular arcs), which is called as a fuzzy c elliptic shells (FCES) algorithm.

This paper is organized as follows. Section 2 presents the fuzzy c elliptic shells algorithm and introduces an estimation method of optimal prototypes. Section 3 describes a way of determining the optimum number of clusters. Section 4 shows simulation results using several synthetic images. Finally, a conclusion is drawn.

2. Fuzzy C Elliptic Shells Algorithm

Assume that each cluster represents a p-dimensional hyperellipsoidal shell. Then, the prototypes λ_i consist of three parameters ($\boldsymbol{c}_i$, r_i, $\boldsymbol{A}_i$), where $\boldsymbol{c}_i$ is the center of the hyperellipsoid, r_i is the size, and $\boldsymbol{A}_i$ is a $p \times p$ positive definite matrix characterizing the eccentricity and orientation of the ellipsoid. The distance d_{ij}^2 of a feature point $\boldsymbol{x}_j$ from the ith hyperellipsoidal shell prototype λ_i is defined as

$$d_{ij}^2 = d^2(\boldsymbol{x}_j, \lambda_i) = ((\boldsymbol{x}_j - \boldsymbol{c}_i)^T \boldsymbol{A}_i (\boldsymbol{x}_j - c_i) - r_i^2)^2 \quad (1)$$

Therefore, the value of distance decreases as the $\boldsymbol{x}a_j$ is close to the specific hyperellipsoid.

We define the objective function to be minimized when the degree that $\boldsymbol{x}_j$ belongs to a cluster λ_i is fuzzy as

$$J(L, U) = \sum_{i=1}^{K} \sum_{j=1}^{N} (\mu_{ij})^m d_{ij}^2 \quad (2)$$

where K is the total number of clusters, N is the total number of feature vectors, $\boldsymbol{L}=(\lambda_1, \lambda_2, \cdots, \lambda_K)$, $m \in [1, \infty)$ is a weighting exponent called the fuzzifier, and $\boldsymbol{U}=[\mu_{ij}]$ is a $K \times N$ matrix called the fuzzy K-partition matrix[3] satisfying the following two conditions:

This paper was supported by the Dong-A university Research Fund, 1998.

$$\sum_{i=1}^{K}\mu_{ij} = 1\,(j = 1, 2, \cdots, N),$$

$$0 < \sum_{j=1}^{N}\mu_{ij} < N\,(i = 1, 2, \cdots, K) \tag{3}$$

Here $\mu_{ij}\in\Xi[0, 1]$ is the grade of membership of the feature point $\boldsymbol{x}_j$ into a cluster λ_i.

In order to minimize the objective function in Eq. (2), we rewrite the distance in Eq. (1) as

$$d_{ij}^2 = \boldsymbol{p}_i^T \boldsymbol{M}_j \boldsymbol{p}_i + \boldsymbol{v}_j^T \boldsymbol{p}_i + b_j, \tag{4}$$

where

$$b_j = \boldsymbol{x}_j^T \boldsymbol{A}_i \boldsymbol{x}_j, \quad \boldsymbol{v}_j = 2b_j \boldsymbol{y}_j, \quad \boldsymbol{y}_j = \begin{bmatrix} \boldsymbol{x}_j \\ 1 \end{bmatrix}$$

$$\boldsymbol{M}_j = \boldsymbol{y}_j \boldsymbol{y}_j^T, \quad \boldsymbol{p}_i = \begin{bmatrix} -(\boldsymbol{A}_i + \boldsymbol{A}_i^T)\boldsymbol{c}_i \\ \boldsymbol{c}_i^T \boldsymbol{A}_i \boldsymbol{c}_i - r_i^2 \end{bmatrix} \tag{5}$$

By combining Eq. (5) into Eq. (2), Eq. (2) can be rewritten as

$$J(\boldsymbol{L}, \boldsymbol{U}) = \sum_{i=1}^{K}\sum_{j=1}^{N}(\mu_{ij})^m \cdot (\boldsymbol{p}_i^T \boldsymbol{M}_j \boldsymbol{p}_i + \boldsymbol{v}_j^T \boldsymbol{p}_i + b_j) \tag{6}$$

Assuming that the vectors $\boldsymbol{p}_i$ are independent of each other, the vectors $\boldsymbol{p}_i$ that minimize the objective function J($\boldsymbol{L}$, $\boldsymbol{U}$) must satisfy the following condition.

$$\sum_{j=1}^{N}(\mu_{ij})^m (2\boldsymbol{M}_j \boldsymbol{p}_i + \boldsymbol{v}_j) = 0 \tag{7}$$

If we define

$$\boldsymbol{H}_i = \sum_{j=1}^{N}(\mu_{ij})^m \boldsymbol{M}_j, \quad \boldsymbol{w}_i = \sum_{j=1}^{N}(\mu_{ij})^m \boldsymbol{y}_i \tag{8}$$

the solution of Eq. (7) is given as

$$\boldsymbol{p}_i = -\frac{1}{2}(\boldsymbol{H}_i)^{-1} \boldsymbol{w}_i \tag{9}$$

By equating Eq. (9) to Eq. (5), we can obtain the center $\boldsymbol{c}_i$ and the size r_i of the prototype λ_i.

One more parameter $\boldsymbol{A}_i$ of the prototype λ_i can be estimated by using the Dave's adaptive norm theorem [4]. According to the theorem, for fixed m and $det(\boldsymbol{A}_i) = \rho_i$, the estimated value $\hat{\boldsymbol{A}}_i$ is a local minimum of the functional

$$[\rho_i \times \det(\boldsymbol{S}_i)]^{1/p}\,(\boldsymbol{S}_i)^{-1}, \quad 1 \le i \le K \tag{10}$$

where

$$\boldsymbol{S}_i = \sum_{j=1}^{N}(\mu_{ij})^m \frac{d_{ij}}{D_{ij}}(\boldsymbol{x}_j - \boldsymbol{c}_j)(\boldsymbol{x}_j - \boldsymbol{c}_j)^T \tag{11}$$

where

$$D_{ij} = [(\boldsymbol{x}_j - \boldsymbol{c}_j)^T \boldsymbol{A}_i (\boldsymbol{x}_j - \boldsymbol{c}_j)]^{1/2} \tag{12}$$

Note that $\boldsymbol{S}_i$ is the cluster scatter matrix of distances of point $\boldsymbol{x}_j$ from the shells and D_{ij} is the distance of point $\boldsymbol{x}_j$ from the cluster center $\boldsymbol{c}_i$. In the simulations, $\rho_i = \det(\boldsymbol{A}_i) = 1$ is chosen for keeping the volume under the geometric transformations constant and an identity matrix I is used for the initial value of $\boldsymbol{A}_i$.

The fuzzy K-partition matrix $\boldsymbol{U}$ is updated by the Bezdek's formula[3] as

$$\mu_{ij} = \begin{cases} \dfrac{1}{\sum_{k=1}^{K}\left(\frac{d_{ij}}{d_{kj}}\right)^{\frac{1}{m-1}}} & \text{if } \boldsymbol{I} = \Phi \\ 0 & \text{if } \boldsymbol{I} \neq \Phi \text{ and } \vee i \in \bar{\boldsymbol{I}}_j \\ \sum_{i=1}^{K}\mu_{ij} = 1 & \text{if } \boldsymbol{I} \neq \Phi \text{ and } \vee i \in \boldsymbol{I}_j \end{cases} \tag{13}$$

where $\boldsymbol{I}_j = \{i \mid 1 \le i \le K,\ d_{ij}^2 = 0\}$, and $\bar{\boldsymbol{I}}_j = \{1, 2, \cdots, K\} - \boldsymbol{I}_j$. They are the sets of points located just on the shells. The stop condition is the Frobenius norm of the difference between two consecutive partition matrix $|\boldsymbol{U}^l - \boldsymbol{U}^{(l-1)}|$ is less than a predefined infinitesimal ε. The fuzzy c elliptic shells algorithm is given below.

The Fuzzy C Elliptic Shells Algorithm

Fix the number of clusters K;

Fix the fuzzifier constant m=2;

Initialize $\boldsymbol{U}$ to $\boldsymbol{U}^{(0)}$ by fuzzy k-means algorithm;

Initialize $\boldsymbol{A}_i$ to an identity matrix $\boldsymbol{I}_{p\times p}$;

Set iteration counter l=1;

Repeat

Calculate $\boldsymbol{H}_i^l$ and $\boldsymbol{w}_i^l$ for each cluster λ_i using Eq. (8);

Calculate $\boldsymbol{p}_i^l$ for each cluster λ_i using Eq. (9);

Estimate $\boldsymbol{c}_i^l$ and r_i^l for each cluster λ_i using Eq. (5) and (9);

Calculate $\boldsymbol{S}_i^l$ and D_{ij}^l for each cluster λ_i using Eq. (11) and (12);

Estimate $\boldsymbol{A}_i^l$ for each cluster λ_i using Eq. (10);

Update $\boldsymbol{U}_l$ using Eq. (13);

Increment iteration count l;

Until ($|U^{(l-1)}-U^{l}| < \varepsilon$);

3. Determination of the Optimal Number of Clusters

The optimum number of clusters is determined as the one that satisfies the cluster validity criteria best. Often, fuzzy hypervolume, average partition density, or average shell thickness are used as the validity criteria of shell clusters[5]. In this paper, the average shell thickness is chosen as the validity criteria due to its computational simplicity. In fuzzy case, the average shell thickness is defined as

$$T_i = \frac{\sum_{j=1}^{N}(\mu_{ij})^m((\boldsymbol{x}_j-\boldsymbol{c}_c)^T\boldsymbol{A}_i(\boldsymbol{x}_j-\boldsymbol{c}_i)-r_i^2)^2}{r_i\sum_{j=1}^{N}(\mu_{ij})^m} \quad (14)$$

An overall validity measure over the all clusters is calculated by adding the individual average shell thickness as

$$T = \sum_{i=1}^{K} T_i \quad (15)$$

The determination method progressively clusters the input data starting from the overestimated number of clusters K_{max}. We apply the FCES algorithm with $K=K_{max}$. After the algorithm converges, two compatible clusters λ_i and λ_j that satisfy the following conditions are merged into one. Whenever the merging occurs, the value of K is reduced by one.

$$|\boldsymbol{c}_i-\boldsymbol{c}_j|<\varepsilon_1 \text{ and } |r_i-r_j|<\varepsilon_2 \text{ and } |\boldsymbol{A}_i-\boldsymbol{A}_j|<\varepsilon_3 \quad (16)$$

Next, a cluster is a good one when the thickness of the cluster is less than an threshold value (T_{min}) and the number of points in the cluster is sufficiently large as

$$T_i < T_{min} \text{ and } N_i \gg N_{min} \quad (17)$$

where $N_{min}=\frac{N}{K+1}$ is often chosen.

Next, all points that are near to the good cluster (satisfying Eq. (18)) are removed from the input data temporarily in order to reduce the computational effort.

$$|((\boldsymbol{x}_j-\boldsymbol{c}_c)^T\boldsymbol{A}_i(\boldsymbol{x}_j-\boldsymbol{c}_i)-r_i^2)|<D_{min} \quad (18)$$

Next, the clusters having too small number of points are eliminated from the cluster set and the number of clusters K is reduced by the number of the eliminated clusters. The FCES algorithm is applied for the reduced number of K again. This process will be repeated until no more merging, removing, and elimination occurs, i.e. until the value of K can no longer be reduced.

4. Simulation Results

The FCES algorithm was tested using six two-dimensional radnomly generated synthetic data with the circles and ellipses. Each data point has an position error of $\pm 10\%$ from the original position along the x and y axis. Experimental parameters used are m=2, $\varepsilon_1=\varepsilon_2=\varepsilon_3$=0.01, and K_{max}=10. All other parameters are determined experimentally. In each figure, each clusters are represented by different symbols and each point is assigned to the cluster with the maximum value of membership. When the clusters are separated each other as in Fig. 1 and 3, the FCES algorithm works perfectly without misclassifications. When the clusters are crossed each other as in Fig. 2 and 4, some points near to the crossings are misassigned. Simulation result shows that the membership values of points near to the crossings are similar each other and the values are close to the inverse of the number of total crossed clusters. Thus, it is reasonable to decide that some points near to the crossings are common

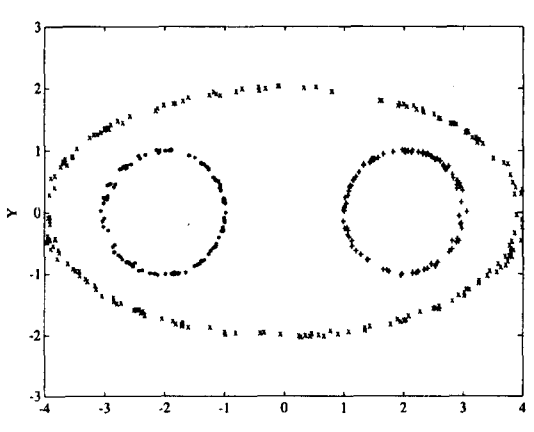

Fig. 1. Result of FCES for one ellipse and two cirlces.

to all the crossed clusters rather than to assign a cluster with the maximum membership value. Fig. 5 shows that the FCES algorithm also works well when the input data is given partially. Some errors occur where two ellipses are crossed. (One is a real one that is represented by input data and another is an imaginary one that is obtained by reflecting the real one on the x-axis). In this case, it is reasonable to decide that those error points are really erroneous. Fig. 6 shows a little errors compared to the previous cases. This is because there exists non-symmetric clusters, which violates the assumption that the clusters are symmetric when A_i is estimated. Table 1 shows the comparison results between the real parameters and estimated parameters by the FCES algorithm. The number of clusters is determined correctly in all cases and the estimated prototypes are nearly same as those of true prototypes.

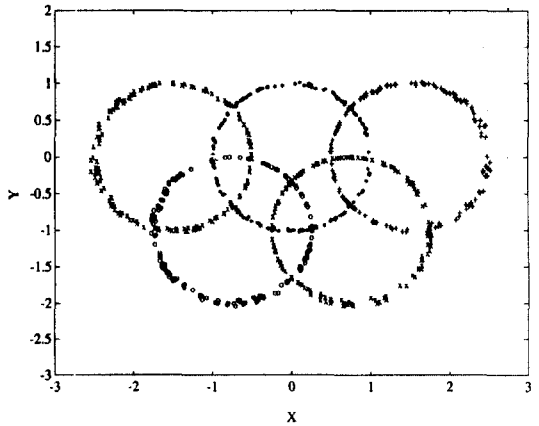

Fig. 2. Result of FCES for the olympic flag.

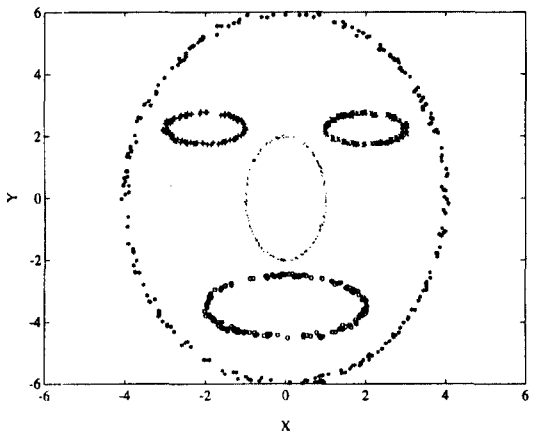

Fig. 3. Result of FCES for a face.

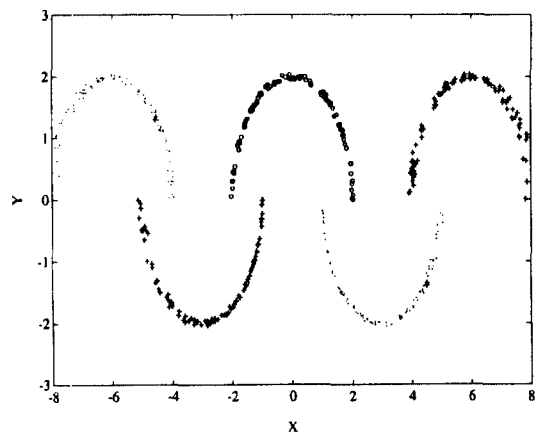

Fig. 4. Result of FCES for three ellipses.

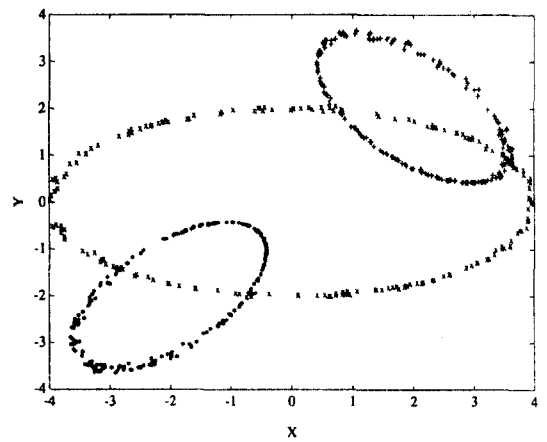

Fig. 5. Result of FCES for five partial circles.

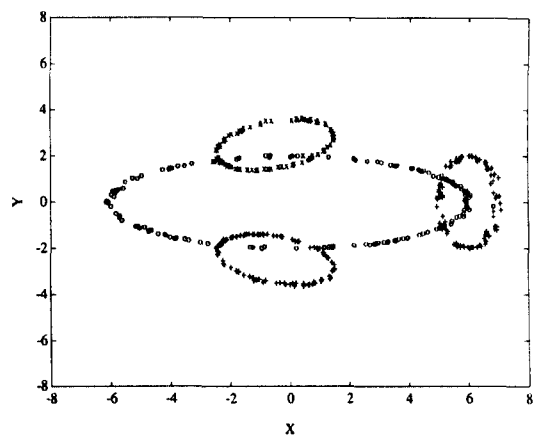

Fig. 6. Result of FCES for a fish.

Table 1. Comparison of the real and estimated parameters

		Real parameters			Estimated parameters		
		c_i	r_i	A_i	$\hat{c}_i$	$\hat{r}_i$	$\hat{A}_i$
Fig. 1	λ_1	(-2.0, 0.0)	1.0	(1.0, 0.0, 0.0, 1.0)	(-2.01, -0.01)	1.01	(0.93, 0.02, 0.02, 1.07)
	λ_2	(2.0, 0.0)	1.0	(1.0, 0.0, 0.0, 1.0)	(2.01, 0.01)	1.00	(0.93, 0.02, 0.02, 1.07)
	λ_3	(0.0, 0.0)	2.8	(0.5, 0.0, 0.0, 2.0)	(-0.00, -0.00)	2.83	(0.46, -0.01, -0.01, 2.16)
Fig. 2	λ_1	(0.0, 0.0)	1.0	(1.0, 0.0, 0.0, 1.0)	(-0.00, -0.00)	1.00	(0.98, 0.00, 0.00, 1.02)
	λ_2	(1.5, 0.0)	1.0	(1.0, 0.0, 0.0, 1.0)	(1.50, 0.00)	1.00	(0.94, 0.03, 0.03, 1.06)
	λ_3	(-1.5, 0.0)	1.0	(1.0, 0.0, 0.0, 1.0)	(-1.49, 0.01)	1.00	(0.90, 0.04, 0.04, 1.12)
	λ_4	(0.75, -1.0)	1.0	(1.0, 0.0, 0.0, 1.0)	(0.75, -1.01)	1.01	(0.99, -0.07, -0.07, 1.02)
	λ_5	(-0.75, -1.0)	1.0	(1.0, 0.0, 0.0, 1.0)	(-0.74, -0.99)	0.99	(0.99, -0.09, -0.09, 1.02)
Fig. 3	λ_1	(0.0, 0.0)	5.0	(1.5, 0.0, 0.0, 1.5)	(0.00, 0.01)	4.90	(1.59, -0.01, -0.01, 1.63)
	λ_2	(-2.0, 2.25)	0.7	(0.5, 0.0, 0.0, 2.0)	(-2.01, 2.20)	0.71	(0.51, -0.03, -0.03, 1.95)
	λ_3	(2.0, 2.25)	0.7	(0.5, 0.0, 0.0, 2.0)	(2.01, 2.25)	0.71	(0.55, 0.13, 0.13 1.84)
	λ_4	(0.0, 0.0)	1.4	(2.0, 0.0, 0.0, 0.5)	(-0.01, -0.01)	1.42	(1.95, 0.09, 0.09, 0.51)
	λ_5	(0.0, -3.5)	1.4	(0.5, 0.0, 0.0, 2.0)	(0.04, -3.38)	1.52	(0.69, -0.13, -0.13, 1.88)
Fig. 4	λ_1	(0.0, 0.0)	2.8	(0.5, 0.0, 0.0, 2.0)	(-0.03, 0.01)	2.82	(0.56, 0.10, 0.10, 1.79)
	λ_2	(-2.0, -2.0)	1.4	(1.4, -0.7, -0.7, 1.4)	(-2.00, -1.99)	1.42	(1.28, -0.73, -0.73, 1.19)
	λ_3	(2.0, 2.0)	1.4	(1.4, 0.7, 0.7, 1.4)	(2.00, 1.99)	1.42	(1.25, 0.89, 0.89, 1.44)
Fig. 5	λ_1	(0.0, 0.0)	2.0	(1.0, 0.0, 0.0, 1.0)	(0.00, -0.00)	2.01	(1.14, 0.04, 0.04, 0.87)
	λ_2	(-3.0, 0.0)	2.0	(1.0, 0.0, 0.0, 1.0)	(-3.01, 0.01)	2.01	(1.03, -0.19, -0.19, 1.00)
	λ_3	(3.0, 0.0)	2.0	(1.0, 0.0, 0.0, 1.0)	(2.99, -0.02)	1.98	(0.97, 0.23, 0.23, 1.09)
	λ_4	(-6.0, 0.0)	2.0	(1.0, 0.0, 0.0, 1.0)	(-6.02, -0.06)	2.05	(0.86, 0.11, 0.11, 1.17)
	λ_5	(6.0, 0.0)	2.0	(1.0, 0.0, 0.0, 1.0)	(6.01, -0.04)	2.03	(0.83, -0.16, -0.16, 1.23)
Fig. 6	λ_1	(0.0, 0.0)	3.5	(0.33, 0.0, 0.0, 3.0)	(0.04, -0.04)	3.46	(0.26, -0.03, -0.03, 3.13)
	λ_2	(6.0, 0.0)	1.4	(2.0, 0.0, 0.0, 0.5)	(6.02, 0.01)	1.43	(1.72, 0.04, 0.04, 0.58)
	λ_3	(-0.5, 2.5)	1.4	(0.5, -0.25, -0.25, 2.0)	(-0.51, 2.51)	1.42	(0.57, -0.31, -0.31, 1.97)
	λ_4	(-0.5, -2.5)	1.4	(0.5, 0.25, 0.25, 2.0)	(-0.48, -2.49)	1.41	(0.66, 0.34, 0.34, 1.87)

5. Conclusion

The proposed FCES algorithm is computationally efficient because it does not requrie to solve two coupled nonlinear equations for the center and radius of each cluster in each iteration and the eccentricity matrix is estimated from a closed functional. Also, it worked when the optimum number of cluster are not predetermined. The FCES algorithm can be further extended to detect other objects of arbitrary shapes simply by defining their corresponding geometric prototypes. One problem to be pursuit is how to determine the algorithmic execution parameters automatically.

References

[1] R. N. Dave, "Fuzzy-shell clustering and application to circle detection in digital images," *Int. J. of general Systems*, **16**, pp. 343-355, 1990.

[2] R. Krishnapuram, H. Frigui and O. Nasraoui, "New fuzzy shell clustering algorithms for boundary detection and pattern recognition," *Proc. SPIE Conf. Intelligent Robots and Computer Vision X: Algorithms and Techniques*, pp. 458-465, Boston, Nov. 1991.

[3] H. J. Zimmermann, *Fuzzy set theory and its applications*, Kluwer Academic Publishers, 1991.

[4] R. N. Dave and K. Bhaswan, "Adaptive fuzzy c-shells clustering and detection of ellipses," *IEEE Transactions on Neural Networks*, **3**(5), pp. 643-662, Sep. 1992.

[5] I. Gath and A. Geva, "Unsupervised optimal fuzzy clustering," *IEEE Trans. Pattern Anal. Mach. Intell.*, **11**, pp. 772-781, 1989.

김 대 진(Daijin Kim) 정회원

제 8권 제 2호 참조

현재 : 동아대학교 전기전자컴퓨터공학부 부교수