Applied Soft Computing 9 (2009) 226–236

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

Automatic image pixel clustering with an improved differential evolution

Swagatam Das *, Amit Konar

Department of Electronics & Telecommunication Engineering, Jadavpur University, Kolkata 700032, India

ARTICLE INFO

Article history:
Received 20 March 2006
Received in revised form 26 October 2007
Accepted 18 December 2007
Available online 26 April 2008

Keywords:
Differential evolution
Fuzzy clustering
Genetic algorithms
Image segmentation

ABSTRACT

This article proposes an evolutionary-fuzzy clustering algorithm for automatically grouping the pixels of an image into different homogeneous regions. The algorithm does not require a prior knowledge of the number of clusters. The fuzzy clustering task in the intensity space of an image is formulated as an optimization problem. An improved variant of the differential evolution (DE) algorithm has been used to determine the number of naturally occurring clusters in the image as well as to refine the cluster centers. We report extensive performance comparison among the new method, a recently developed genetic-fuzzy clustering technique and the classical fuzzy *c*-means algorithm over a test suite comprising ordinary grayscale images and remote sensing satellite images. Such comparisons reveal, in a statistically meaningful way, the superiority of the proposed technique in terms of speed, accuracy and robustness.

1. Introduction

Image segmentation may be defined as the process of dividing an image into disjoint homogeneous regions. These homogeneous regions usually contain similar objects of interest or part of them. The extent of homogeneity of the segmented regions can be measured using some image property (e.g., pixel intensity [1]). Segmentation forms a fundamental step towards several complex computer vision and image analysis applications including digital mammography, remote sensing and land cover study. Segmentation of non-trivial images is one of the most difficult tasks in image processing. Segmentation accuracy determines the eventual success or failures of computerized image analysis procedures.

Clustering can be defined as the optimal partitioning of a given set of n data points into c subgroups, such that data points belonging to the same group are as similar to each other as possible whereas data points from two different groups share the maximum difference. Image segmentation can be treated as a clustering problem where the features describing each pixel correspond to a pattern, and each image region (i.e., segment) corresponds to a cluster [1]. Therefore, many clustering algorithms have widely been used to solve the segmentation problem (e.g., *k*-means [2], FCM [3], ISODATA [4] and Snob [5]).

Clustering algorithms can be *hierarchical* or *partitional* [1,6]. Within each of the types, there exists a wealth of subtypes and different algorithms for finding the clusters. In hierarchical clustering, the output is a tree showing a sequence of clustering with each cluster being a partition of the dataset [7]. Hierarchical algorithms can be agglomerative (bottom-up) or divisive (top-down). Agglomerative algorithms begin with each element as a separate cluster and merge them in successively larger clusters. Divisive algorithms begin with the whole set and proceed to divide it into successively smaller clusters. Hierarchical algorithms have two basic advantages [6]. Firstly, the number of classes need not be specified a priori and secondly, they are independent of the initial conditions. However, the main drawback of hierarchical clustering techniques is they are static, i.e., data points assigned to a cluster cannot move to another cluster. In addition to that, they may fail to separate overlapping clusters due to lack of information about the global shape or size of the clusters.

Partitional clustering algorithms, on the other hand, attempt to decompose the dataset directly into a set of disjoint clusters. They try to optimize certain criteria (e.g., a square error function). The criterion function may emphasize the local structure of the data, as by assigning clusters to peaks in the probability density function, or the global structure. Typically, the global criteria involve minimizing some measure of dissimilarity in the samples within each cluster, while maximizing the dissimilarity of different clusters. The advantages of the hierarchical algorithms are the disadvantages of the partitional algorithms and vice versa. An extensive survey of various clustering techniques can be found in Ref. [1].

Clustering can also be performed in two different modes: *crisp* (or *hard*) and *fuzzy* (or *soft*). In crisp clustering, the clusters are disjoint and non-overlapping in nature. Any pattern may belong to one and only one class in this case. In case of fuzzy clustering, a pattern may belong to all the classes with a certain fuzzy membership grade [1]. Popular crisp clustering approaches do not consider overlapping of classes that occur in many practical image

* Corresponding author.
E-mail addresses: swagatamdas19@yahoo.co.in (S. Das), konaramit@yahoo.com (A. Konar).

1568-4946/$ – see front matter © 2008 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2007.12.008

segmentation problems. For example, in remote sensing satellite images, a pixel corresponds to an area of the land space, which may not necessarily belong to a single type of land cover. This in turn indicates that the pixels in a satellite image can be associated with a large amount of imprecision and uncertainty. Therefore, application of the principles of fuzzy set theory appears to be natural and appropriate in such domains.

The fuzzy *c*-means (FCM) [8] seems to be the most popular algorithm in the field of fuzzy clustering. Many researchers have attempted modifications of the classical FCM (to be detailed in Section 2.1) and their application to image segmentation in the past few years [9–15].

A considerable amount of research effort has gone in the recent past to evolve the naturally occurring clusters in a complex dataset. However, little work has been taken up to determine the optimal number of clusters at the same time. Most of the existing clustering techniques, from both crisp and fuzzy domains, accept the number of classes K as an input instead of determining the same on the run. Nevertheless, in many practical situations, the appropriate number of groups in a previously unhandled dataset may be unknown or impossible to determine even approximately. For example, in multi-agent robotic cooperation, a robot has to identify its fellow robots, obstacles, target, etc. from the images of its surroundings grabbed by a digital camera (working as the 'robot eye'). For good perception and sensing, the images taken by a robot need fast and automatic clustering so that similar objects can be marked identically in the image.

Finding an optimal number of clusters in a large dataset has always remained a challenging task. Several researchers [16,17] have investigated the problem but the outcome is still unsatisfactory [18]. Works on automatic clustering with evolutionary strategies (ES) [19], evolutionary programming (EP) [20] and genetic algorithm (GA) [21] have been reported in Refs. [22–24].

Recently, researchers working in this area have started taking some interest on two promising approaches to numerical optimization, namely the particle swarm optimization (PSO) [25] and the differential evolution (DE) [26]. In Ref. [27] Omran et al. proposed an image segmentation algorithm based on the PSO. The algorithm finds the centroids of a user-specified number of clusters, where each cluster groups together similar pixels. They used a crisp criterion function for evaluating the partitions on the image data. Only in 2005, the same authors came up with another automatic hard clustering scheme [28]. The algorithm starts by partitioning the dataset into a relatively large number of clusters to reduce the effect of the initialization. Using binary PSO [29], an optimal number of clusters is selected. Finally, the centroids of the chosen clusters are refined through the *k*-means algorithm. The authors applied the algorithm for segmentation of natural, synthetic and multi-spectral images. Omran et al. also devised a non-automatic crisp clustering scheme based on DE and illustrated the application of the algorithm to image segmentation problems in Ref. [30]. However, to the best of our knowledge, DE has not been applied to the *automatic fuzzy* clustering of image pixels until date.

The aim of the present work is to show that DE, with a modification of the chromosome representation scheme, can be used for the automatic determination of fuzzy clusters in the intensity space of an image. This leads to the segmentation of a previously unhandled image with minimal user intervention. Our new soft clustering scheme uses a modified DE algorithm in conjunction with the symmetry-based fuzzy clustering validity index recently reported in Ref. [31]. We changed the DE from its classical form to improve its convergence properties. In addition to that, we used a novel representation scheme for the search variables for determining the optimal number of clusters. The new algorithm, referred to here as AFDE (automatic fuzzy clustering differential evolution), is detailed in the subsequent sections. We provide extensive comparisons between our method, a very recent automatic fuzzy clustering technique [32,33] and the classical FCM algorithm in terms of final clustering accuracy, speed of obtaining an acceptable solution and algorithm robustness. The experimental results over several images of varying range of complexity show that the DE-based algorithm should be the first choice for off-line fuzzy image segmentation when accuracy is of prime importance.

The rest of the paper is organized as follows. In Section 2, we formally describe the problem of fuzzy clustering, briefly outline the FCM algorithm, and provide a short summary of different fuzzy clustering validity measures. Section 3 gives an outline of the classical DE scheme, explains the modifications used in the present work, and presents our DE-based soft segmentation algorithm. Experimental results along with statistical tests of significance are presented in Section 4. We conclude the paper in Section 5.

2. The fuzzy clustering problem

A pattern is a physical or abstract structure of objects. It is distinguished from others by a collective set of attributes called *features*, which together represent a pattern [34]. Let $P = \{P_1, P_2, \ldots, P_n\}$ be a set of n patterns or data points, each having d features. These patterns can also be represented by a profile data matrix $X_{n \times d}$ having n d-dimensional row vectors. The ith row vector x_i characterises the ith object from the set P and each element $x_{i,j}$ in $\vec{X}_i$ corresponds to the jth real value feature ($j = 1, 2, \ldots, d$) of the ith pattern ($i = 1, 2, \ldots, n$).

Given such an $X_{n \times d}$, a partitional clustering algorithm tries to find a partition $C = \{C_1, C_2, \ldots, C_c\}$ such that the similarity of the patterns in the same cluster C_i is maximum and patterns from different clusters differ as far as possible. The partitions should maintain the following properties:

(1) Each cluster should have at least one pattern assigned i.e., $C_i \neq \Phi \ \forall i \in \{1, 2, \ldots, c\}$.
(2) Two different clusters should have no pattern in common i.e., $C_i \cap C_j = \Phi$, $\forall i \neq j$ and $i, j \in \{1, 2, \ldots, c\}$.
(3) Each pattern should definitely be attached to a cluster i.e., $\bigcup_{i=1}^{c} C_i = P$.

In the case of fuzzy clustering, a pattern may belong to all the classes with a certain fuzzy membership grade for each class. So, in this case we need to evolve an appropriate partition matrix $U = [u_{ij}]_{c \times n}$, where $u_{ij} \in [0, 1]$, such that u_{ij} denotes the grade of membership of the jth element to the ith cluster. In fuzzy partitioning of the data, the following conditions hold:

$$\left.\begin{aligned} 0 < \sum_{j=1}^{n} u_{ij} < n \quad & \text{for } i = 1, 2, \ldots, c \\ \sum_{i=1}^{c} u_{ij} = 1 \quad & \text{for } j = 1, 2, \ldots, n \\ \sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij} = n & \end{aligned}\right\} \tag{1}$$

2.1. The fuzzy c-means algorithm

In the classical fuzzy *c*-means algorithm, a *within cluster sum* function J_m is minimized to evolve the proper cluster centers:

$$J_m = \sum_{j=1}^{n} \sum_{i=1}^{c} (u_{ij})^m \|\vec{X}_j - \vec{V}_i\|^2 \tag{2}$$

where $\vec{V}_i$ is the ith cluster center, $\vec{X}_j$ is the jth d-dimensional data vector and $\|\cdot\|$ is an inner product-induced norm in d dimensions. Given c classes, we can determine their cluster centers $\vec{V}_i$ for $i = 1$ to c by means of the following expression:

$$\vec{V}_i = \frac{\sum_{j=1}^{n} (u_{ij})^m \vec{X}_j}{\sum_{j=1}^{n} (u_{ij})^m} \tag{3}$$

Here m $(m > 1)$ is any real number that influences the membership grade. Now differentiating the performance criterion with respect to $\vec{V}_i$ (treating u_{ij} as constant) and with respect to u_{ij} (treating $\vec{V}_i$ as constant) and setting them to zero the following relation can be obtained:

$$u_{ij} = \frac{\|\vec{X}_j - \vec{V}_i\|^{2/1-m}}{\sum_{k=1}^{c} \|\vec{X}_j - \vec{V}_k\|^{2/1-m}} \tag{4}$$

2.2. Cluster validity indices in the fuzzy environment

To judge the quality of a partition provided by some clustering algorithm, it is necessary to have a well-defined statistical–mathematical function, called a *cluster validity index*, evaluated on the final clustering solutions. In what follows we describe the three well-known validity indices used in the experimental results reported in this paper.

2.2.1. Xie–Beni index

This index, due to Xie and Beni [35], is given by:

$$XB_m = \frac{\sum_{i=1}^{c} \sum_{j=1}^{n} u_{ij}^2 \|\vec{X}_j - \vec{V}_i\|^2}{n \times \min_{i \neq j} \|\vec{V}_i - \vec{V}_j\|^2} \tag{5}$$

The optimal number of clusters can be obtained by minimizing the index value.

2.2.2. The PS measure

This measure rests upon the concept of a *point symmetry distance, which* was shown to be very effective for clustering by Chou and co-workers [36,37]. The point symmetry distance is defined as follows. Given N patterns described by feature vectors $\vec{X}_k$, where $k = \{1, 2, \ldots, n\}$ and a reference vector $\vec{V}$ (e.g., a cluster center), the "point symmetry distance" between a pattern $\vec{X}_j$ and the reference vector $\vec{V}$ is defined as

$$d_s(\vec{X}_j, \vec{V}) = \min_{\substack{k = 1,2,\ldots,n \\ \text{and } k \neq j}} \left\{ \frac{\|(\vec{X}_j - \vec{V}) + (\vec{X}_k - \vec{V})\|}{(\|\vec{X}_j - \vec{V}\| + \|\vec{X}_k - \vec{V}\|)} \right\} \tag{6}$$

In Ref. [36], Su et al. defined a clustering validity index based on the point symmetry measure as

$$\mathrm{PS}(c) = \frac{1}{c} \sum_{i=1}^{c} \left[\frac{1}{n_i} \sum_{j \in C_i} \frac{d_s(\vec{X}_j, \vec{V}_i) \times d_e(\vec{X}_j, \vec{V}_i)}{\min_{\substack{m,n = 1,2,\ldots,c \\ \text{and } m \neq n}} \{d_e(\vec{V}_m, \vec{V}_n)\}} \right] \tag{7}$$

where d_e is the Euclidean distance between two points in the d-dimensional feature space, n_i is the number of data points belonging to cluster C_i. The cluster centers $\vec{V}_i$'s are calculated as per Eq. (3) and while computing the point symmetry distance d_s according to Eq. (6) we keep in mind that the vectors $\vec{X}_j$ and $\vec{X}_k$ belong to the same class C_i. Finally, the smallest PS(c) indicates a valid optimal partition with the optimal cluster number c. In Ref. [31] the usefulness of the PS measure has been shown over many difficult non-image datasets having clusters of widely different geometrical shapes.

2.2.3. The PBMF index

Recently Pakhira et al. have proposed a new clustering validity index [32], the hard version of which may be given as,

$$\mathrm{PBM}(c) = \left(\frac{1}{c} \times \frac{E_1}{E_c} \times D_c \right)^2 \tag{8}$$

where c is the number of clusters and

$$E_c = \sum_{i=1}^{c} E_i$$

$$\text{and} \qquad E_i = \sum_{j=1}^{n} u_{ij} \|\vec{X}_j - \vec{V}_i\|^2 \tag{9}$$

$$\text{and} \qquad D_c = \max_{i,j=1}^{c} \|\vec{V}_i - \vec{V}_j\| \tag{10}$$

$\vec{V}_i$ being the ith cluster center. u_{ij} is the (i, j)th entry of the partition matrix $U = [u_{ij}]_{c \times n}$. In Ref. [32], Pakhira et al. have also proposed a fuzzy version of this validity measure:

$$\mathrm{PBMF}(c) = \left(\frac{1}{c} \times \frac{E_1}{J_m} \times D_c \right) \tag{11}$$

$$\text{where} \qquad J_m = \sum_{j=1}^{n} \sum_{i=1}^{c} (u_{ij})^m \|\vec{X}_j - \vec{V}_i\| \tag{12}$$

In (8) the term E_1 usually appears as a constant for a particular dataset. The maximum value of the PBMF index is supposed to give the optimal number of clusters in a dataset.

3. DE-based automatic fuzzy clustering

3.1. The DE algorithm and its modification

The DE is a population-based global optimization algorithm that uses a floating-point (real-coded) representation. The ith individual (parameter vector or *chromosome*) of the population at generation (time-step) t is a D-dimensional vector containing a set of D optimization parameters:

$$\vec{Z}_i(t) = [Z_{i,1}(t), Z_{i,2}(t), \ldots, Z_{i,D}(t)] \tag{13}$$

Now in each generation (or one iteration of the algorithm) to change the population members $\vec{Z}_i(t)$ (say), a *donor* vector $\vec{Y}_i(t)$ is created. It is the method of creating this donor vector which demarcates between the various DE schemes. In one of the earliest variants of DE, now called DE/rand/1 scheme, to create $\vec{Y}_i(t)$ for each ith member, three other parameter vectors (say the r_1, r_2, and r_3th vectors (say the r_1, r_2, and r_3th vectors such that $r_1, r_2, r_3 \in [1, \mathrm{NP}]$ and $r_1 \neq r_2 \neq r_3$) are chosen at random from the current population. Next the difference of any two of the three vectors is multiplied by a scalar number F and the scaled difference is added to the third one, whence we obtain the donor vector $\vec{Y}_i(t)$. The process for the jth component of the ith vector may be expressed as,

$$Y_{i,j}(t) = Z_{r_1,j}(t) + F(Z_{r_2,j}(t) - Z_{r_3,j}(t)) \tag{14}$$

Next a crossover operation takes place to increase the potential diversity of the population. The DE family uses two kinds of crossover schemes, namely 'exponential' and 'binomial' [28]. To save space, we here briefly describe the binomial crossover, which is also employed by the modified DE algorithm. The binomial crossover is performed on each of the D variables whenever a randomly picked number between 0 and 1 is within the Cr value. In this case the number of parameters inherited from the mutant has a (nearly) binomial distribution. Thus for each target vector $\vec{Z}_i(t)$,

a trial vector $\vec{R}_i(t)$ is created in the following fashion:

$$R_{i,j}(t) = \begin{array}{ll} Y_{i,j}(t) & \text{if } \text{rand}_j(0,1) \le \text{Cr or } j = rn(i) \\ Z_{i,j}(t) & \text{if } \text{rand}_j(0,1) > \text{Cr or } j \ne rn(i) \end{array} \quad (15)$$

for $j = 1, 2, \ldots, D$ and $\text{rand}_j(0, 1) \in [0, 1]$ is the jth evaluation of a uniform random number generator. $rn(i) \in [1, 2, \ldots, D]$ is a randomly chosen index which ensures that $\vec{R}_i(t)$ gets at least one component from $\vec{Z}_i(t)$. To keep the population size constant over subsequent generations, the next step of the algorithm calls for 'selection' in order to determine which one between the target vector and trial vector will survive in the next generation i.e., at time $t = t + 1$. If the trial vector yields a better value of the fitness function, it replaces its target vector in the next generation; otherwise the parent is retained in the population:

$$\left.\begin{array}{ll} \vec{Z}_i(t+1) = \vec{R}_i(t) & \text{if } f(\vec{R}_i(t)) \ge f(\vec{Z}_i(t)) \\ = \vec{Z}_i(t) & \text{if } f(\vec{R}_i(t)) < f(\vec{Z}_i(t)) \end{array}\right\} \quad (16)$$

where $f(\cdot)$ is the function to be maximized.

The algorithm illustrated above is known as DE/rand/1/bin in literature [26]. It belongs to the family of 10 basic DE variants (which mainly differ in their mutation strategies) proposed by Storn and Price. A thorough review of the large volume of work devoted to the improvement of DE over the past few years is out of the scope of this work. We here instead, confine ourselves to a new variant of DE that incorporates two novel parameter tuning strategies to escape from stagnation and/or premature convergence.

To improve the convergence properties of DE we have tuned its parameters in two different ways here. In classical DE the difference vector $(\vec{X}_i(t) - \vec{X}_j(t))$ is scaled by a constant factor 'F'.

Table 1
Algorithm parameters

FVGA		AFDE	
Parameter	Value	Parameter	Value
Pop_size	20	Pop_size	10*dim
Crossover probability, P_c	0.6	R_{max}	1.5
Mutation probability, P_m	0.05	R_{min}	0.5
K_{max}	10	K_{max}	10
K_{min}	2	K_{min}	2

* signifies the multiplication (into) symbol, i.e. the Pop_size is 10 times dimension.

The usual choice for this control parameter is a number between 0.4 and 1. We propose to vary this scale factor in a random manner in the range (0.5, 1) by using the relation

$$F = 0.5(1 + \text{rand}(0, 1)) \quad (17)$$

where rand (0, 1) is a uniformly distributed random number within the range [0, 1]. The mean value of the scale factor is 0.75. This allows for stochastic variations in the amplification of the difference vector and thus helps retain population diversity as the search progresses. In Ref. [38] we have already shown that the DERANDSF (DE with random scale factor) can meet or beat the classical DE and also some versions of PSO in a statistically significant manner. In addition to that, here we also decrease the crossover rate Cr linearly with time from $Cr_{max} = 1.0$ to $Cr_{min} = 0.5$. If Cr = 1.0, it means that all components of the parent vector are replaced by the difference vector operator according to (12). But at the later stages of the optimizing process, if Cr be decreased, more components of the parent vector are then inherited by the

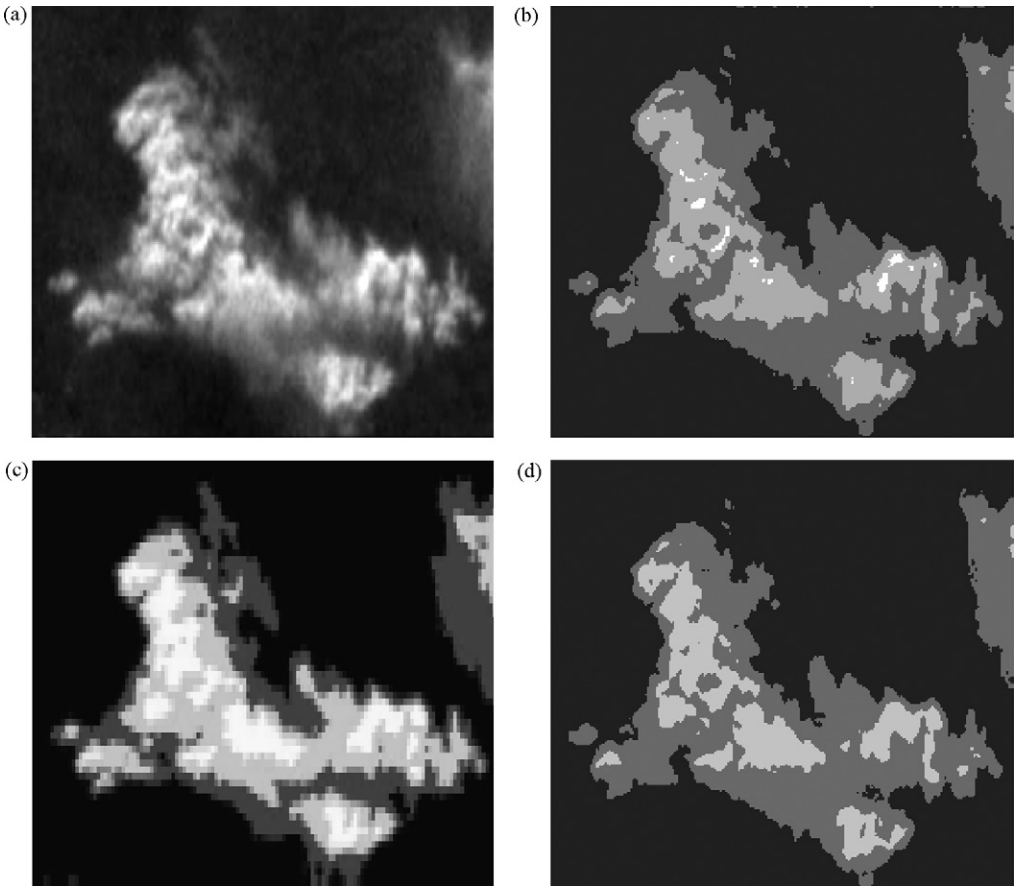


Fig. 1. (a) The original clouds image. (b) Segmentation by FVGA ($c = 3$). (c) Segmentation by AFDE ($c = 4$). (d) Segmentation with FCM (provided with $c = 4$).

offspring. Such a tuning of Cr helps to explore the search space exhaustively at the beginning, but adjust the movements of trial solutions finely during the later stages of search, so that they can explore the interior of a relatively small space in which the suspected global optimum lies. The time-variation of Cr may be expressed in the form of the following equation,

$$\mathrm{Cr} = \mathrm{Cr}_{\max} - (\mathrm{Cr}_{\max} - \mathrm{Cr}_{\min}) \times \frac{\mathrm{iter}}{\mathrm{MAXITER}} \tag{18}$$

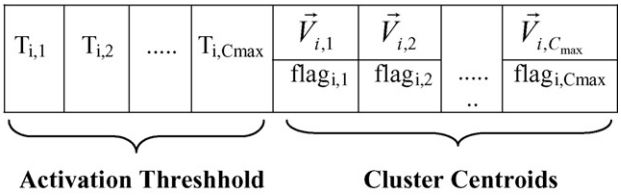
where $\mathrm{Cr}_{\max}$ and $\mathrm{Cr}_{\min}$ are the maximum and minimum values of crossover rate Cr, iter is the current iteration number and MAXIT is the maximum number of allowable iterations.

3.2. Fuzzy clustering algorithm based on DE

3.2.1. Chromosome representation

In the proposed method, for n data points, each d-dimensional, and for a user-specified maximum number of clusters $C_{\max}$, a chromosome is a vector of real numbers of dimension $C_{\max} + C_{\max} \times d$. The first $C_{\max}$ entries are positive floating-point numbers in (0, 1), each of which controls whether the corresponding cluster is to be activated (i.e., to be really used for classifying the data) or not. The remaining entries are reserved for $C_{\max}$ cluster centers, each d-dimensional. A single chromosome can be shown as:

$\vec{Z}_i(t) =$



Every probable cluster center $\vec{V}_{i,j}$ has p features and a binary $\mathrm{flag}_{i,j}$ associated with it. The cluster center is active (i.e., selected for classification) if $\mathrm{flag}_{i,j} = 1$ and inactive if $\mathrm{flag}_{i,j} = 0$. Each flag is set or reset according to the value of the activation threshold $T_{i,j}$. Note that these flags are latent information associated with the cluster centers and do not take part in the DE-type mutation of the chromosomes. The rule for selecting the clusters specified by one chromosome is:

$$\text{IF } T_{i,j} > 0.5 \text{ THEN } \mathrm{flag}_{i,j} = 1 \text{ ELSE } \mathrm{flag}_{i,j} = 0 \tag{19}$$

Note that the flags in an offspring are to be changed only through the T_{ij}'s (according to the above rule). When a new offspring chromosome is created according to (14) and (15), the T values are first obtained which then are used to select (via (19)) the active cluster centroids. If due to mutation some threshold T in an offspring exceeds 1 or becomes negative, it is fixed to 1 or zero,

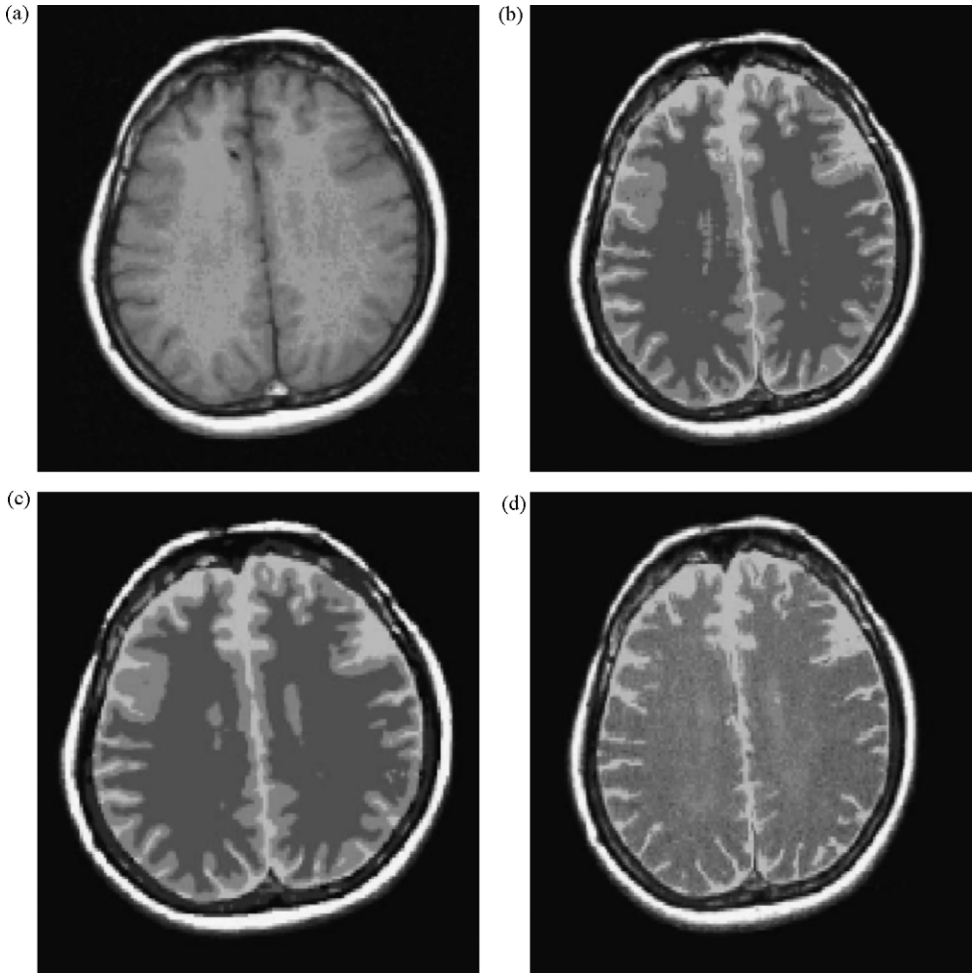


Fig. 2. (a) The original MRI image. (b) Segmentation by FVGA ($c = 5$). (c) Segmentation by AFDE ($c = 5$). (d) Segmentation with FCM (provided with $c = 5$).

respectively. However, if it is found that no flag could be set to one in a chromosome (all activation threshholds are smaller than 0.5), we randomly select two thresholds and re-initialize them to a random value between 0.5 and 1.0. Thus the minimum number of possible clusters is 2.

3.2.2. Fitness function

The quality of a partition can be judged by an appropriate cluster validity index. In the present work we have based our fitness function on the PS measure described in (6) and (7). The fitness function may be written as

$$f = \frac{1}{\mathrm{PS}_i(c) + \mathrm{eps}} \tag{20}$$

where PS_i is the PS measure of the ith particle and eps is a very small constant (we used 0.0002). So maximization of this function means minimization of the PS measure.

3.2.3. Avoiding erroneous chromosomes with empty clusters or unreasonable fitness evaluation

There is a possibility that in our scheme, during computation of the PS measure, a division by zero may be encountered. This may occur when one of the selected cluster centers is outside the boundary of distributions of the dataset. To avoid this problem we first check to see if any cluster has fewer than two data points in it. If so, the cluster center positions of this special chromosome are re-initialized by an average computation. We put n/c data points for every individual cluster center, such that a data point goes with a center that is nearest to it.

3.3. The pseudo-code of the proposed algorithm

The pseudo-code for the complete algorithm for dynamic clustering is given below:

- Step 1: Initialize each chromosome to contain c randomly selected cluster centers and c (randomly chosen) activation thresholds between 0 and 1.
- Step 2: Find out the active cluster centers in each chromosome by evaluating the activation thresholds according to (19) and set or reset the corresponding flag.
- Step 3: For $t = 1$ to $t_{\max}$ do
  - (i) For each data vector $\vec{X}_\mathrm{p}$, calculate its Euclidean distance $d(\vec{X}_\mathrm{p}, \vec{V}_{i,j})$ from all active cluster centers $\vec{V}_{i,j}$ of a parameter vector $\vec{Z}_i$ ($i = 1, 2, \ldots \mathrm{NP}$).
  - (ii) Assign $\vec{X}_\mathrm{p}$ to cluster center $\vec{V}_{i,j}$ such that
    $d(\vec{X}_\mathrm{p}, \vec{V}_{i,j}) = \min_{\forall b \in \{1,2,\ldots,c\}}\{d(\vec{X}_\mathrm{p}, \vec{V}_{i,b})\}$.
  - (iii) Check if the number of data points belonging to any cluster center is less than two. If so, update the cluster centers of the DE vector using the concept of average described in Section 3.2.3.
  - (iv) Now perform mutation on each population member $\vec{Z}_i(t)$ of DE using (14) to form the corresponding donor vectors $\vec{Y}_i(t)$. Exchange body parts of the donor with the target vector $\vec{Z}_i(t)$ according to (15) to form the trial vectors $\vec{R}_i(t)$.
  - (v) Locate the active cluster centers of the trial vectors thus formed, by applying rule (19) and set or reset the associated flags correspondingly.
  - (vi) Repeat steps (i), (ii) and (iii) for each trial vector.

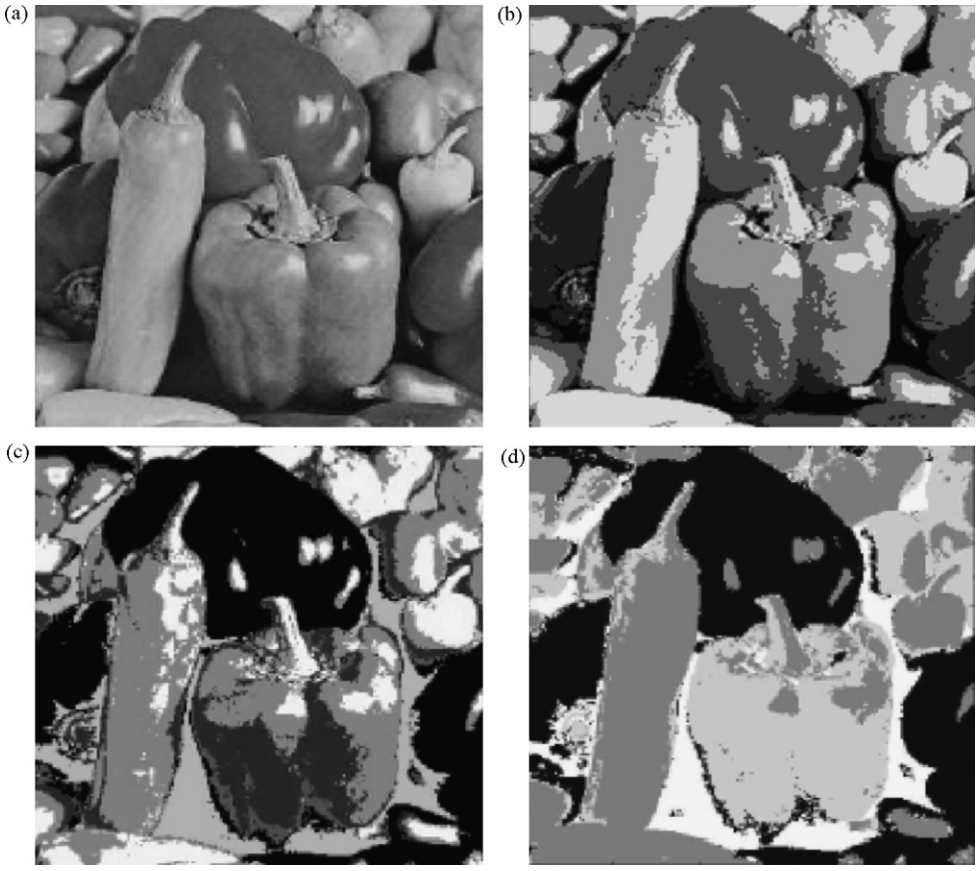


Fig. 3. (a) The original Pepper image. (b) Segmentation by FVGA ($c = 7$). (c) Segmentation by AFDE ($c = 7$). (d) Segmentation with FCM (provided with $c = 7$).

(vii) Evaluate fitness of both the target and trial vectors according to (20). Use only the active cluster centers in both the vectors (that is the ones with the flag set to 1). Replace the target vector $\vec{Z}_i(t)$ with the trial vector $\vec{R}_i(t)$ only if the latter yields a higher value of the fitness function.

- Step 4: Report as the final solution the cluster centers and the partition obtained by the best chromosome (the one yielding the lowest value of the objective function) at time $t = t_{max}$.

From now onwards, we shall refer to the above algorithm as AFDE (automatic fuzzy clustering with DE) algorithm.

4. Experimental setup and results

4.1. The FVGA algorithm

We compared AFDE with the FVGA (fuzzy variable string genetic algorithm) based clustering technique [32,33] in the present work. The FVGA-clustering algorithm tries to determine appropriate number of clusters present in a dataset and the corresponding best partition. Here the chromosomes (or strings) encode the cluster centers as a sequence of real numbers. For example, if the number of clusters is three, then the string will contain these three cluster centers in any arbitrary order. Each string can have a certain maximum length, which is equal to the maximum possible number of clusters C_{max} that may be present in the data. Out of this total maximum number of positions in the string, only some are used to store the cluster centers. The other positions remain do not care (represented by '#' symbol). The value of c is assumed to lie in the range $[C_{min}, C_{max}]$, where C_{min} is chosen to be 2, unless specified otherwise. Note that the choice of C_{max} should not exceed the number of data patterns present in the dataset.

A finite size, P, of the initial population is generated randomly. The strings contain different numbers of cluster centers. If we consider d-dimensional data, and if a string represents c clusters, then there are $c \times d$ random values of real numbers in the string. In the initial population, the cluster centers are selected from within the data range. We have considered a minimum string length to be $C_{min} = 2$, which represents a two-cluster solution. The choice of C_{max} depends on the datasets used. Following the authors, we selected $C_{max} \leq \sqrt{n}$, where, n is number of patterns present in the dataset. Moreover, in the initial population, the positions of the centers within the strings are selected randomly.

Here although the strings are variable in length, they are usually converted to a fixed length one by use of do not care ('#') symbols, i.e., the physical length of all the strings are $C_{max} \times d$. Out of these $C_{max} \times d$ locations only $c \times d$ are used by a string representing c clusters, other locations remain do not care. Therefore, the conventional single point *crossover* can be applied as usual. *Mutation* is performed in the following manner.

If the value at a gene position is v, after mutation it becomes

$$\begin{cases} v \times (1 \pm 2\delta) & \text{if } v \neq 0 \\ \pm 2\delta & \text{if } v = 0 \end{cases}$$

+ or − symbol appear with equal probability. Gene positions represented by '#' (do not care) need not be altered during mutation.

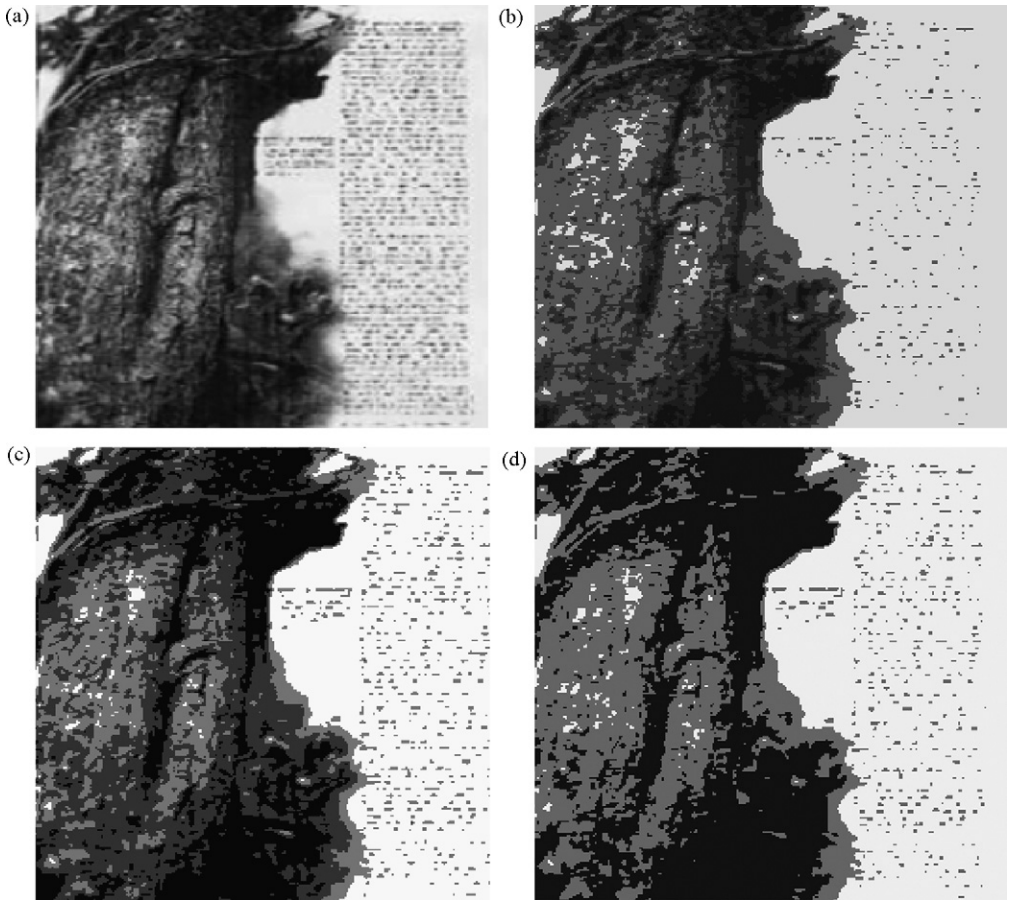


Fig. 4. (a) The original Science Magazine image. (b) Segmentation by FVGA ($c = 3$). (c) Segmentation by AFDE ($c = 4$). (d) Segmentation with FCM (provided with $c = 4$).

4.2. The simulation strategy

We have selected a test suite of six grayscale images among which 'clouds', 'MRI image of brain', 'the pepper image' and 'robot' come in 256 pixels × 256 pixels, while 'the IRS (Indian remote sensing satellite) image of Mumbai (a mega city of India)' and 'the Science Magazine' image are of size 512 × 512. The IRS image of Mumbai was obtained using the LISS-II sensor. It is available in five bands, viz. blue, green, red and near infrared. Fig. 5(a) shows the IRS image of a part of Mumbai in the near infrared band. All the images have been clustered in their intensity space using only pixel intensities as features. The parameter setup is given in Table 1.

All the algorithms have been developed from scratch in Visual C++ on a Pentium IV, 1.2 GHz PC, with 512 kB cache and 2 GB of main memory with Windows Server 2003 environment.

For each image dataset, each run continues until the number of function evaluations (FEs) reaches 100,000. Thirty independent runs (with different seeds for the random number generator) have been taken for each algorithm. The results have been stated in terms of the mean best-of-run values and standard deviations over these 30 runs in each case. Performance comparisons are made on three aspects of the solution: (a) quality of the solution as determined by the three cluster validity indices (described in (5), (7) and (9)), (b) ability to find the optimal number of clusters, and (c) time required to find the solution.

4.3. Results

Figs. 1–6 show the six original images and their segmented counterparts obtained using AFDE, FVGA and the classical FCM. The FCM algorithm cannot handle an unknown number of clusters and has, therefore, in each case been fed with the number of classes yielded by the better between the AFDE and the FVGA. In Figs. 1–6 the segmented portions of an image have been marked with the grey level intensity of the respective cluster centers. Table 2 contains the mean and standard deviations of the number of classes obtained by the two automatic clustering algorithms. In five of the six cases, AFDE yields a better segmentation that FVGA does. In Table 3, we report the mean value of three fuzzy validity measures calculated over the 'best-of-run' solutions in each case. AFDE meets or beats the two competitors in all the cases (for all three indices). Tables 4–6 show results of unpaired *t*-tests performed between AFDE and the better between FVGA and FCM. For all cases in Tables 4–6, sample size = 30 and degrees of freedom = 58. Table 7 reports the mean time taken by each algorithm to terminate on the image data. Since, for all the datasets, the minimum and maximum number of clusters are 2 and 10 respectively, the FCM algorithm is executed nine times (once for each given number of clusters) by supplying to it the number of clusters as 2–10, and the total running time is reported in the last column of Table 7. For all cases of FCM, the algorithm is executed until the difference between the cluster centers in two consecutive iterations falls below 0.00001.

4.4. Discussions on the results

In this study, the proposed AFDE has been compared with one state-of-the-art GA based automatic clustering algorithm and the classical FCM algorithm. To make the performance evaluation/comparison meaningful and effective, we have used variety of test images containing real life images, medical images as well as remote sensing satellite images.

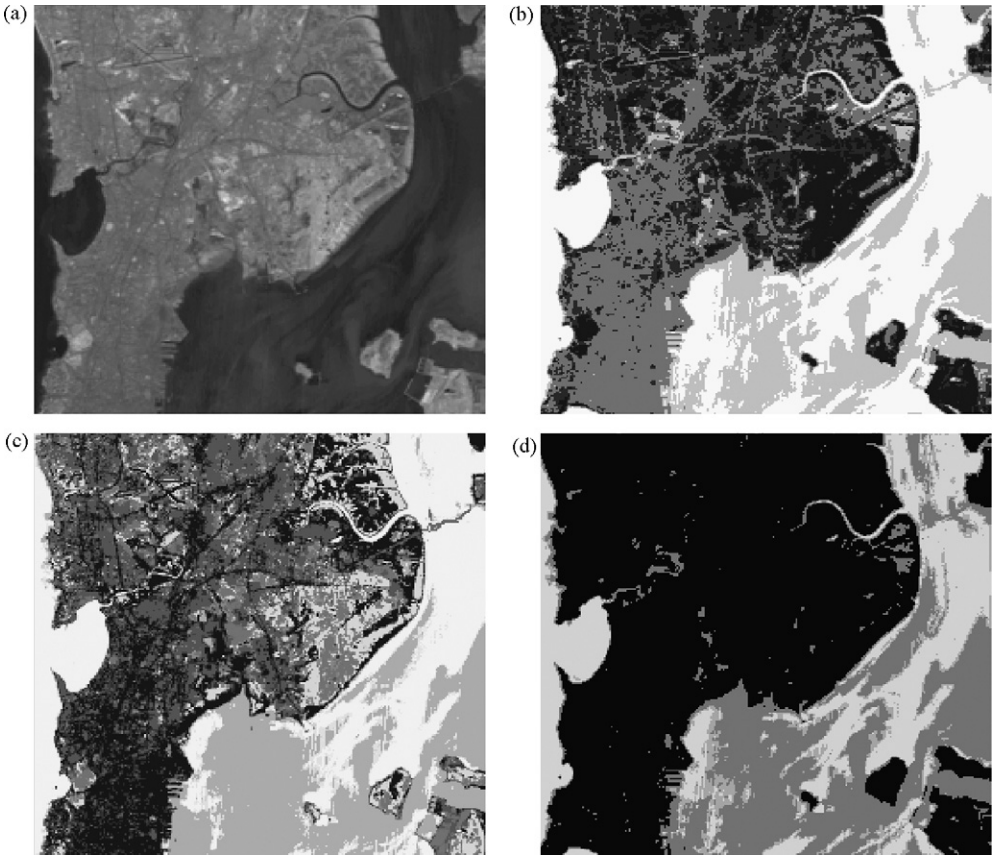


Fig. 5. (a) The original IRS image of Mumbai. (b) Segmentation by FVGA ($c = 5$). (c) Segmentation by AFDE ($c = 6$). (d) Segmentation with FCM (provided with $c = 6$).

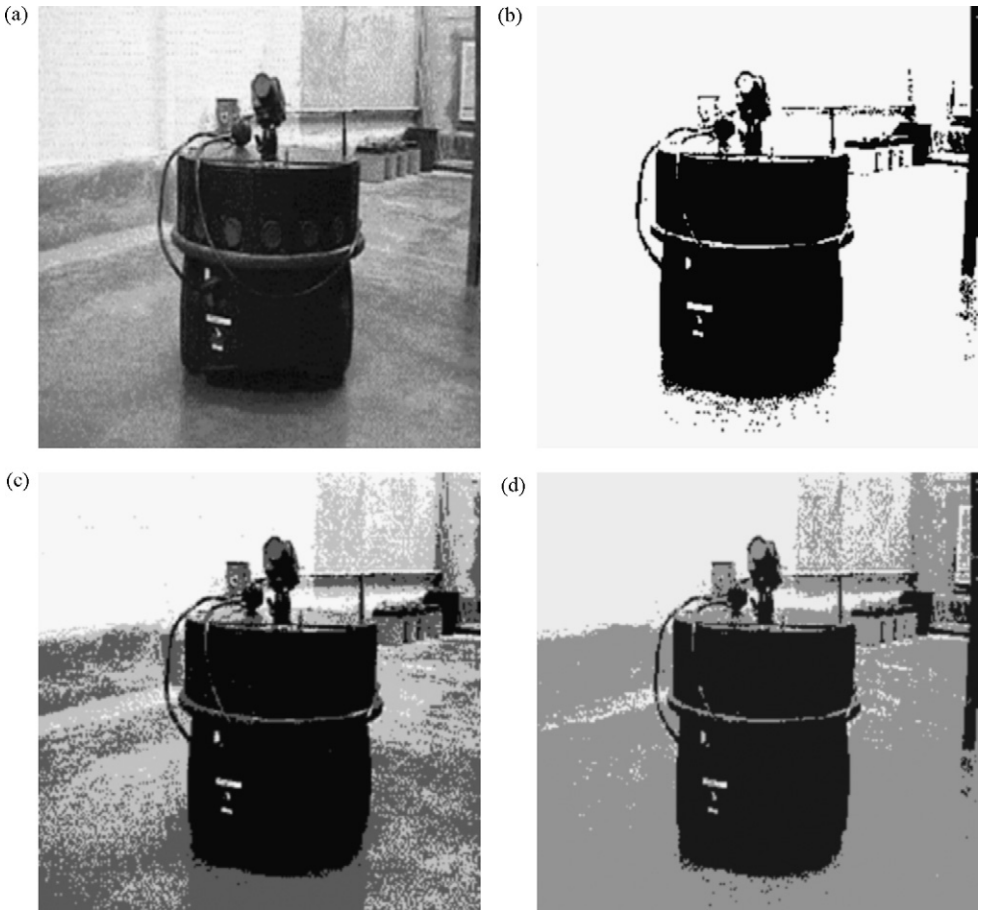


Fig. 6. (a) The original Nomadic Super Scout II Robot image. (b) Segmentation by FVGA ($c = 2$). (c) Segmentation by AFDE ($c = 3$). (d) Segmentation with FCM (provided with $c = 3$).

From Tables 2 and 3 one may observe that our approach outperforms the state-of-the-art FVGA and classical FCM over all the six test images in a statistically significant manner. Not only does the method find the optimal number of clusters, it also manages to find better clustering of the data points for the three major cluster validity indices used in the literature. The performance of the FVGA method may be improved by suitably tuning its parameters and/or trying other validity indices to form its fitness function. This renders itself to further research with FVGA for automatic pixel clustering task.

We used unpaired *t*-tests to compare the means of the results produced by best and the second best algorithms. Unpaired *t*-test assumes that the data has been sampled from a normally distributed population. From the concepts of central limit theorem, one may note that as sample sizes increase, the sampling distribution of the mean approaches a normal distribution regardless of the shape of the original population [39]. From Table 4 we see that for all the test cases, the mean clustering accuracy of FNDE is significantly better than that of the second best competitor i.e., the FVGA as judged by the Xie–Beni index. In three out of these six cases (clouds, MRI brain image and the pepper image) the difference of the means is *extremely significant*. However, Tables 5 and 6 show that the PBMF and PS indices calculated over the final clustering results do not differ significantly for the IRS image of Mumbai. This is perhaps due to the fact that different validity indices judge the cohesiveness and separation between the clusters in different fashions (over the same dataset) [40].

The remote sensing satellite image of Mumbai in Fig. 5 bears a special significance in the present context. Usually segmentation of such images helps in the land cover analysis of different areas in a country. The new method yielded six clusters on average for this

Table 2
Automatic clustering result over six real life grayscale images (over 30 runs; each runs continued for 100,000 FE)

Image	Optimal number of clusters	Mean and standard deviation of the number of classes estimated by the competitor algorithms	
		AFDE	FVGA
Clouds	4	3.25 ± 0.211	4.50 ± 0.132
MRI image of brain	5	5.05 ± 0.428	5.25 ± 0.212
Pepper image	4	4.15 ± 0.772	5.25 ± 0.982
The Science Magazine page	4	4.35 ± 0.038	5.30 ± 0.564
IRS Mumbai image	6	6.20 ± 0.479	4.85 ± 2.489
Robot	3	3.05 ± 0.076	2.25 ± 0.908

Table 3
Automatic clustering result over six real-life grayscale images (over 30 runs; each runs continued for 100,000 FE)

Image	Validity index	Mean and standard deviation of the validity indices over the final clustering results of 20 independent runs		
		AFDE	FVGA	FCM
Clouds	Xie–Beni	0.7283 (0.0001)	0.7902 (0.0948)	0.7937 (0.0013)
	PBMF	2.6631 (0.7018)	2.1193 (0.8826)	2.1085 (0.0043)
	PS-Measure	0.0543 (0.0002)	0.0638 (0.0059)	0.0699 (0.0001)
MRI Image of Brain	Xie–Beni	0.2261 (0.0017)	0.2919 (0.0583)	0.3002 (0.0452)
	PBMF	11.7421 (0.9694)	11.0149 (1.231)	10.9528 (0.0146)
	PS-Measure	0.1837 (0.0017)	0.1922 (0.0096)	0.1939 (0.0921)
Pepper Image	Xie–Beni	0.05612 (0.0092)	0.09673 (0.0043)	0.09819 (0.0001)
	PBMF	28.8078 (0.0819)	28.6083 (0.0971)	28.5831 (0.0038)
	PS-Measure	0.8872 (0.0137)	1.1391 (0.0292)	1.1398 (0.0884)
The Science Magazine Page	Xie–Beni	0.2818 (0.0112)	0.3137 (0.0191)	0.3232 (0.0043)
	PBMF	5.2943 (0.0492)	5.0147 (0.0339)	4.9052 (0.0032)
	PS-Measure	0.0122 (0.0022)	0.0153 (0.0039)	0.0191 (0.0081)
IRS Mumbai Image	Xie–Beni	0.1718 (0.0112)	0.1943 (0.0391)	0.2032 (0.0029)
	PBMF	4.2492 (0.0386)	4.0198 (0.904)	4.0052 (0.061)
	PS-Measure	0.0564 (0.0083)	0.0658 (0.0109)	0.0791 (0.0051)
Robot	Xie–Beni	0.0105 (0.0074)	0.01692 (0.0094)	0.01739 (0.0009)
	PBMF	12.3421 (0.0842)	9.6739 (1.032)	8.4528 (0.0043)
	PS-Measure	0.0396 (0.0017)	0.0422 (0.0095)	0.0439 (0.0021)

Table 4
Unpaired *t*-test results for Xie–Beni index

Dataset	Standard error	*t*	95% confidence interval	Two-tailed *P*	Significance
Clouds	0.001	7.1968	−0.0121 to −0.0068	<0.0001	Extremely significant
MRI image of brain	0.002	3.8990	−0.0129 to −0.0040	<0.0001	Extremely significant
Pepper image	0.007	34.9267	−0.2665 to −0.2373	<0.0001	Extremely significant
The Science Magazine	0.001	3.0961	−0.0051 to −0.0010	0.0037	Very significant
IRS Mumbai image	0.003	3.0684	−0.0156 to −0.0032	0.0040	Very significant
Robot	0.002	3.0584	−0.0109 to −0.0022	0.0041	Very significant

image. A close inspection of Fig. 5(c) reveals that, most of the land cover categories have been correctly distinguished in this image. For example, the *Santa Cruz* airport, dockyard, bridge connecting Mumbai to New Mumbai and many other road structures have come out distinctly. In addition, the predominance of one category of pixels in the southern part of the image conforms to the ground truth; this part is known to be heavily industrialized, and hence the majority of the pixels in this region should belong to the same class of concrete. The *Arabian Sea* has come out as a combination of pixels of two different classes. The seawater is found to be decomposed into two classes, turbid water 1 and turbid water 2, based on the difference of their reflectance properties.

Table 7 indicates that our algorithm has a worse run-time than FCM, but clearly this is an unfair comparison as the FCM has to be fed with the correct number of clusters, whereas the proposed method finds it itself. It can be further observed that the AFDE

Table 5
Unpaired *t*-test Results for the PBMF Index

Image	Standard error	*t*	95% confidence interval	Two-tailed *P*	Significance
Clouds	0.021	2.9201	−0.1050 to −0.0189	0.0059	Very significant
MRI image of brain	0.013	5.0453	−0.0922 to −0.0394	<0.0001	Extremely significant
Pepper image	0.002	17.965	−0.0452 to −0.0360	<0.0001	Extremely significant
The Science Magazine	0.005	6.4431	−0.0419 to −0.0219	<0.0001	Extremely significant
IRS Mumbai image	0.009	1.3744	−0.0309 to 0.0059	0.1774	Not significant
Robot	0.003	2.3999	−0.0118 to −0.0010	0.0214	Significant

Table 6
Unpaired *t*-test results for PS-measure index

Image	Standard error	*t*	95% confidence interval	Two-tailed *P*	Significance
Clouds	0.021	2.9201	−0.1050 to −0.0189	0.0059	Very significant
MRI image of brain	0.013	5.0453	−0.0922 to −0.0394	<0.0001	Extremely significant
Pepper image	0.002	17.965	−0.0452 to −0.0360	<0.0001	Extremely significant
The Science Magazine page	0.005	6.4431	−0.0419 to −0.0219	<0.0001	Extremely significant
IRS Mumbai image	0.009	1.3744	−0.0309 to 0.0059	0.1774	Not significant
Robot	0.003	2.3999	−0.0118 to −0.0010	0.0214	Significant

Table 7
The mean execution time taken in seconds and standard deviations (in parentheses) by the competitor algorithms over 30 independent runs

Image	Mean and standard deviation of the execution time taken by the competitor algorithms			
	AFDE	FVGA	FCM fed with correct number of clusters	FCM – total time for 2–10 clusters
Clouds	32.05 ± 0.076	47.25 ± 0.162	4.35 ± 0.001	52.25 ± 0.018
MRI image of brain	24.15 ± 0.016	34.65 ± 0.029	12.35 ± 0.001	98.50 ± 0.018
Pepper image	49.20 ± 0.201	67.85 ± 0.817	3.75 ± 0.002	38.45 ± 0.002
The Science Magazine page	84.55 ± 0.336	107.50 ± 0.49	18.35 ± 0.001	144.20 ± 0.03
IRS Mumbai image	93.00 ± 0.006	177.50 ± 0.16	24.60 ± 0.001	197.25 ± 0.21
Robot	19.50 ± 0.032	27.95 ± 0.342	2.90 ± 0.006	24.65 ± 0.117

consumes lesser computational time on average than the FVGA for six images and the one possible reason of this may be the use of less complicated variation operators (like differential mutation) in DE as compared to the operators used for GA.

5. Conclusion

This paper has presented a new, differential evolution-based strategy for fuzzy clustering of images. An important feature of the proposed algorithm is that it is able to find the optimal number of clusters automatically (that is, the number of clusters does not have to known in advance). Experimental results show that our approach outperforms the state-of-the-art FVGA strategy and the classic FCM over a variety of image datasets in a statistically significant manner.

Future research may focus on employing other improved cluster validity indices to form the fitness function. A combination of different validity indices may also be used to form the fitness function and a multi-objective DE [41]. Besides the pixel intensity alone, it may be interesting to take into account other features related to texture, shape and color for the segmentation task by AFDE.

References

[1] A.K. Jain, M.N. Murty, P.J. Flynn, Data clustering, a review, ACM Comput. Surv. 31 (3) (1999) 264–323.
[2] J.T. Tou, R.C. Gonzalez, Pattern Recognition Principles, Addison-Wesley, London, 1974.
[3] M.M. Trivedi, J.C. Bezdek. Low-level segmentation of aerial images with fuzzy clustering, IEEE Trans. Syst. Man and Cybernetics SMC-16, pp. 589–598, 1986.
[4] G. Ball, D. Hall, A clustering technique for summarizing multivariate data, Behav. Sci. 12 (1967) 153–155.
[5] C.S. Wallace, D.M. Boulton, An information measure for classification, Comput. J. 11 (2) (1968) 185–194.
[6] H. Frigui, R. Krishnapuram, A robust competitive clustering algorithm with applications in computer vision, IEEE Trans. Pattern Anal. Machine Intell. 21 (5) (1999) 450–465.
[7] Y. Leung, J. Zhang, Z. Xu, Clustering by space-space filtering, IEEE Trans. Pattern Anal. Machine Intell. 22 (12) (2000) 1396–1410.
[8] J.C. Bezdek, Pattern Recognition with Fuzzy Objective Function Algorithms, Plenum, New York, 1981.
[9] L.O. Hall, I.B. Özyurt, J.C. Bezdek, Clustering with a genetically optimized approach, IEEE Trans. Evol. Comput. 3 (2) (1999) 103–112.
[10] I. Gath, A. Geva, Unsupervised optimal fuzzy clustering, IEEE Trans. PAMI 11 (1989) 773–781.
[11] A.M. Bensaid, L.O. Hall, J.C. Bezdek, L.P. Clarke, Partially supervised clustering for image segmentation, Pattern Recognit. 29 (1996) 859–871.
[12] M.C. Clark, L.O. Hall, D.B. Goldgof, L.P. Clarke, R.P. Velthuizen, M.S. Silbiger, MRI segmentation using fuzzy clustering techniques, IEEE Eng. Med. Biol. 13 (1994) 730–742.
[13] M.N. Ahmed, S.M. Yamany, N. Mohamed, A.A. Farag, T. Moriarty, A modified fuzzy c-means algorithm for bias field estimation and segmentation of MRI data, IEEE Trans. Med. Imaging 21 (2002) 193–199.
[14] X. Wang, Y. Wang, L. Wang, Improving fuzzy c-means clustering based on feature-weight learning, Pattern Recognit. Lett. 25 (2004) 1123–1132.
[15] W. Pedrycz, J. Waletzky, Fuzzy clustering with partial supervision, IEEE Trans. Syst. Man Cybernetics, Part B, Cybernetics 27 (1997) 787–795.
[16] M. Halkidi, Y. Batistakis, M. Vazirgiannis, On clustering validation techniques, J. Intell. Inform. Syst. (JIIS) 17 (2–3) (2001) 107–145.
[17] S. Theodoridis, K. Koutroubas, Pattern Recognition, Academic Press, 1999.
[18] C. Rosenberger, K. Chehdi, Unsupervised clustering method with optimal estimation of the number of clusters: application to image segmentation, in: Proceedings of the IEEE International Conference on Pattern Recognition (ICPR), vol. 1, Barcelona, September, (2000), pp. 1656–1659.
[19] H.-P. Schwefel, Evolution and Optimum Seeking, first ed., Wiley, New York, NY, 1995.
[20] L.J. Fogel, A.J. Owens, M.J. Walsh, Artificial Intelligence Through Simulated Evolution, Wiley, New York, 1966.
[21] J.H. Holland, Adaptation in Natural and Artificial Systems, University of Michigan Press, Ann Harbor, 1975.
[22] C.-Y. Lee, E.K. Antonsson, in: M. Laudon, B. Romanowicz (Eds.), Self-Adapting Vertices for Mask Layout Synthesis Modeling and Simulation of Microsystems Conference, San Diego, March 27–29, (2000), pp. 83–86.
[23] M. Sarkar, B. Yegnanarayana, D. Khemani, A clustering algorithm using an evolutionary programming-based approach, Pattern Recognit. Lett. 18 (1997) 975–986.
[24] S. Bandyopadhyay, U. Maulik, Genetic clustering for automatic evolution of clusters and application to image classification, Pattern Recognit. 35 (2002) 1197–1208.
[25] J. Kennedy, R. Eberhart, Particle swarm optimization, in: Proceedings of IEEE International Conference on Neural Networks, IEEE Service Center, Piscataway, NJ, (1995), pp. 1942–1948.
[26] K. Price, R. Storn, J. Lampinen, Differential Evolution – A Practical Approach to Global Optimization, Springer, Berlin, 2005.
[27] M. Omran, A. Engelbrecht, A. Salman, Particle swarm optimization method for image clustering, Int. J. Pattern Recognit. Artificial Intell. 19 (3) (2005) 297–322.
[28] M. Omran, A. Salman, A. Engelbrecht, Dynamic clustering using particle swarm optimization with application in unsupervised image classification, in: Fifth World Informatika Conference (ICCI 2005), Prague, Czech Republic, 2005.
[29] J. Kennedy, R.C. Eberhart, A discrete binary version of the particle swarm algorithm, in: Proceedings of the 1997 Conference on Systems, Man, and Cybernetics, IEEE Service Center, Piscataway, NJ, 1997, pp. 4104–4109.
[30] M. Omran, A.P. Engelbrecht, A. Salman, Differential Evolution Methods for Unsupervised Image Classification, Proceedings of the Seventh Congress on Evolutionary Computation (CEC-2005), Edinburgh, Scotland, IEEE Press, 2005.
[31] M.-C. Su, C.-H. Chou, A modified version of the K-means algorithm with a distance based on cluster symmetry, IEEE Trans. Pattern Anal. Machine Intell. 23 (6) (2001) 674–680.
[32] M.K. Pakhira, S. Bandyopadhyay, U. Maulik, A study of some fuzzy cluster validity indices, genetic clustering and application to pixel classification, Fuzzy Sets Syst. 155 (2005) 191–214.
[33] U. Maulik, S. Bandyopadhyay, Fuzzy partitioning using a real-coded variable length genetic algorithm for pixel classification, In: IEEE Trans. Geosci. Remote Sensing, vol. 41, No. 5, pp. 1075–1081, 2003.
[34] A. Konar, Computational Intelligence: Principles, Techniques and Applications, Springer, 2005.
[35] X. Xie, G. Beni, Validity measure for fuzzy clustering, IEEE Trans. Pattern Anal. Machine Learn. 3 (1991) 841–846.
[36] M.-C. Su, C.-H. Chou, E. Lai, A new cluster validity measure for clusters with different densities, in: IASTED International Conference on Intelligent Systems & Control, Salzburg, Austria, (2003), pp. 276–281.
[37] M.-C. Su, C.-H. Chou, A competitive learning algorithm using symmetry, IEICE Trans. Fundam. Electron., Commun. Comput. Sci. E82-A (4) (1999) 680–687.
[38] S. Das, A. Konar, U.K. Chakraborty, Two improved differential evolution schemes for faster global search, in: ACM-SIGEVO, Proceedings of Genetic and Evolutionary Computation Conference (GECCO-2005), Washington, DC, June, 2005.
[39] B. Flury, A First Course in Multivariate Statistics, vol. 28, Springer, 1997.
[40] N.R. Pal, S.K. Pal, A review on image segmentation techniques, Pattern Recognit. 26 (1993) 1277–1294.
[41] H.A. Abbass, R. Sarker, C. Newton, PDE: a Pareto-frontier differential evolution approach for multi-objective optimization problems, in: Proceedings of IEEE Congress on Evolutionary Computation (CEC 2001), IEEE Press, (2001), pp. 971–978.