Evolving a Fuzzy Rule-Base for Image Segmentation

A. Borji, and M. Hamidi

Abstract—A new method for color image segmentation using fuzzy logic is proposed in this paper. Our aim here is to automatically produce a fuzzy system for color classification and image segmentation with least number of rules and minimum error rate. Particle swarm optimization is a sub class of evolutionary algorithms that has been inspired from social behavior of fishes, bees, birds, etc, that live together in colonies. We use comprehensive learning particle swarm optimization (CLPSO) technique to find optimal fuzzy rules and membership functions because it discourages premature convergence. Here each particle of the swarm codes a set of fuzzy rules. During evolution, a population member tries to maximize a fitness criterion which is here high classification rate and small number of rules. Finally, particle with the highest fitness value is selected as the best set of fuzzy rules for image segmentation. Our results, using this method for soccer field image segmentation in Robocup contests shows 89% performance. Less computational load is needed when using this method compared with other methods like ANFIS, because it generates a smaller number of fuzzy rules. Large train dataset and its variety, makes the proposed method invariant to illumination noise.

Keywords—Comprehensive learning Particle Swarm optimization, fuzzy classification.

I. INTRODUCTION

The process of partitioning a digital image into multiple regions (sets of pixels) is called image segmentation. Actually, partitions are different objects in image which have the same texture or color. The result of image segmentation is a set of regions that collectively cover the entire image, or a set of contours extracted from the image. All of the pixels in a region are similar with respect to some characteristic or computed property, such as color, intensity, or texture. Adjacent regions are significantly different with respect to the same characteristics. Some of practical applications of image segmentation are: image processing, computer vision, face recognition, medical imaging, digital libraries, image and video retrieval, etc [1].

Image segmentation methods fall into five categories: Pixel based segmentation [2], Region based segmentation [3], Edge based segmentation [4], [5], Edge and region Hybrid segmentation [6] and Clustering based segmentation [7], [8], [9], [10], [11]. Color image segmentation using fuzzy classification is a pixel based segmentation method. A pixel is assigned a specific color by the fuzzy system. One approach in designing such a fuzzy system is an expert to look at training data and try to manually develop a set of fuzzy rules. Two drawbacks with such method are that first it is very cumbersome and time consuming and second there is no guarantee that the produced fuzzy rules are the best possible ones. So, there is need for a method which could produce fuzzy rules and membership functions automatically. Several methods have been proposed in literature for automatic production of fuzzy rules like genetic algorithms and ANFIS [12], [13], [14], [15]. One problem with these methods is that they generate a large number of fuzzy rules which causes slow classification and processing speed [8]. In this paper, we propose a method which produces a smaller number of rules while preserving low error rate. To that aim we use CLPSO to search for a set of such fuzzy rules.

The rest of this paper is organized as follows: in section two, fuzzy color classification is explained. An introduction to comprehensive learning particle swarm optimization is illustrated in section three. In section four we give the details of our proposed method for fuzzy color image segmentation using CLPSO. Our experimental setup and results are shown in section five. Finally, conclusions and discussions come in section six.

II. FUZZY COLOR CLASSIFICATION

Fuzzy color classification is a supervised learning method for segmentation of color images. This method assigns a color class to each pixel of an input image by applying a set of fuzzy rules on it. A set of training image pixels, for which the colors are known are used to train the fuzzy system. The trained fuzzy system will be later evaluated on test images. Different color spaces like HSL, RGB, YIQ, etc, have been suggested in image processing, each suitable for different domains [1]. In this study, we chose HSL color space because a color in this space is represented in three dimensions: one which codes the color itself (H) and another two which explain details of the color, saturation (S) and lightness (L). Because of this segregation of color components, this color space is most suitable for our purpose [1].

As it can be seen in Fig. 1, H dimension is shown in a circle with colors occupying a range of degrees around it. Instead of assigning a specific hue value to each color around this circle, a fuzzy membership function can code for a color by giving it a range of hues each with different membership value. As an
example, H dimension in Fig. 2 is partitioned into ten trapezoidal membership functions each one coding a different color.

In Fig. 3, trapezoidal membership function showed needs four parameters to be specified [1].

To represent remaining two dimensions of a color, because of their less importance for determining a color compared with Hue dimension, we divide each dimension to three parts: weak, medium and strong. Combining these two dimensions we come to nine regions for representing a color shown in Fig. 4.

A two dimensional membership function is then placed over each dimension and then by multiplying these functions a set of nine 2D membership functions is generated. Following figure illustrates above concept.

Each fuzzy rule is represented as follows:

\[ f-\text{th rule}: \]
if \( x_1 \) is \( A_{j1} \) and \( x_2 \) is \( A_{j2} \) and ... and \( x_n \) is \( A_{jn} \)
then \( x=(x_1,x_2,...,x_n) \) belongs to class \( H_j \) with \( CF_j = CF_j \quad j=1,2,...,R \)
in which \( R \) is the number of fuzzy rules, \( m \) is the dimensionality of input vector, \( H_j \in \{1,2,...,M\} \) is output of the \( j \)th rule, \( M \) is the number of color classes, \( CF_j \in [0,1] \) is the certainty factor of \( j \)th rule. We tried three types of membership functions in our experiments. Equation (2), gives an example of how we code membership functions on a Gaussian membership function.

\[
\mu_{ji}(x) = \begin{cases} 
\exp\left(-\frac{x - m_{ji}(1)}{m_{ji}(2)}^2\right) & \text{if } x \leq m_{ji}(1) \\
\exp\left(-\frac{x - m_{ji}(1)}{m_{ji}(2)}^2\right) & \text{if } x > m_{ji}(1) 
\end{cases}
\]

(2)

in above equation, \( m_{ji} = [m_{j1}, m_{j2}, ..., m_{jP}] \) are the parameters of the \( j \)th membership function of \( j \)th rule, \( P \) is the number of parameters. \( \Gamma_j = [\mu_{j1}, \mu_{j2}, ..., \mu_{jM}] \) is the \( j \)th rule and \( \Gamma = [\Gamma_1, \Gamma_2, ..., \Gamma_R] \) represents the whole fuzzy rule base. \( a = [H_1, CF_1, H_2, CF_2, ..., H_m, CF_m] \) is the output of the fuzzy rule base. When an input vector \( x=(x_1, x_2, ..., x_m) \) is presented to the system, output of the system is calculated as:

\[
y = \arg \max_{j=1}^{R} q_j(x).CF_j
\]

(3)

Classification performance of a fuzzy system is very much dependent on its parameters. We are going to use a global
Particle Swarm Algorithm (PSO) is a recently proposed algorithm by James Kennedy and R. C. Eberhart in 1995, motivated by the social behavior of organisms such as birds flocking and fish schooling [17, 18]. PSO as an optimization tool provides a population-based search procedure in which individuals called particles change their position (state) with time [19]. In PSO, individuals represent points in the n-dimensional search space [20]. A particle represents a potential solution. The velocity of any one particle. CLPSO ensures that the diversity of the swarm is preserved to discourage premature convergence. In this new learning strategy, the following velocity updating equation is used:

\[ V_i^d \leftarrow \omega \cdot V_i^d + c_1 \cdot rand(0,1) \cdot (pbest_i^d - X_i^d) + c_2 \cdot rand(0,1) \cdot (gbest_i^d - X_i^d) \]

(6)

Where \( f_j = [f_j(0), f_j(2), ..., f_j(D)] \) defines which particles’ pbest the particle \( i \) should follow. pbest\(_{f_j(i)}\) can be the corresponding dimension of any particle’s including its own pbest, and the decision depends on probability \( P_c \), referred to learning probability, which can take different values for different particles. For each dimension of particle \( i \), we generate a random number. If this random number is larger than \( P_c \), the corresponding dimension will learn from its own pbest; otherwise it will learn from another particle’s pbest. We employ the tournament selection procedure when the particle’s dimension learns from another particle’s pbest as follows:

1) Randomly choose two particles out of the population which excludes the particle whose velocity is updated.
2) Compare the fitness values of these two particles’ pbests and select the better one.
3) Use the winner’s pbest as the exemplar to learn from for that dimension. If all exemplars of a particle are its own pbest, we will randomly choose one dimension to learn from another particle’s pbest’s corresponding dimension. All these pbest\(_{j}\) can generate new positions in the search space using the information derived from different particles’ historical best positions [24].
exists an element $g^h_i$. If $g^h_i \geq 0.5$ then the corresponding rule is considered eligible for adding to the rule base. Let $r_h$ be the number of acceptable rules, then index of these rules and their representation are: $r^i_h \in \{1,2,...,B\}$, $r = 1,2,...,r_h,$ 
\[ \{\xi^1_h, \xi^2_h, ..., \xi^r_h\} \] respectively.

Consequently, the rule base of the generated fuzzy classification system is described as follows:

\[ \text{r-th rule:} \]
\[ \text{if } x_i \text{ is } A^i_{\gamma} \text{ and } x_j \text{ is } A^j_{\nu} \text{ and ... and } x_n \text{ is } A^n_{\mu} \]  
\[ \text{then } \Sigma = (x_1, x_2, ..., x_n) \text{ belongs to class } H_r, \]
\[ \text{with } CF = CF_r \quad r = 1,2,...,r_h \]

where $A^i_{\gamma}$, $i = 1,2,...,m$ are the fuzzy sets of the rth generated fuzzy rule.

To completely define each rule, certainty factor CF and output H must be calculated [2]. To that purpose, we use a set of training data. There are N training data patterns, each belonging to one of M colors. Each training pattern is represented by a vector $\langle x_i, y_i \rangle$, $n = 1,2,...,N$, in which $x_i = (x_{i1}, x_{i2}, ..., x_{in})$ and $y_i \in \{1,2,...,M\}$ are input and output of rth sample respectively.

For the rth rule, CF and H are determined as follows:

1) $\theta_i = \sum_{t=1}^{M} q_t(x_i), t = 1,2,...,M$ 
\[ \theta_i = \sum_{t=1}^{M} q_t(x_i), t = 1,2,...,M \]  

2) $H_r = \arg \max_{i=1}^{M} \theta_i$ 
\[ H_r = \arg \max_{i=1}^{M} \theta_i \]

3) Determine the grade of certainty $CF_r$ of the rth fuzzy rule by:
\[ CF_r = \frac{\theta_i - \theta}{\sum_{i=1}^{M} \theta_i} \]  
\[ CF_r = \frac{\theta_i - \theta}{\sum_{i=1}^{M} \theta_i} \]

where, $\theta = \sum_{i=1}^{M} \theta_i$ 
\[ \theta = \sum_{i=1}^{M} \theta_i \]

Fitness of each particle should satisfy two criteria when selecting rule bases. A rule base is better if it has the less number of rules and minimum error rate. An example of such a fitness function is:
\[ f_h = \text{fit}(p_h) = g_1(p_h)g_2(p_h) \]  
\[ f_h = \text{fit}(p_h) = g_1(p_h)g_2(p_h) \]

in above equations, functions $g_1$ and $g_2$ serve for the first and second criteria correspondingly.

\[ g_1(p_h) = \exp \left( \frac{-\text{NICP}(p_h)}{\sigma_r} \right) \]  
\[ g_1(p_h) = \exp \left( \frac{-\text{NICP}(p_h)}{\sigma_r} \right) \]

\[ g_2(p_h) = \exp \left( \frac{-r_h}{\sigma_r} \right) \]  
\[ g_2(p_h) = \exp \left( \frac{-r_h}{\sigma_r} \right) \]

$\text{NICP}(p_h)$ is the number of false classified patterns. Gradually, in the course of running algorithm, particles try to find rule bases which maximize the fitness function. To put all things together, we describe the method in an algorithmic step-wise manner as follows:

Step 1) Initialize the CLPSO-based method. General parameters of the algorithm are listed in Table I.

<table>
<thead>
<tr>
<th>Table I Method Parameters</th>
</tr>
</thead>
<tbody>
<tr>
<td>Parameter name</td>
</tr>
<tr>
<td>Population size</td>
</tr>
<tr>
<td>Max number of rules</td>
</tr>
<tr>
<td>Max number of iterations</td>
</tr>
<tr>
<td>Constants of fitness function</td>
</tr>
<tr>
<td>Constants of CLPSO</td>
</tr>
</tbody>
</table>

Each member of the population $p_h = [r_h g_h]$ is randomly initialized as follows:
\[ r_h = [m^i_1 m^i_2 ... m^i_M m^j_1 m^j_2 ... m^j_M] \]  
\[ r_h = [m^i_1 m^i_2 ... m^i_M m^j_1 m^j_2 ... m^j_M] \]
\[ m^i_1 m^i_2 ... m^i_M \text{ and } m^j_1 m^j_2 ... m^j_M \]  
\[ m^i_1 m^i_2 ... m^i_M \text{ and } m^j_1 m^j_2 ... m^j_M \]
\[ g_h = [g^h_{i1} g^h_{i2} ... g^h_{iM} g^h_{j1} g^h_{j2} ... g^h_{jM}].m^{j}_{i,k}, j \in \{1,2,...,B\}, i \in \{1,2,...,M\}, k \in \{1,2,3\} \]
\[ g_h = [g^h_{i1} g^h_{i2} ... g^h_{iM} g^h_{j1} g^h_{j2} ... g^h_{jM}].m^{j}_{i,k}, j \in \{1,2,...,B\}, i \in \{1,2,...,M\}, k \in \{1,2,3\} \]
\[ \text{is randomly generated as follows:} \]
\[ m^{j}_{i,k} = m^{\text{min}}_{i,k} + (m^{\text{max}}_{i,k} - m^{\text{min}}_{i,k}) \text{rand}() \]  
\[ m^{j}_{i,k} = m^{\text{min}}_{i,k} + (m^{\text{max}}_{i,k} - m^{\text{min}}_{i,k}) \text{rand}() \]
\[ \text{where the range of the parameter } m^{j}_{i,k} \text{ is defined as} \]
\[ [m^{\text{min}}_{i,k} m^{\text{max}}_{i,k}] \text{which is determined from the minimum and maximum of membership functions in each dimension.} \]
\[ g^h_i = \text{rand}(). \]  
\[ g^h_i = \text{rand}(). \]

Velocity vector $v_h = [\alpha^i_h \beta^i_h], h = 1,2,...,L$ of a particle is initialized randomly as:
\[ v_h = [\alpha^i_h \beta^i_h] \]
\[ v_h = [\alpha^i_h \beta^i_h] \]
\[ \alpha_h = [\alpha^i_{11} \alpha^i_{12} ... \alpha^i_{1M} \alpha^i_{21} \alpha^i_{22} ... \alpha^i_{2M} ... \alpha^i_{BM}] \]
\[ \alpha_h = [\alpha^i_{11} \alpha^i_{12} ... \alpha^i_{1M} \alpha^i_{21} \alpha^i_{22} ... \alpha^i_{2M} ... \alpha^i_{BM}] \]
\[ \alpha^i_h = [\alpha^i_{11h} \alpha^i_{21h} ... \alpha^i_{BMh}] \]
\[ \alpha^i_h = [\alpha^i_{11h} \alpha^i_{21h} ... \alpha^i_{BMh}] \]
\[ \beta_h = [\beta^i_{1h} \beta^i_{2h} ... \beta^i_{BMh}] \]
\[ \beta_h = [\beta^i_{1h} \beta^i_{2h} ... \beta^i_{BMh}] \]
\[ \text{in above formulas } \alpha^i_h, \beta^i_h \text{ are calculated as follows:} \]
\[ \alpha^i_{(j,k)} = m^{\text{max}}_{(j,k)} - m^{\text{min}}_{(j,k)} \text{rand}(\) \]
\[ \alpha^i_{(j,k)} = m^{\text{max}}_{(j,k)} - m^{\text{min}}_{(j,k)} \text{rand}(\) \]
\[ \beta^i_h = \text{rand}(\) \]
\[ \beta^i_h = \text{rand}(\) \]

Step 2) Select particle’s neighbors. For each particle generate $f_i = \{f_j(1), f_j(2), ... f_j(D)\}$ as follows: Select two particles $\Omega^{\text{th}}_1$ and $\Omega^{\text{th}}_2$ randomly. If $\text{Fitness}[\text{pbest}(\Omega^{\text{th}}_1)] > \text{Fitness}[\text{pbest}(\Omega^{\text{th}}_2)]$ then $f(d) = \Omega^{\text{th}}_1$ else $f(d) = \Omega^{\text{th}}_2$.
Step 3) Update $g_h = [g_1^h, g_2^h, ..., g_j^h, ..., g_B^h]$:

\[
\text{if } \psi \geq \text{rand}() \text{ then } \\
g_j^h = 1 - g_j^h, j^* = \text{rand}(B \text{rand}() + 0.5) \\
\]

Step 4) Update $pbests$ and $gbest$. If $\text{Fitness}(X_i) > \text{Fitness}(pbest_i)$ then $pbest_i = X_i$. If $\text{Fitness}(X_i) > \text{Fitness}(gbest_i)$ then $gbest_i = X_i$.

Step 5) Update velocity and position of each particle according to (6) and (5).

Step 6) Decay Velocities:

\[
v_a = v_a, d_1, d_2 = [0, 1] \\
\psi = \psi + d_2, d_2 = [0, 1] \\
\]

Step 7) Check Stopping Criteria:

\[
gen = gen + 1 \\
\]

if $\text{gen} > K$ or the average velocity of particles surpasses a small threshold near zero then go to next step, else go to step 3.

Step 8) Stop: Report particle with the best fitness value as best rule base.

For a detailed explanation of CLPSO algorithm, reader is referred to [25].

V. EXPERIMENTS AND RESULTS

In this section, details of our implementation and experimental results are presented. To evaluate the efficiency of this method we conducted experiments on segmentation of color images taken from a Robocop football field which is a benchmark problem in robotic contests.

A. Training Data

Color images taken from a small size Robocop soccer field are used as training data. Because we were to build a segmentation system invariant to illumination, we tried to choose images taken at different lightening situations. Each training point belongs to one of ten color classes (Red1, Orange, Yellow, Green, Cyan, Blue, Purple, Magenta, Pink, Red2). (Red1 and Red2 correspond to first and last membership function in Hue dimension). A total number of 9,000 training data and 3,000 test data were used. From 1,200 data points available for each class, 900 were used for training and remained 300 for testing.

B. Fuzzy System Structure

We used a Sugeno-type Fuzzy inference system with three inputs consisting dimensions of HSL color space. Ten membership functions for H dimension and three for each S and L dimensions was used. Ten constant membership functions were used for output variable. Three types of membership functions trapezoidal, Gaussian and Bell-shaped were investigated for the purpose of their comparison over this problem. Maximum and minimum of membership functions are determined according to [2]. Our results rank membership functions as Bell-shaped, Gaussian and trapezoidally respectively according to their performance.

C. Algorithm Parameters

We manually set values for parameters based on previous studies and problem characteristics [24]. Those values are listed in Table I.

D. Initial Population Generation

Proper initialization of particles plays an important role in algorithm efficiency. We compared two initialization methods, randomly generating rules and rule generation using neurofuzzy method. In order to generate fuzzy rules, we used ANFIS toolbox provided with MATLAB® programming environment.

<table>
<thead>
<tr>
<th>TABLE II</th>
<th>ANFIS PERFORMANCE</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Trapezoidal membership function</td>
</tr>
<tr>
<td>Number of rules</td>
<td>Accuracy</td>
</tr>
<tr>
<td>90</td>
<td>60%</td>
</tr>
</tbody>
</table>

As it is shown in Table III, a higher performance is achievable with ANFIS initialization compared with random method. The reason might be that we constrain the maximum number of iterations which is needed for algorithm to be computationally tractable. The total number of rules generated with this method is about one third of the average number of rules generated with ANFIS.

An example of running our method to segment a soccer field image is shown in Fig. 6. Top row shows the performance of our method with ANFIS initialization and Bell-shaped membership functions. Bottom row illustrates the performance of ANFIS method. Images at the left are originals and those at the right are segmented ones.
A new method for segmentation of color images using a CLPSO-based fuzzy system was introduced in this paper. A particle of the swarm codes for a set of fuzzy rules. A fitness function rates for the Optimality of each particle. During iterations, particles try to maximize fitness function by cooperatively working on search space. This process is continued until either maximum number of iteration is met or average velocity approaches zero. Finally, the rule base represented by the best particle is used for the task of image segmentation. Because of smaller number of rules generated by this method, it shows higher computation speed. Using CLPSO for searching in the space of all possible solutions we achieved a better solution in terms of small size and high performance compared with ANFIS. Because we used a large training set including images taken at different illuminations, this method is robust to variations in illumination.

VI. CONCLUSION

This paper introduced a new method for segmentation of color images using a CLPSO-based fuzzy system. The new method showed better performance compared to ANFIS due to its smaller number of rules and higher computation speed. Future work could include testing the method on a larger and more diverse dataset to further validate its effectiveness.

REFERENCES


Ali Borji received the B.S. degree in computer engineering (software) from Petroleum University of Technology, Tehran, Iran, in 2001, and the M.S. degree in computer engineering (artificial intelligence) from Shiraz University, Shiraz, Iran in 2004. He is now working toward his Ph.D. at the School of Cognitive Sciences, Institute for Studies in Theoretical Physics and Astronomy, Tehran, Iran. His research interests include, human and machine vision, machine learning, pattern recognition, and biologically motivated algorithms.

Mandana Hamidi received the B.S. degree in computer engineering (software) from Azad University of Shiraz, Shiraz, Iran, in 2000, and she is working toward her M.S. degree in computer engineering (artificial intelligence) in Azad University of Qazvin, Qazvin, Iran. Her research interests include, human and machine vision, machine learning, pattern recognition, and biologically motivated algorithms.