I J C T A, 9(2-A), 2016, pp. 763-777

A Novel Strategy of Differential Evolution Algorithm's Crossover Operator Based on Graylevel Clusters Similarity for Automatic Multilevel Image Thresholding

Khoirul Umam[1], Agus Zainal Arifin[2] and Dini Adni Navastara[3]

ABSTRACT

Automatic multilevel image thresholding approach has wide optimal solution search space due to its ability to determine thresholds number and positions, simultaneously. Searching its optimal solution using standard Differential Evolution (DE) algorithm can decrease its efficiency due to slow convergence. Therefore, a strategy that can restrict the search space is needed in order for optimizations being efficient. In this paper we propose a novel strategy of DE's crossover operator based on graylevel clusters similarity for automatic multilevel image thresholding. We restrict the search space by only recombining graylevel clusters which have small similarity. Graylevel clusters similarity is performed by computing the inter-class and intra-class variance of adjacent graylevel clusters. Experiments on grayscale image of Berkeley Segmentation Dataset (BSDS500) show that the proposed crossover strategy can generate segmented images with misclassification error of 7.96% better than those of existing crossover strategies which not compute the graylevel clusters similarity. It only requires the average of 636 generation to find the optimal solution less than compared crossover strategies.

Keywords*:* automatic multilevel image thresholding, clusters similarity, crossover, Differential Evolution.

1. INTRODUCTION

Multilevel image thresholding is an approach on graylevel thresholding technique. It can be used to segment image that has more than one distinct object. With this approach, the image histogram is divided into more than two clusters graylevel by using a set of thresholds [1, 2, 3, 4].

Generally, the number of threshold that used to segment an image can be determined manually by experts. However it can lead the subjectivity factor in segmentation results. There are also cases where it difficult to set the threshold number manually [3]. Therefore multilevel image thresholding evolve into automatic multilevel image thresholding. On the approach, the number and position of thresholds that are used to segment an image is determined automatically.

The number and position of thresholds on multilevel image thresholding problem can be found using optimization method. It can be done due to the problem can be considered as an optimization problem to a certain objective function, generally [5]. The method is also more efficient than the exhaustive search method [3].

The algorithm that widely used to solve optimization problems is metaheuristic algorithm [6]. Some studies have proposed the implementation of metaheuristic algorithm for multilevel image thresholding problem. Horng and Jiang [7] proposed the implementation of Firefly Algorithm (FA) to locate thresholds position by optimizing maximum entropy function. Cuevas, et al [2] proposed the implementation of

[1,2,3] Department of Informatics, Faculty of Information Technology, Institut Teknologi Sepuluh Nopember, Surabaya, Indonesia, *Emails: 1khoirul.umam35@gmail.com, 2agusza@cs.its.ac.id, 3dini_navastara@if.its.ac.id*

Differential Evolution (DE) algorithm to locate thresholds position by optimizing Gaussian function parameters. Sarkar and Das [8] also proposed the implementation of DE to locate thresholds position by optimizing maximum Tsallis entropy function on 2D histogram. Ali, et al [9] proposed the implementation of Synergetic Differential Evolution (SDE) algorithm, i.e. a modification of DE to optimize the thresholds position. Ayala, et al [10] propose a Beta Differential Evolution (BDE) algorithm i.e. also a modification of DE to optimize thresholds position. Djerou, et al [11], Ouadfel and Meshoul [3], Hammouche, et al [1], and Hosseini [6] proposed the implementations of Binary Particle Swarm Optimization (BPSO) algorithm, a hybrid of Particle Swarm Optimization (PSO) with Simulated Annealing (SA) algorithm, Genetic Algorithm (GA), and Intelligent Water Drops (IWDs) algorithm for automatic multilevel image thresholding problem, respectively.

Among the optimization algorithms that have been proposed to solve multilevel image thresholding problems, DE has better performance than the other algorithms. DE has ability to maintain convergences to optimal solution better than GA and PSO. DE also can generate better segmentation results compared with SA, Ant Colony Optimization (ACO), and Tabu Search (TS) [5]. Furthermore, DE has better efficiency level because it takes less time than PSO and Artificial Bee Colony (ABC) to reach the optimal solution [4].

DE's ability to reach optimal solution is influenced by the evolution process of its solution population. The process is performed through series of operations. One of the operations that affect DE's evolution process is crossover operation. The operation is used to expand optimal solution search space by using certain strategy. In the standard of DE, the strategy of crossover operation involves random values which generated by the system and crossover rate which determined by experts. Comparison of the two values is used as a benchmark for crossover operation [12].

DE's standard crossover strategy has proven can be used to solve the multilevel image thresholding problem which only optimizing the combination of thresholds position [2, 4, 5,8, 9, 10]. But if it is used in automatic multilevel image thresholding problem, then it will affect the changes of thresholds number and positions combination simultaneously. Thus, the possible combinations that will be searched are larger, or in other words the search space becomes wider. It can lead to an increase of times or iterations that required for reaching the optimal solution. Increasing the population size in order to covers wider search space is also not an efficient solution because it can increase the memory requirements [13]. Therefore, a strategy that can restrict the search space is needed to find optimal solutions more efficient.

In this paper we propose a novel strategy of DE's crossover operator based on graylevel clusters similarity for automatic multilevel image thresholding problem. The similarity is calculated based on the value of adjacent graylevel cluster's inter-class and intra-class variances [14] which formed in each solution. Thus, the searching of optimal solution is only performed on solutions that offer combination of thresholds number and position which capable to separate each graylevel cluster with maximum distance.

2. DIFFERENTIAL EVOLUTION (DE) ALGORITHM

DE is an example of evolutionary algorithm that introduced by Storn and Price [12]. It searches the optimal solution by applying parallel searching method using a set of solutions in a population. Generally, optimization process using DE begins by initializing the population. The population then evolves through mutation, crossover, and selection operation. The evolutionary process is repeated until a stopping criterion is reached.

DE's population initialization is performed by generating NP solutions. Each solution is represented as a vector with D components and called as target vector X_s. Each target vector's component $x_{s,d}$ for $s = [1, ..., NP]$ and $d = [1, ..., D]$ is randomly initialized by a real-value between specified lower-bound x_{min} and upper-bound x_{max} as formulated in (1):

$$x_{s,d}(0) = x_{min} + (x_{max} - x_{min}) \cdot rand_{s,d,} \tag{1}$$

where $rand_{s,d}$ denotes a uniform random value between 0 and 1 which generated for dth component of target vector [12, 15].

After the initialization phase, the optimizations using DE proceed to mutations. On each generation g, the operation will generate NP donor vector $V_s(g)$[15]. In DE's mutations standard scheme that known as *DE/rand/1*, the mutations conducted by adding the weight of a target vector $X_{s1}(g)$ with the weighted difference between two other target vectors ($X_{s2}(g)$ and $X_{s3}(g)$). The weighted difference is scaled by a mutation factor F [12] as shown in equation (2):

$$V_s(g) = X_{s1}(g) + F \cdot (X_{s2}(g) - X_{s3}(g)). \tag{2}$$

Target vector $X_{s1}(g)$, $X_{s2}(g)$, and $X_{s3}(g)$ are three different vectors that randomly chosen and not equal to the target vector $X_s(g)$.

Mutations may cause the value of donor vector's component $v_{s,d}(g)$ violates the specified boundaries (i.e. x_{min} and x_{max}). In order to normalize the component value, a rule that formulated in equation (3) can be used [16]:

$$v_{s,d}(g) = \begin{cases} 2x_{\min} - v_{s,d}(g), & v_{s,d}(g) < x_{\min} \\ 2x_{\max} - v_{s,d}(g), & v_{s,d}(g) < x_{\max} \end{cases}. \tag{3}$$

Every pair of target vector $X_s(g)$ and donor vector $V_s(g)$ then recombined into a trial vector $U_s(g)$ through crossover operation. Target vector's components $x_{s,d}(g)$ and donor vector's components $v_{s,d}(g)$ are mixed to produce trial vector's components $u_{s,d}(g)$ using certain strategy. The examples of crossover strategy that can be used are binomial crossover [12], exponential crossover [15], and self-adaptive crossover [16]. The strategies use a random generated value and crossover rate (CR) to decide the recombination process of a vector solution's component.

The last operation on each DE's generation is selection operation. It selects the solution vectors that will be maintained on the next generation. The selections conducted by comparing a target vector's fitness value $f(X_s(g))$ with a trial vector's fitness value $f(U_s(g))$. Solution vector with the best fitness value is chosen as target vector on the next generation ($X_s(g+1)$) [15]. Selection formulation is shown in equation (4):

$$X_s(g+1) = \begin{cases} U_s(g), & f(U_s(g)) \geq f(X_s(g)) \\ X_s(g), & f(U_s(g)) < f(X_s(g)) \end{cases}. \tag{4}$$

3. GRAYLEVEL CLUSTERS SIMILARITY

The similarity of adjacent graylevel clusters can be known by measuring their distance. The farther distance between two adjacent graylevel clusters indicates smaller similarity between both of graylevel clusters, and vice versa.

Let n_l denotes the occurrence frequency of pixel with graylevel l, N denotes the total number of pixels in the image, p_l denotes the probability of pixel's occurrence with graylevel l as formulated in (5):

$$p_l = \frac{n_l}{N}, \tag{5}$$

C_k denotes the kth graylevel cluster, t_k denotes the kth threshold, along with ω_k and μ_k denotes the occurrence probability of pixels belonging to C_k and the mean of C_k, respectively, as formulated in equation (6) and equation (7):

$$\omega_k = \sum_{l=t_{k-1}+1}^{t_k} p_l, \tag{6}$$

$$u_k = \frac{1}{\omega_k} \sum_{l=t_{k-1}+1}^{t_k} l p_l. \tag{7}$$

The distance of two adjacent graylevel clusters C_{k_1} and C_{k_2} ($Dist\left(C_{k_1}, C_{k_2}\right)$ can be calculated using inter-class and intra-class variances of the adjacent graylevel clusters as formulated in equation (8):

$$Dist\left(C_{k_1}, C_{k_2}\right) = \sigma_I^2\left(C_{k_1} \cup C_{k_2}\right) \cdot \sigma_A^2\left(C_{k_1} \cup C_{k_2}\right). \tag{8}$$

Inter-class variance of C_{k_1} and C_{k_2} $\left(\sigma_I^2\left(C_{k_1} \cup C_{k_2}\right)\right)$ is defined as sum of the square distances between the means of the two graylevel clusters and the total mean of both clusters. While intra-class variance of C_{k_1} and C_{k_2} $\left(\sigma_A^2\left(C_{k_1} \cup C_{k_2}\right)\right)$ is defined as variance of all pixel graylevel values in the merged cluster [14]. Formulations for both of values are shown in equation (9) and equation (10), respectively:

$$\sigma_I^2\left(C_{k_1} \cup C_{k_2}\right) = \frac{\omega_{k_1} \cdot \omega_{k_2}}{\left(\omega_{k_1} + \omega_{k_2}\right)^2}\left(\mu_{k_1} - \mu_{k_2}\right)^2. \tag{9}$$

$$\sigma_A^2\left(C_{k_1} \cup C_{k_2}\right) = \frac{1}{\omega_{k_1} + \omega_{k_2}} \sum_{l=t_{k_1-1}+1}^{t_{k_2}} \left(\left(l - M\left(C_{k_1} \cup C_{k_2}\right)\right)^2 \cdot p_l\right). \tag{10}$$

where $M\left(C_{k_1} \cup C_{k_2}\right)$ denotes the global mean of C_{k_1} and C_{k_2} which formulated in equation (11):

$$M\left(C_{k_1} \cup C_{k_2}\right) = \frac{\omega_{k_1} \cdot \mu_{k_1} + \omega_{k_2} \cdot \mu_{k_2}}{\omega_{k_1} + \omega_{k_2}}. \tag{11}$$

4. METHOD

In this section we describe our proposed crossover strategy and DE scheme that implements the strategy.

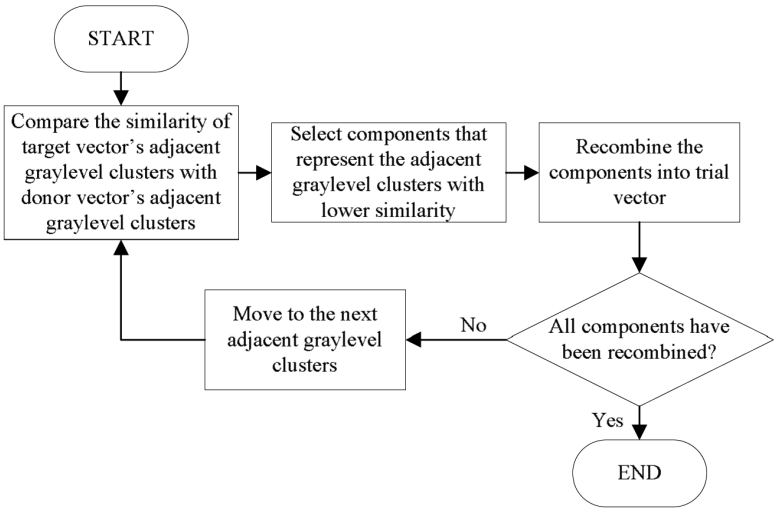

Figure 1: Flowchart of proposed crossover

4.1. Proposed Crossover Strategy

As mentioned before, in this paper we propose a novel strategy of DE's crossover operator based on graylevel clusters similarity that formed at thresholding solutions. The thresholding solutions are represented by target vectors and donor vectors. For the convenience, we denote our proposed crossover strategy as Graylevel Clusters Similarity (GCS) strategy. Recombination procedure on GCS is conducted by comparing the adjacent graylevel clusters similarity which represented by target vector's components with the adjacent graylevel clusters similarity which represented by donor vector's components. Solution vector's components which represent graylevel clusters with lower similarity (i.e. farther distance) are chosen to recombine into trial vector's components. The procedure is repeated until all components have been recombined into trial vector as illustrated in Figure 1. Thus, the formed trial vector is expected can represent a thresholding solution that has low graylevel clusters similarity. Therefore, the searching of optimal thresholding solution can be restricted only on thresholding solutions which able to minimize the graylevel clusters similarity.

Assume that K_X graylevel clusters and K_V graylevel clusters are formed at thresholding solutions which represented by target vector $X_s(g)$ and donor vector $V_s(g)$, respectively. It indicates that there are $K_X - 1$ thresholds in the threshold set TX of thresholding solution's which represented by $X_s(g)$ $\left(TX = \left[t_1^X, ..., t_{K_{X^{-1}}}^X\right]\right)$. It also indicates that there are $K_v - 1$ thresholds in the threshold set TV of thresholding solution's which represented by $V_s(g)$ $\left(TV = \left[t_1^V, ..., t_{K_{V^{-1}}}^V\right]\right)$. If the largest graylevel intensity that appear on the image L_{max} is considered as Kth threshold (i.e. $t_k = L_{max}$), then $TX = \left[t_1^X, ..., t_{K_{X^{-1}}}^X, L_{\max}\right]$ and $TV = \left[t_1^V, ..., t_{K_{V^{-1}}}^V, L_{\max}\right]$. Let L_{min} denotes the smallest graylevel intensity that appear on the image, c denotes the cluster's first graylevel, C_m^X denotes graylevel cluster which represented by $X_s(g)$, C_n^V denotes nth graylevel cluster which represented by $V_s(g)$, along with DX and DV denotes the distance of two adjacent graylevel clusters at $X_s(g)$ and $V_s(g)$, respectively. The algorithm of $X_s(g)$ and $V_s(g)$ recombination process to produce trial vector $U_s(g)$ in GCS is described in Algorithm 1.

Algorithm 1. Proposed Crossover Strategy Algorithm

1. Initialize $c \rightarrow c = L_{min}$.
2. Choose threshold in $TX\left(t_m^X\right)$ and nth threshold in $TV\left(t_n^V\right)$ where t_m^X as well as t_n^V are located closest to c and have greater graylevel intensity value than c respectively.
3. Form two adjacent graylevel clusters on thresholding solution which represented by $X_s(g)$ and $V_s(g)$ respectively:
   a. $C_m^X = \left[C, \ ..., \ t_m^X\right]$ *and* $C_{m+1}^X = \left[t_m^X + 1, \ ..., \ t_{m+1}^X\right]$.
   b. $C_n^V = \left[c, \ ..., \ t_n^V\right]$ *and* $C_{n+1}^V = \left[t_n^V + 1, \ ..., \ t_{n+1}^V\right]$.
4. Measure the adjacent graylevel clusters similarity or distance using equation (8):
   a. $DX = Dist\left(C_m^X, C_{m+1}^X\right)$.
   b. $DV = Dist\left(C_n^V, C_{n+1}^V\right)$.

5. Compare DX with DV and do one of following conditions:

   a. If $DX > DV$, then:

   i. $u_{s,d}(g) = x_{s,d}(g)$, for $d = [(c - L_{min} + 1), ..., (t_m^X - L_{min} + 1)]$.

   ii. Update $c \rightarrow c = t_{m+1}^X + 1$.

   b. If $DX \leq DV$, then:

   i. $u_{s,d}(g) = v_{s,d}(g)$, for $d = [(c - L_{min} + 1), ..., (t_n^V - L_{min} + 1)]$.

   ii. Update $c \rightarrow c = t_{n+1}^V + 1$.

6. Repeat steps 2-5 until $c > L_{max}$.

On the last iteration of each recombination process using GCS there is possibility that the last graylevel cluster C_K of $X_s(g)$'s or $V_s(g)$'s thresholding solution are remain and not recombined yet into $U_s(g)$. In order for the calculation and comparison of graylevel clusters similarity can still be made on C_K, we assume that there is a dummy cluster C_{K+1}. It acts as neighbor of C_K on each thresholding solution as illustrated on Figure 2. It is assumed has threshold $t_{K+1} = L_{max} + 1$ and probability $\omega_{K+1} = 0$. In order for two adjacent graylevel clusters similarity or distance value still has meaning even though one of graylevel cluster has 0 probability value, the inter-class variance formulation in equation (9) is modified into equation (12):

$$\sigma_I^2\left(C_{k_1} \cup C_{k_2}\right) = 1 + \frac{\omega_{k_1}.\omega_{k_2}}{\left(\omega_{k_1} + \omega_{k_2}\right)^2}\left(\mu_{k_1} - \mu_{k_2}\right)^2. \tag{12}$$

4.2. Optimization Scheme

In this paper, the optimization scheme using DE to solve the automatic multilevel image thresholding problem is illustrated by flowchart on Figure 3. For the convenience, we denote the scheme which implements GCS strategy using DE's scheme naming rules described in [12,15] as *DE/rand/1/-GCS*. The optimization uses DE so that DE's general phases (i.e. initialization, mutation, crossover, and selection) are used. Moreover, we also add three stages of binarization phase and an evaluation phase

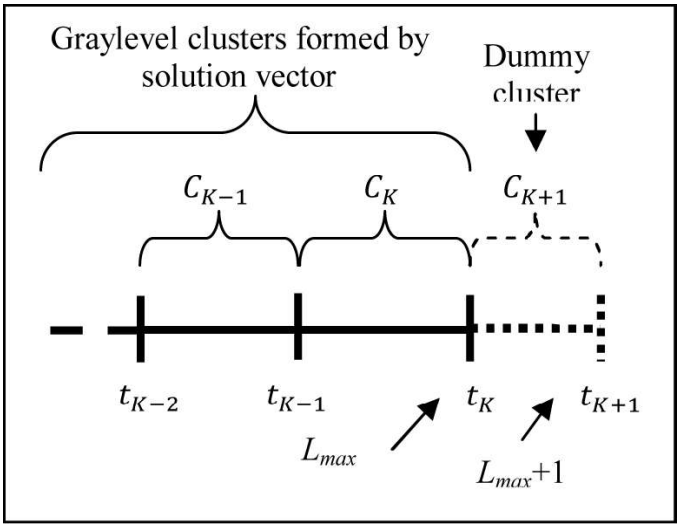

Figure 2: Illustration of dummy cluster's position.

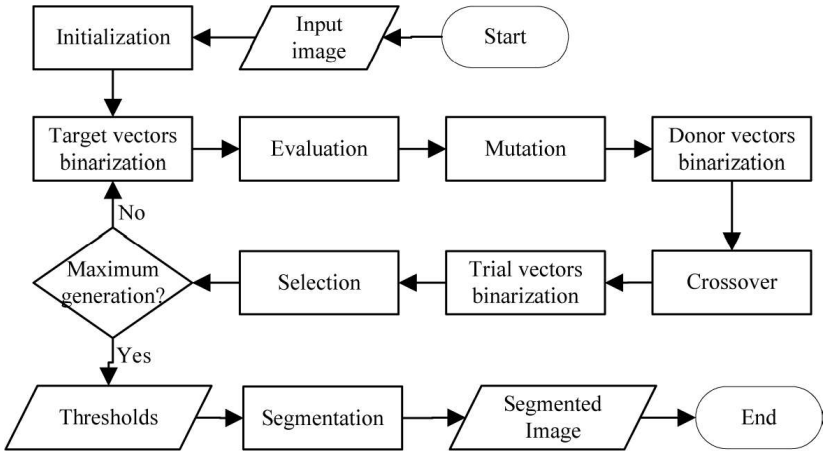

Figure 3: Optimization scheme

where fitness value of each founded thresholding solution is computed. The brief description for each phase is described in next subsections.

4.2.1. Initialization

In this phase, target vectors $X_s(g)$ are generated, where each $X_s(g)$ is consists of D components. Each component $x_{s,d}(g)$ is initialized using equation (1). The values of NP, x_{min}, and x_{max} are manually defined, the value of $rand_{s,d}$ is automatically and randomly generated by system, whereas the value of D is depending on image's L_{min} and L_{max} values as formulated in equation (13):

$$D = L_{max} - L_{min.} \tag{13}$$

4.2.2. Binarization

Binarization is a phase to encode real-value of each solution vector's component (i.e. target vector, donor vector, or trial vector) into binary-value. The binary-values are used to indicate the positions and number of thresholds which offered by thresholding solution that represented by the solution vector. If the binarization result of dth component in vector solution $x'_{s,d}(g) = 1,$ then the graylevel l which associated to the component is assumed as kth threshold (t_k). But if $x'_{s,d}(g) = 0,$ then l is not assumed as t_k. The connectivity between indices of solution vector's component d with l is formulated in equation (14):

$$l = L_{min} + d - 1. \tag{14}$$

An example of thresholds position that indicated by binary form of a vector solution $X'_s(g)$ is illustrated on Figure 4. Figure 4 illustrates that there are $K-1$ components of $X'_s(g)$ which have code 1, i.e. components with index 3, 6, … and $D-2$. Therefore, on the thresholding solution that represented by $X'_{s,d}(g)$ there are $K-1$ thresholds too, where $t_1 = L_{min} + 2$, $t_2 = L_{min} + 5$, ..., and $t_{K-1} = L_{max} - 3$. It will also lead to the formation of K graylevel clusters ($[C_1, ..., C_K]$). In this paper, L_{max} has no connectivity with vector solution's components and always considered to be selected as Kth threshold (t_K).

The rule to encode real form of $x_{s,d}(g)$ into its binary form $x'_{s,d}(g)$ is formulated in equation (15):

$$x'_{s,d}(g) = \begin{cases} 1, & rand_{s,d} < sigm\left(x_{s,d}(g)\right) \text{ and } p_l > 0 \\ 0 & rand_{s,d} \geq sigm\left(x_{s,d}(g)\right) \text{ or } p_l > 0 \end{cases}. \tag{15}$$

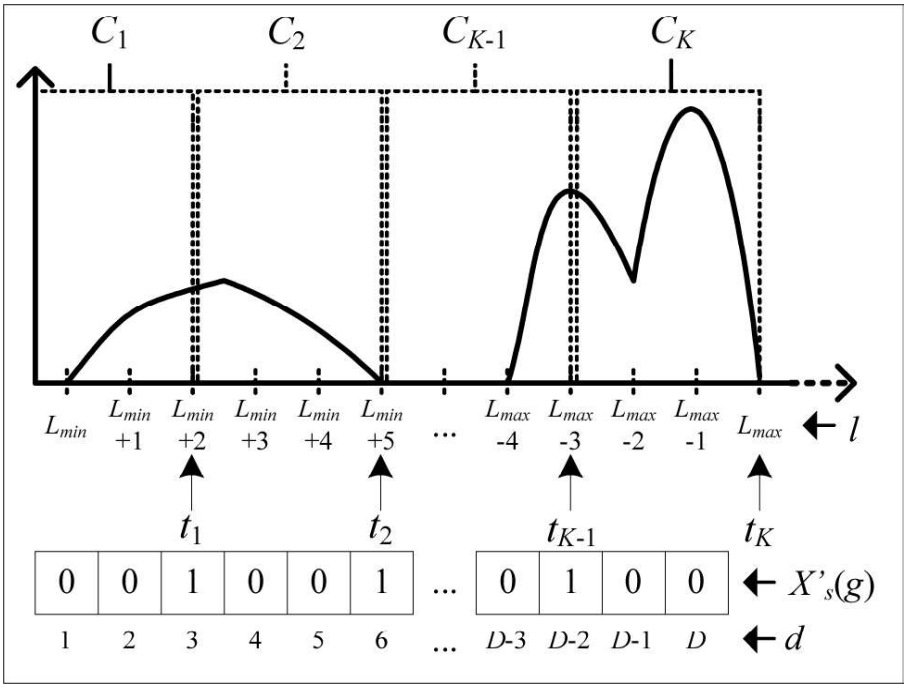

Figure 4: Example of thresholds encoding position and graylevel clusters forming on image histogram based on binarization result of vector solution's components

where *sigm*(*Y*) denotes the sigmoid value of *Y* as formulated in equation (16):

$$sigm(Y) = \frac{1}{1+e^{-Y}}. \tag{16}$$

The binarization rule in equation (15) lead to graylevels that can be selected as threshold are only graylevels that occur in input image (i.e. graylevels with probability value $p_l > 0$). Thus, there is no possibility that graylevel cluster C_k which has probability $\omega_k = 0$ is formed in the range of graylevel L_{min} to L_{max}.

4.2.3. Evaluation

In this phase, the computation of fitness value for each $X_s(g)$ is performed. The objective function that used to compute $X_s(g)$'s fitness value is Automatic Thresholding Criterion (ATC) function from [17]. It considers the number of formed graylevel clusters (*K*) and the graylevel clusters dissimilarity which denoted by its within-class variance $\left(\sigma_W^2(K)\right)$. It is formulated in equation (17):

$$ATC(K) = \rho\left(\sigma_W^2(K)\right)^{1/2} + \left(\log_2 K\right)^2, \tag{17}$$

where ρ is a positive weighting constant which manually defined. The value of $\sigma_W^2(K)$ can be computed by integrating the total-class variance σ_T^2 and between-class variance of *K* formed graylevel clusters $\sigma_B^2(K)$ [18]. The formulations of each variance are shown in equation (18)–(20):

$$\sigma_W^2(K) = \sigma_T^2 - \sigma_B^2(K), \tag{18}$$

$$\sigma_T^2 = \sum_{l=0}^{L-1} (l-\mu)^2 p_l, \tag{19}$$

$$\sigma_B^2(K) = \sum_{k=1}^{K} \omega_k (\mu_k - \mu)^2, \tag{20}$$

where μ denotes the total means of all image pixels. If there are L graylevel in the image, then μ of image can be computed using equation (21):

$$\mu = \sum_{l=0}^{L-1} lp_l. \tag{21}$$

The ATC value of a thresholding solution can indicate the solution's quality. The smaller ATC value of the solution indicates that the solution has better quality, and vice versa. Therefore, the fitness value of thresholding solution that represented by $X_s(g)$ ($f(X_s(g))$) is inversely proportional with the solution's ATC value as formulated in equation (22):

$$f\left(X_s\left(g\right)\right) = \frac{1}{ATC\left(X_s^{'}\left(g\right)\right)}. \tag{22}$$

4.2.4. Mutation

In this phase, $X_s(g)$ is mutated using mutation scheme *DE/rand/1* which formulated in equation (2) until $V_s(g)$ is produced for s = [1, ..., NP]. The value of F which used on the scheme is manually defined. The constraint handling rule for $v_{s,d}(g)$ which formulated in equation (3) is also used in this phase.

4.2.5. Crossover

In this phase, NP pairs of $X_s(g)$ and $V_s(g)$ are recombined using GCS strategy that described on Subsection 4.1. This phase will produce NP trial vectors $U_s(g)$.

4.2.6. Selection

In this phase, the selection of solution vectors that will be maintained on the next generation is performed. Selections rule in this paper is following the general DE's selections rule which formulated in equation (4).

4.2.7. Segmentation

After the iteration or generation of optimal solution searching reaches the maximum generation, the searching is stopped and the optimal solution is obtained. The optimal solution is a thresholding solution that represented by binary form of global best vector $X_{gbest}(g)$ on g_{max} ($X'\square_{gbest}(g_{max})$). $X_{gbest}(g)$ is defined as a solution vector that has the best fitness value compared with the others vectors from the first generation until the last generation [16].

Thresholding solution that represented by $X'\square_{gbest}(g_{max})$ is used for segmenting input image into segmented image. Segmentations are conducted by changing the graylevel intensity l on each image pixels ($pix(l)$) into the rounding result of k*th graylevel cluster mean μ_k* where l is located. The formulation of segmentations is shown in equation (23):

$$pix(l) = round(\mu_k^*) \tag{23}$$

for $l \in C_k$*, l = [L_{min}, ..., L_{max}], and k* = [1, ..., K*], where K* denotes the number of graylevel clusters represented by optimal solution.

5. EXPERIMENTS AND RESULTS

In this paper we use images from Berkeley Segmentation Datasets (BSDS500)[1] as testing images. The dataset provides real-scene color images along with its human segmentation as the ground truths. We choose 150 images randomly for testing. But before we use it in testing, we convert each image into grayscale image using *rgb2gray* function on Matlab. Figure 5 shows the sample images from BSDS500

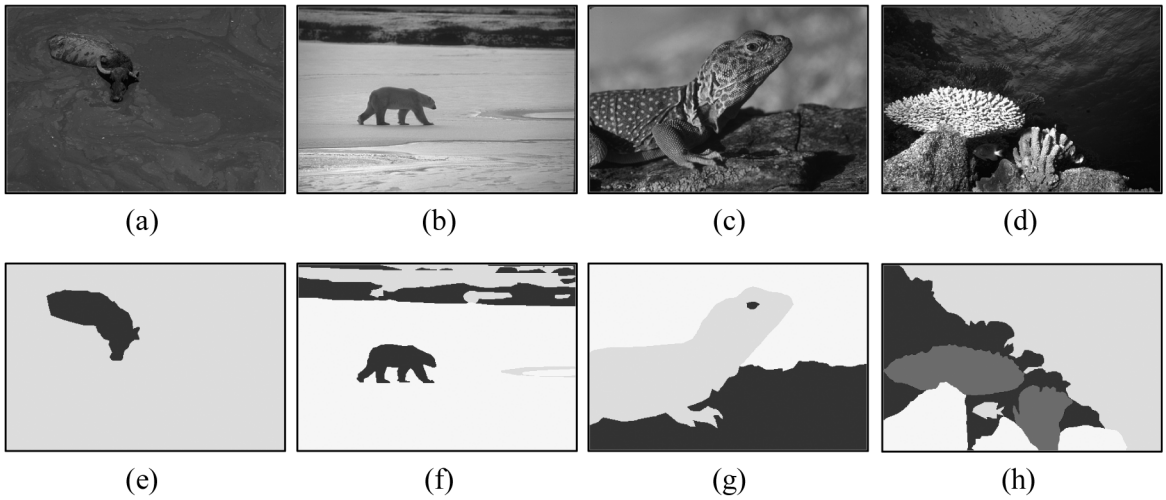

Figure 5: Sample images from BSDS500. (a)-(d) Image 1 – Image 4. (e)-(h) Ground truths.

which have been converted into grayscale images that used in testing along with its ground truths. The ground truths are colored using *label2rgb* function in Matlab with "*jet*" color map in order to point out different threshold levels in a better way. Testing is performed using Matlab R2013a software. It runs on Ms. Windows 7 operating system in computer that uses Intel i3 2.4 GHz processor and 3 GB of RAM.

DE algorithm that implements GCS strategy (*DE/rand/1/GCS* scheme) use specified DE parameters i.e. NP, g_{max}, F, x_{min}, and x_{max} to find the optimal thresholding solution (number and positions of thresholds) for each testing images. These parameters values are set to 50, 2500, 0.5, -25, and 25, respectively. While parameter that used on ATC function is set to 0.4. These values are the optimal value for each parameter that obtained from series of testing.

As the compared strategies, we also implement the existing crossover strategy i.e. binomial, exponential, and self-adaptive crossover strategies which described in [12,15,16] into crossover phase in optimization scheme (Figure 3). We called the compared schemes as *DE/rand/1/bin*, *DE/rand/1/exp*, and *DE/rand/1/self-adaptive* scheme, respectively. Each scheme then used to find the optimal thresholding solution for each testing images using same aforementioned parameters setting. Specifically for *DE/rand/1/bin* and *DE/rand/1/exp* schemes which also use *CR* parameter, this parameter value is set to 0.8 as suggested by [2].

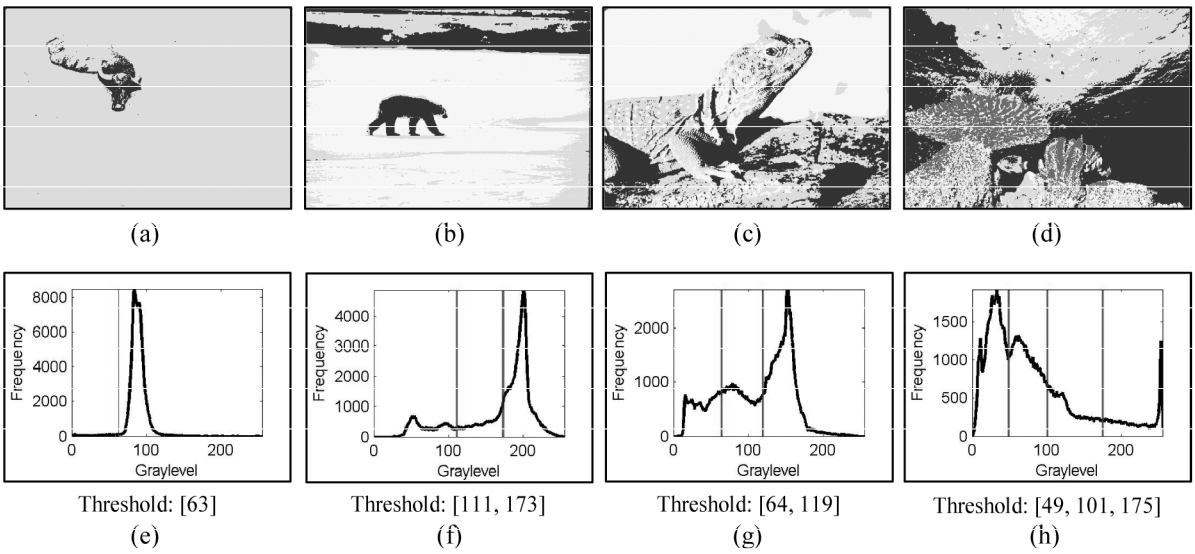

Figure 6: Segmentation results using *DE/rand/1/GCS* scheme. (a)-(d) Segmented images of sample images. (e)-(h) Thresholds posistion for each segmented result (indicated by red bars).

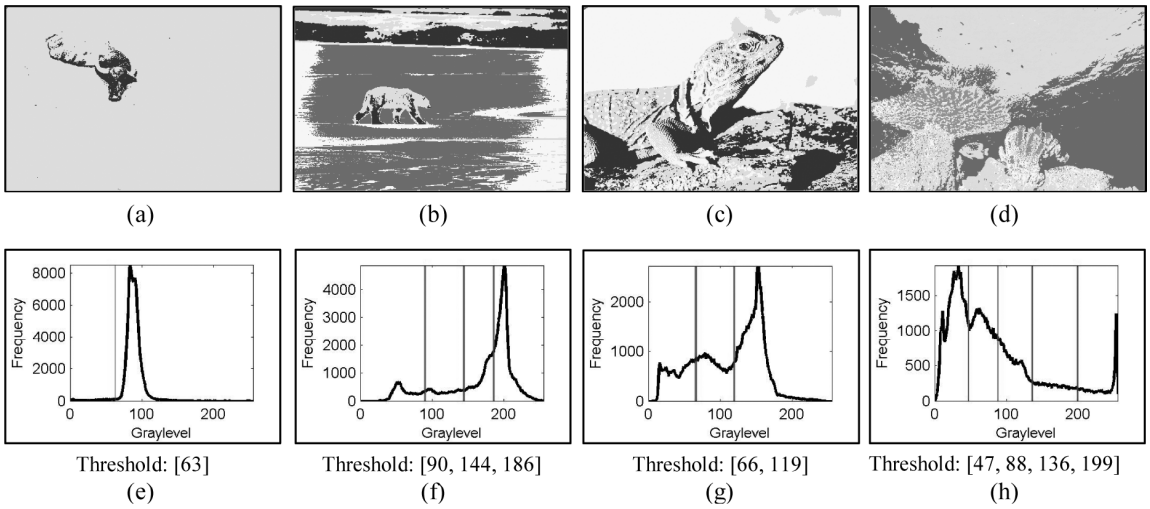

Figure 7: Segmentation results using *DE/rand/1/bin* scheme. (a)-(d) Segmented results of sample images. (e)-(h) Thresholds position for each segmented result (indicated by red bars).

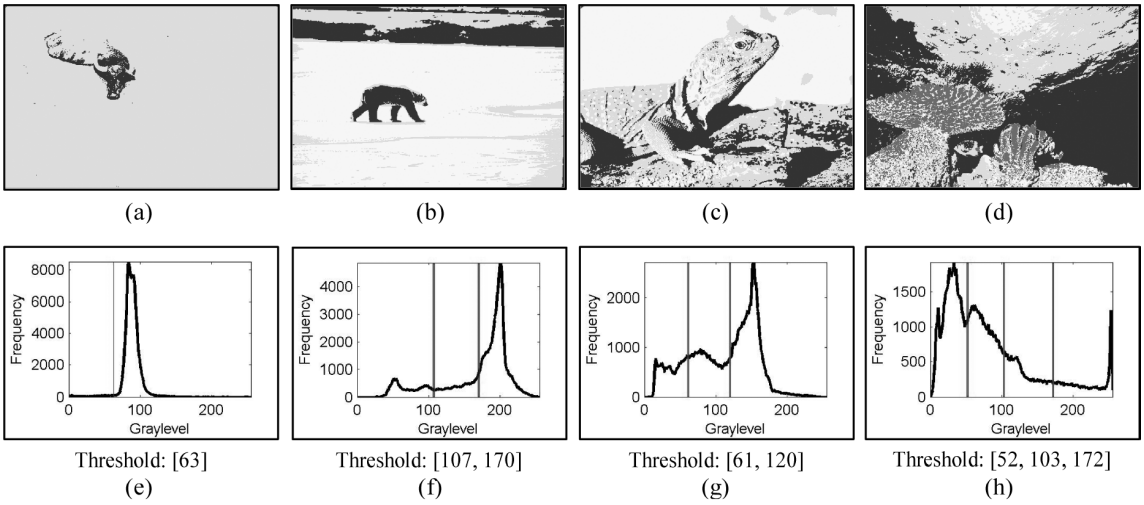

Figure 8: Segmentation results using *DE/rand/1/exp* scheme (a)-(d) Segmented results of sample images. (e)-(h) Thresholds position for each segmented result (indicated by red bars).

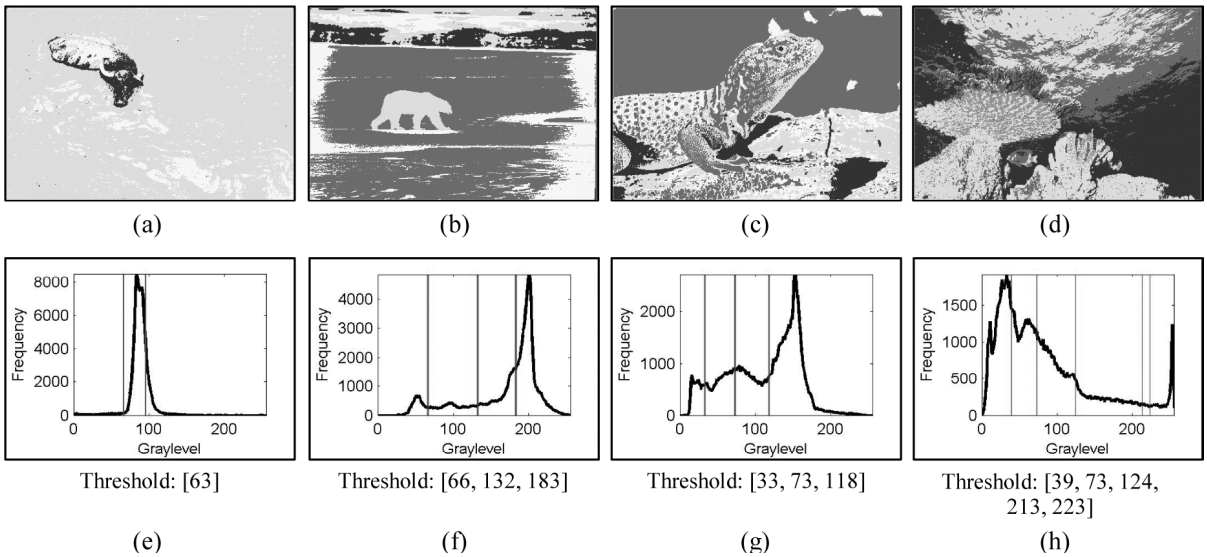

Figure 9: Segmentation results using *DE/rand/1/self-adaptive* scheme. (a)-(d) Segmented results of sample images. (e)-(h) Thresholds position for each segmented result (indicated by red bars).

In order to make the combination of random number occurrences is not affecting the optimizations result, the random generator for each data is reset on the beginning of optimizations. The resetting is performed by using *rng* function, i.e. a built-in function on Matlab R2013a. Thus the optimizations results and performances are only affected by the crossover strategy that implemented on the schemes.

The segmentation results of sample images (Image 1 – Image 4) that produced by *DE/rand/1/-GCS* scheme are shown in Figure 6. While the compared segmentation results of sample images that produced by compared schemes are shown in Figure 7 – Figure 9. We coloring the grayscale segmented images using *label2rgb* function in Matlab with "*jet*" color map to point out different threshold levels in a better way.

The performance of each scheme is evaluated by observing its convergences ability and quality of its generated segmented images. The scheme's convergences ability can be observed from its generation number and CPU time that required by the scheme to find the optimal solution. The quality of segmentation result can be observed using uniformity measure and misclassification error (ME) percentage.

Formulation to measure the uniformity of a segmented image $I_T(U)$ is shown in equation (24):

$$U = 1 - 2(K-1)\frac{\sum_{j=1}^{K}\sum_{i\in C_j}\left(L_i - \mu_j\right)^2}{N \times \left(L_{\max} - L_{\min}\right)^2}, \tag{24}$$

where K denotes the number of graylevel clusters that formed on I_T's histogram, C_j denotes the jth graylevel cluster on I_T, μ_j denotes the mean of graylevel intensity on C_j, L_i denotes the graylevel intensity of pixel i on original image, and N denotes the total number of I_T's pixels. The value of U is between 0 and 1, where a higher value of U means that the quality of I_T is better [1, 3, 9]. While the percentage of I_T's ME which has $M \bullet N$ dimension can be calculated based on the number of I_T's pixels which have different label of cluster compared with its related pixels on original image as formulated in equation (25) and equation (26):

$$ME = \left(1 - \frac{\sum_{m=1}^{M}\sum_{n=1}^{N}\left|I_0(m,n) \cap I_T(m,n)\right|}{M \cdot N}\right) \times 100. \tag{25}$$

$$\left|I_0(m,n) \cap I_T(m,n)\right| = \begin{cases} 1, & I_0(m,n) = I_T(m,n) \\ 0, & I_0(m,n) \neq I_T(m,n) \end{cases}. \tag{26}$$

where $I_0(m, n)$ and $I_T(m, n)$ denote the label of cluster in ground truth and segmented images, respectively. A lower percentage of ME means the quality of I_T is better. Equation (25) and equation (26) are a modification of ME's formula for bilevel thresholding problem which defined in [19].

The comparison of each tested scheme's ability to reach the optimal solution is depicted by the graph on Figure 10. The graph shows the average of the best fitness value that found on each scheme's generation. While the comparisons of the averages of generation number, CPU time that required to find the optimal solution, and CPU time that required to complete the searching process until the last generation by each tested scheme is shown in Table 1 along with its averages of uniformity values and ME percentages. The best values for each performance are indicated by bolded text.

6. DISCUSSION

Series of experiment has been performed to evaluate the performances of GCS which implemented on DE (*DE/rand/1/GCS* scheme) to solve the automatic multilevel image thresholding problem. Existing DE crossover strategies are used as compared crossover strategies. Based on graph in Figure 10, we know that the average of best fitness value that found on each generation of *DE/rand/1/GCS* scheme has the most

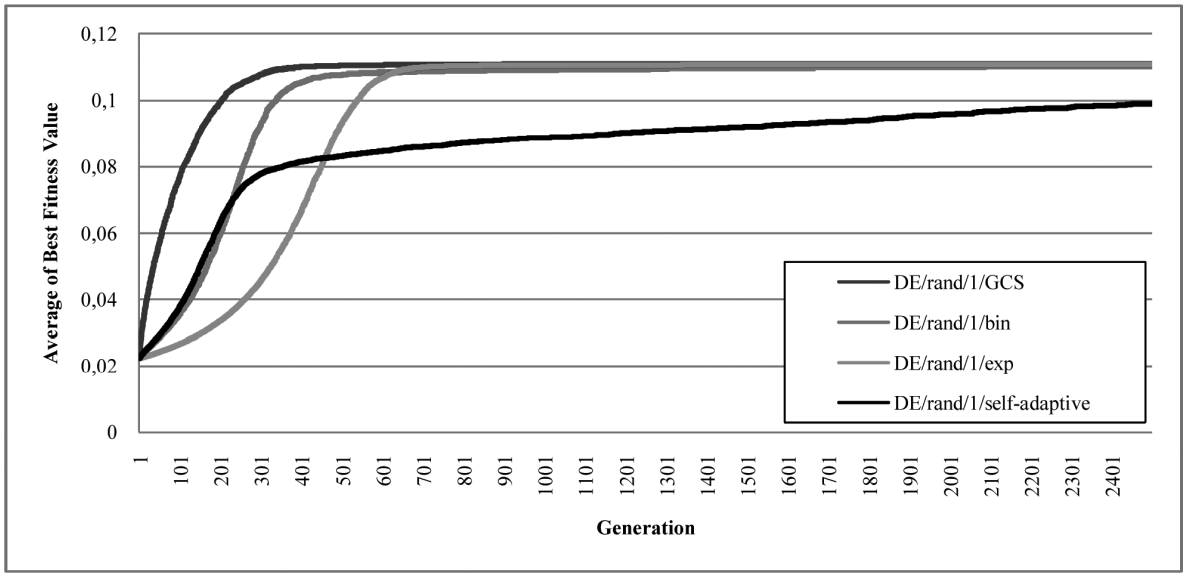

Figure 10: Comparison of the best fitness value averages that found on each generation for each tested scheme

Table 1
Comparison of Each Tested Scheme's Performances Averages Value

Performances	*Performances Averages Value for Each Tested Schemes*			
	DE/rand/ 1/GCS	*DE/rand/ 1/bin*	*DE/rand/ 1/exp*	*DE/rand/ 1/self adaptive*
Generations number that required to find the optimal solution	1106	1707	1647	1873
Best fitness value on the last generation	0.1109	0.1101	0.1108	0.0991
CPU time until optimal solution found (seconds)	94.35	36.75	36.47	40.54
CPU time until the last generation (seconds)	178.84	53.87	55.32	54.30
Uniformity	0.9943	0.9942	0.9943	0.9928
ME (%)	40.47	44.20	41.50	59.60

significant increment than the average of best fitness value that found on each generation of compared schemes since the first generation. It indicates that crossover operation based on graylevel clusters similarity has capability to direct the searching process to solutions that able to separate graylevel clusters with maximum distance (i.e. solutions with high fitness value). Thus the GCS strategy has capability to find the optimal solution better than compared crossover strategy.

The number of generation that required by *DE/rand/1/GCS* scheme to find the optimal solution is also better than the compared schemes. Based on Table 1, we know that *DE/rand/1/GCS* scheme requires the least generation numbers to find the optimal solution. It only requires average of 636 generations less than compared schemes to find the optimal solution. It indicates that crossover operation based on graylevel clusters similarity able to decrease the generation numbers that required on finding the optimal solution.

DE/rand/1/GCS scheme is also able to reach the average of best fitness value on the last generation higher than compared schemes even though it is not significant. Based on Table 1 and Figure 10, *DE/rand/ 1/GCS* scheme only has significant differences of best fitness value's average with *DE/rand/-1/self-adaptive* scheme, which is 0.0118 or 11.91% higher. While compared with *DE/rand/1/exp* and *DE/rand/1/bin* schemes, *DE/rand/1/GCS* scheme only have the averages of best fitness value 0.0001 and 0.0008 (0.09% and 0.73%)

higher than the schemes, respectively. It indicates that GCS strategy has capability to give thresholding solutions that have no significant differences with binomial and exponential crossover strategies, but significantly better than self-adaptive crossover strategy.

The average of uniformity values which obtained by *DE/rand/1/GCS* scheme is also indicates that the scheme able to produce the segmentation results that have less significant differences compared with *DE/rand/1/bin* and *DE/rand/1/exp* schemes. More significant differences of segmentation results are only shown by *DE/rand/1/self-adaptive* scheme. Based on Table 1, *DE/rand/1/GCS* scheme has same average value of uniformity with *DE/rand/1/exp* scheme and only 0.0001 higher than *DE/-rand/1/bin* scheme's average value of uniformity. While comparison with the average value of *DE/-rand/1/self-adaptive* scheme's uniformity shows that the *DE/rand/1/GCS* scheme has the average value of uniformity 0.0015 higher than the scheme.

The average percentages of ME which obtained by *DE/rand/1/GCS* scheme is also indicates that the scheme able to produce segmentation results that have less significant differences than *DE/rand/1/-bin* and *DE/rand/1/exp* schemes. The average percentages of ME which obtained by *DE/rand/1/GCS* is only 1.03% higher than the average percentage of ME which obtained by *DE/rand/1/exp* scheme and only 3.73% higher than the average percentage of ME which obtained by *DE/rand/1/bin* scheme. Significant difference of ME's average percentage is only shown by the comparison with *DE/rand/1/selfadaptive* scheme. *DE/rand/1/GCS* scheme has the average percentage of ME that 19.13% better than *DE/rand/1/self-adaptive* scheme. It means *DE/rand/1/GCS* scheme is able to generate segmentation results with qualities 7.96% better than the compared schemes.

However, when viewed in terms of computational times, *DE/rand/1/GCS* scheme significantly requires more time than compared schemes. Based on Table 1, we know that *DE/rand/1/GCS* scheme requires the average of CPU time 53.81 – 57.88 seconds (rounded to be 54 – 58 seconds) or 2.33 –2.59 times longer than compared schemes to find the optimal solution. While to complete the searching process until the last generation, *DE/rand/1/GCS* requires the average of CPU time 123.52 – 124.97 seconds (rounded to be 124 – 125 seconds) or 3.23 – 3.32 times longer than compared schemes. It indicates that the computation of graylevel clusters similarity which is the base of GCS strategy can increase the requirement of computational times. Thus, if viewed in terms of computational times, then the proposed crossover strategy is not more efficient than the existing DE crossover strategies.

7. CONCLUSION

This paper has proposed and described a novel strategy of DE's crossover operator based on graylevel clusters similarity for solving the automatic multilevel image thresholding problem. The proposed crossover strategy tries to restrict the optimal solution search space by only recombining vector solution's components that represent combination of graylevel clusters which have smaller similarity. Thus, the searching of optimal solution is only performed on solutions that offer combination of thresholds number and positions which capable to separate each graylevel cluster with maximum distance.

The experimental results show that the proposed crossover strategy is able to provide segmentation results that have same quality with segmentation results that provided by existing crossover strategies, even better. In our experiments its segmentation results qualities which indicated by its average percentages of ME are 7.96% better than the compared crossover strategies. The proposed crossover strategy also requires less searching generations than the existing strategies to provide the segmentation results. It only requires 636 generations less than the compared crossover strategies.

NOTES

1. http://www.eecs.berkeley.edu/Research/Projects/CS/vision/grouping/resources.html

REFERENCES

[1] K. Hammouche, M. Diaf, P. Siarry, "A Multilevel Automatic Thresholding Method Based on A Genetic Algorithm for A Fast Image Segmentation", *Computer Vision and Image Understanding*, 109:163-175, 2008.

[2] E. Cuevas, D. Zaldivar, M.P. Cisneros, "A Novel Multi-Threshold Segmentation Approach Based on Differential Evolution Optimization", *Expert Systems with Applications*, 37:5265-5271, 2010.

[3] S. Ouadfel, S. Meshoul, "A Fully Adaptive and Hybrid Method for Image Segmentation Using Multilevel Thresholding", *International Journal of Image, Graphics, and Signal Processing*, 1:46-57, 2013.

[4] V.O. Enciso, E. Cuevas, H. Sossa, "A Comparison of Nature Inspired Algorithms for Multi- Threshold Image Segmentation", *Expert Systems with Applications*, 40:1213-1219, 2013.

[5] K. Hammouche, M. Diaf, P. Siarry, "A Comparative Study of Various Meta-Heuristic Techniques Applied to The Multilevel Thresholding Problem", *Engineering Applications of Artificial Intelligence*, 23:676-688, 2010.

[6] H.S. Hosseini. "Intelligent Water Drops Algorithm for Automatic Multilevel Thresholding of Grey-Level Images Using a Modified Otsu's Criterion", *International Journal Modelling, Identification, and Control*, 15(4):241-249, 2012.

[7] M.H. Horng, T.W. Jiang, "Multilevel Image Thresholding Selection Based on The Firefly Algorithm", *Proceeding of Symposia and Workshops on Ubiquitous, Automatic, and Trusted Computing*, Xian, Shaanxi, 58-63, 2010.

[8] S. Sarkar, S. Das, "Multilevel Image Thresholding Based on 2D Histogram and Maximum Tsallis Entropy - A Differential Evolution Approach", *IEEE Transactions on Image Processing*, 22(12):4788-4797, 2013.

[9] M. Ali, C.W. Ahn, M. Pant, "Multi-Level Image Thresholding by Synergetic Differential Evolution", Applied Soft Computing, 17:1-11, 2014.

[10] H.V.H. Ayala, F.M.d. Santos, V.C. Mariani, L.d.S. Coelho, "Image Thresholding Segmentation Based on a Novel Beta Differential Evolution Approach", *Expert Systems with Applications*, 42:2136-2142, 2015.

[11] L. Djerou, H.E. Dehimi, N. Khelil, M. Batouche, "Automatic Multilevel Thresholding using Binary Particle Swarm Optimization for Image Segmentation", *Proceeding of International Conference of Soft Computing and Pattern Recognition*, Malacca, 66-71, 2009.

[12] R. Storn, K. Price, "Differential Evolution- A Simple and Efficient Heuristic for Global Optimization over Continous Spaces", *Journal of Global Optimization*, 11:341-359, 1997.

[13] R. Mallipeddi, P.N. Suganthan, "Empirical Study on the Effect of Population Size on Differential Evolution Algorithm", *Proceeding of IEEE Congress on Evolutionary Computation*, Hong Kong, 3663 - 3670, 2008.

[14] A.Z. Arifin, A. Asano, "Image Segmentation by Histogram Thresholding Using Hierarchical Cluster Analysis", *Pattern Recognition Letters*, 27:1515-1521, 2006.

[15] S. Das, P.N. Suganthan, "Differential Evolution: A Survey of The State-of-The-Art", *IEEE Transactions on Evolutionary Computation*, 15(1):4-31, 2011.

[16] R.M. Alguliev, R.M. Aliguliyev, N.R. Isazade, "Multiple Documents Summarization Based on Evolutionary Optimization Algorithm", *Expert Systems with Applications*, 40:1675-1689, 2013.

[17] J.C. Yen, F.J. Chang, S. Chang, "A New Criterion for Automatic Multilevel Thresholding", *IEEE Transactions on Image Processing,* 4(3):370-378, 1995.

[18] N. Otsu, "A Threshold Selection Method from Gray-Level Histograms", *IEEE Transactions on Systems, Man, and Cybernetics*, 9(1):62-66, 1979.

[19] M. Sezgin, B. Sankur, "Survey Over Image Thresholding Techniques and Quantitative Performance Evaluation", *Journal of Electronic Imaging*, 13(1):146-165, 2004.