International Journal of Computer Applications (0975 – 8887)
Volume 163 – No 11, April 2017

Optimal Multilevel Threshold Selection for Gray Level Image Segmentation using SMS Algorithm

Kotte Sowjanya
Research Scholar
Department of ECE
College of Engineering (A),
Andhra University
Visakhapatnam, A.P.,
India-530003

P. Rajesh Kumar, PhD
Professor and H.O.D
Department of ECE
College of Engineering (A),
Andhra University
Visakhapatnam, A.P.,
India-530003

ABSTRACT
Image processing is one of the real research regions in the most recent four decades. Numerous researchers have contributed very great algorithms and reported outstanding results. In this paper, state of matter search optimization based multilevel thresholding is implemented for the segmentation of gray scale Images. Set of standard gray level images are considered for image segmentation. The optimal multilevel threshold is found by maximizing the very popular objectives such as between class variance (Otsu method) and Kapur's entropy. The outcomes are looked at with the aftereffects of the existing algorithms like IDSA, HSA, PSO, and BF. The outcomes uncover that the execution of state of matter search optimization algorithm based optimal multilevel threshold for image segmentation is better and has predictable execution than officially reported techniques.

Keywords
Multilevel thresholding, gray scale image segmentation, state of matter search optimization, qualitative and quantitative analysis

1. INTRODUCTION
Image segmentation plays crucial role in medical image analysis. It is frequently used to segment an image into independent regions, which preferably compares two various true objects [1]. Thresholding is a standout amongst the most important and viable methods for image segmentation, as it works taking a threshold (*th*) value so that pixels, whose intensity level is higher than *th* are marked as first class while the rest relate to second class label. At the point when the picture is portioned into two classes, the undertaking is called bi-level thresholding (*BT*) and requires only one *th* value. Then again, when pixels are isolated into more than two classes, then assignment is named as multilevel thresholding (*MT*) and requests more than one *th* value[2]. Multilevel thresholding fragments a gray level image into a few particular locales by identifying more than one threshold[3][4][5] and straightforwardness in control, thresholding methods have drawn a considerable measure of consideration amid the last couple of decades. Since multilevel thresholding is a very much inquired problem in image processing, there exist numerous techniques for deciding optimal threshold levels of the image.

In general, thresholding techniques are categorized into parametric and nonparametric[6][7]. In Parametric methodology we have to forecast the values of probability density function to model every class. The estimation procedure is tedious and computationally costly. On the other hand, the '*th*' nonparametric utilizes a few criteria, for example, between-class variance, entropy, and error rate [6][8][9] keeping in mind the end goal to check the nature of a '*th*' value. These measurements could likewise be utilized as improvement capacities since they come about as an alluring alternative because of their robustness and exactness.

Otsu's strategy [10] is one of the prevalent histogram thresholding techniques that picks an optimal threshold by expanding the between class variance, while the second strategy, proposed by Kapur et al. in [4], the threshold is controlled by amplifying the entropy of the object and background pixels. The least error thresholding technique [11] characterized a measure taking into account the presumption that the object and background pixels are ordinarily distributed and the optimal threshold is accomplished by upgrading a standard function related to Bayes risk. As a contrasting option to traditional strategies, the MT issue has additionally been investigated through swarm intelligence and evolutionary algorithms. Various authors demonstrated to deliver better solutions than techniques based on classical approach in terms of exactness, quick convergence and robustness. Various optimization techniques based methodologies are presented in the literature.

Genetic algorithm (GA), motivated on the biological evolution, has been utilized for solving segmentation problems. One intriguing case is introduced in [12], where a GA-based technique is consolidated with Gaussian models for multilevel thresholding. [13] proposed an enhanced GA for optimal multilevel thresholding where a learning procedure has been utilized to enhance the rate of convergence. Evolutionary methodologies motivated on swarm knowledge, for example, particle swarm optimization (PSO) [14] and artificial bee colony (ABC) [15], have been utilized to confront the segmentation issue. In [16], both the strategies are utilized to locate the optimal multilevel threshold values by utilizing the Kapur's entropy as objective function. In [17], the optimal threshold values are predicted by utilizing the bacterial foraging algorithm (BFA). Such strategy means to augment the Kapur's and Otsu's target objective functions by considering an arrangement of administrators which depend on the social scavenging conduct of the microorganisms Escherichia Coli. Authors [18] exhibited altered rendition of BFA for the determination of optimal threshold levels for image segmentation taking into account between class variance (Otsu's strategy). Multilevel thresholding for image segmentation, tackled taking into account harmony search algorithm (MHSA), consolidates the original version of harmonic search algorithm (HSA) based on Otsu's and Kapur's strategies are exhibited in [2]. Authors [19] proposed Cuckoo Search algorithm (CS) and a nature inspired algorithm for the determination of optimal multilevel thresholding, exclusively for satellite image segmentation in view of Kapur's entropy. Authors [20] exhibited a point by point correlation of evolutionary and swam based computational strategies for optimal multilevel thresholding selection for color images taking into account Kapur's entropy. Kotte et.al presented PSNR maximization method for better multilevel thresholding image segmentation based on Improved Differential Search Algorithm (IDSA)[21]. The convergence time of the proposed method is very high. From the literature, it is understand that numerous authors proposed their work in light of either Kapur's entropy or Otsu's between class variance as a target objective function for the optimization of multilevel thresholding levels for image segmentation.

The different great results reported by different authors to this specific engineering optimization issue persuaded us to execute a proficient optimization technique for multilevel image segmentation in light of state of matter search optimization (SMS) was proposed by [22]. Nonetheless, from the literature survey it is seen that the use of SMS to multilevel thresholding image segmentation has not been investigated. This persuaded the authors to utilize SMS as a streamlining tool for multilevel thresholding selection for image segmentation taking into account existing target objective functions i.e. Otsu's and Kapur's methods. The proposed method has been assessed by applying it on an arrangement of standard test gray scale images which offered encouraging results. So as to approve the results, subjective, objective and measurable examinations has been exhibited. Rest of the paper is composed as takes after: In section 2, mathematical treatment of bi-level and multilevel thresholding is clarified. In section 3, portrayal of target objective functions (Otsu's, Kapur's methods) maximization strategies are introduced. In section 4, execution of SMS optimization algorithm for optimal selection of multilevel thresholding is depicted. Execution results and examinations are outfitted in section 5. At long last, in section 6, finishes of the work and future scope are accounted for.

2. MULTILEVEL THRESHOLDING

Thresholding is a process in which the pixels of a gray scale image are divided into sets or classes depending on their intensity level (L). For this classification it is necessary to select a threshold value (th) and follow the simple rule of

$$C_1 \leftarrow p \; if \; 0 \leq p < th \tag{1}$$

$$C_2 \leftarrow p \; if \; th \leq p < L - 1 \tag{2}$$

Where, 'p' is one of them × n pixels of the gray scale image g that can be represented in 'L' gray scale levels$\mathrm{L} = \{0, 1, 2, \ldots, \mathrm{L}-1\}$.$C1$ and $C2$ are the classes in which the pixel 'p'can be located, while 'th' is the threshold. The rule in Eq.2 corresponds to a bi-level thresholding and can be easily extended for multiple sets:

$$C_1 \leftarrow p \; if \; 0 \leq p < th_1,$$

$$C_2 \leftarrow p \; if \; th_1 \leq p < th_2,$$

$$C_{i+1} \leftarrow p \; if \; th_i \leq p < th_{i+1},$$

$$C_n \leftarrow p \; if \; th_k \leq p < L - 1, \tag{3}$$

Where,$\{\mathrm{th}_1, \mathrm{th}_2, \ldots, \mathrm{th}_\mathrm{i}, \mathrm{th}_{\mathrm{i}+1}, \mathrm{th}_\mathrm{k}\}$ represents different thresholds. The problem of both bi-level and 'MT' is to select the 'th' values that correctly identify the classes. Although, Otsu's and Kapur's methods are well-known approaches for determining such values, both propose a different objective function which must be maximized in order to find optimal threshold values, just as it is discussed below.

3. OBJECTIVE FUNCTIONS

3.1 Otsu's between class variance

In computer vision and image processing, Otsu's technique is utilized to naturally perform grouping based image thresholding or, the diminishment of a gray level image to a binary image. The algorithm expects that the image contains two classes of pixels taking after bi-level histogram (foreground pixels and background pixels); it then ascertains the optimal threshold isolating the two classes so that their consolidated spread (intra-class difference) is negligible. The expansion of the first strategy to multi-level thresholding is alluded to as the Multi Otsu technique [10].

This is a nonparametric technique for thresholding proposed by Otsu that utilizes the most extreme fluctuation estimation of the distinctive classes as a measure to fragment the image. Taking the L intensity levels from a gray scale image, the probability distribution of the intensity values is computed as follows:

$$ph_i^c = \frac{h_i^c}{NP}$$

$$\sum_{i=1}^{NP} ph_i^c = 1\,; where, c = 1 \tag{4}$$

Where, 'i' is a specific intensity level ($0 \leq i \leq L-1$), $c$is the component of the image. 'NP' is the total number of pixels in the image. h (histogram) is the number of pixels that corresponds to the 'i' intensity level in 'c'. The histogram is normalized within a probability distribution ph_i^c. For the simplest segmentation (bilevel) two classes are defined as

$$C_1 = \frac{ph_1^c}{w_0^c(th)}, \ldots \ldots \ldots \ldots \ldots, \frac{ph_{th}^c}{w_0^c(th)}$$

$$C_2 = \frac{ph_{th+1}^c}{w_1^c(th)}, \ldots \ldots \ldots \ldots \ldots, \frac{ph_L^c}{w_1^c(th)} \tag{5}$$

Where, $\omega_0(th) \; and \; \omega_1(th)$ are probability distributions for $C_1 \; and \; C_2$ as it is shown by

$$\omega_0^c(\mathrm{th}) = \sum_{i=1}^{th} ph_i^c$$

$$\omega_1^c(\mathrm{th}) = \sum_{i=th+1}^{L} ph_i^c \tag{6}$$

It is necessary to compute mean levels $\mu_0^c \; and \; \mu_1^c$that define the classes using Eq.7. Once those values are calculated, the Otsu variance between classes σ^{2c} is calculated using Eq.8 as follows:

$$\mu_0^c = \sum_{i=1}^{th} \frac{iph_i^c}{\omega_0^c(th)}$$

$$\mu_1^c = \sum_{i=th+1}^{L} \frac{iph_i^c}{\omega_1^c(th)} \tag{7}$$

$$\sigma^{2C} = \sigma_1^{2c} + \sigma_2^{2c} \tag{8}$$

Notice that for both equations, Eq.7 and Eq.8, $c = 1$ for gray level image. In Eq.8 the number two is part of the Otsu's variance operator and does not represent an exponent in the mathematical sense. Moreover $\sigma_1^{2c} and \; \sigma_2^{2c}$in Eq.8 are the variances of $C_1 \; and \; C_2$which are defined as

$$\sigma_1^{2c} = \omega_o^c(\mu_0^c + \mu_T^c)^2$$

$$\sigma_2^{2c} = \omega_1^c(\mu_1^c + \mu_T^c)^2 \tag{9}$$

Where, $\mu_T^c = \omega_o^c\mu_0^c + \omega_1^c\,\mu_1^c$and ω_o^c+$\omega_1^c = 1$. Based on the values of $\sigma_1^{2c} and\, \sigma_2^{2c}$, Eq.10 represents the objective function:

$$J(th) = max\left(\sigma^{2C}(th)\right), 0 \le th \le L-1 \quad (10)$$

Where, $\sigma^{2c}(th)$ is the Otsu's variance for a given 'th'value. Therefore, the optimization problem is reduced to find the intensity level (th) that maximizes Eq.10. The previous description of such bi-level method can be extended for the identification of multiple thresholds. Considering 'k' thresholds, it is possible to separate the original image into 'k' classes using Eq.3; then it is necessary to compute the 'k' variances and their respective elements. The objective function $J(th)$ in Eq.10 can thus be rewritten for multiple thresholds as follows:

$$J(\boldsymbol{th}) = max\left(\sigma^{2c}(\boldsymbol{th})\right), 0 \le thi \le L-1, i = 1,2,\ldots,k \quad (11)$$

Where, $\boldsymbol{th} = [th_1, th_2, \ldots, th_{k-1}]$, is a vector containing multiple thresholds and the variances are computed through

$$\sigma^{2c} = \sum_{i=1}^{k}\sigma_i^c = \sum_{i=1}^{k}\omega_i^c\,(\mu_1^c - \mu_T^c)^2 \quad (12)$$

Here, 'i' represents the 'i^{th}'class, $w_i^c and\, \mu_j^c$ are, respectively, the probability of occurrence and the mean of a class. In 'MT', such values are obtained as

$$\omega_0^c(th) = \sum_{i=1}^{th_1} ph_i^c\,,$$

$$\omega_1^c(th) = \sum_{i=th_{i+1}}^{th_2} ph_i^c,$$

$$\omega_{k-1}^c(th) = \sum_{i=th_{k+1}}^{L} ph_i^c \quad (13)$$

And, for the mean values

$$\mu_0^c = \sum_{i=1}^{th_1}\frac{iph_i^c}{\omega_o^c(th_1)},$$

$$\mu_1^c = \sum_{i=th_{i+1}}^{th_2}\frac{iph_i^c}{\omega_o^c(th_2)}\,,$$

$$\mu_{k-1}^c = \sum_{i=th_{k+1}}^{L}\frac{iph_i^c}{\omega_1^c(th_k)}\,. \quad (14)$$

Similar to the bi-level case, for the 'MT' using the Otsu's method, 'c' corresponds to the image components, for gray scale image $c = 1$.

3.2 Kapur's entropy

Another nonparametric technique that is utilized to decide the optimal threshold value was proposed by [4]. It depends on the entropy and the probability distribution of the image histogram. The strategy intends to locate the optimal "th" that amplifies the general entropy. The entropy of image measures the compactness and distinguishableness among classes. In this sense, when the optimal "th" estimate suitably isolates the classes, the entropy has the most extreme worth. For the bi-level illustration, the target capacity of the Kapur's issue can be characterized as

$$J\,(th) = H_1^c + H_2^c, where, c = 1 \quad (15)$$

Where, the entropies$H_1\; and\; H_2$are computed using the following model:

$$H_1^c = \sum_{i=1}^{th}\frac{ph_i^c}{\omega_o^c}\ln\left(\frac{ph_i^c}{\omega_o^c}\right),$$

$$H_2^c = \sum_{i=th+1}^{L}\frac{ph_i^c}{w_1^c}\ln\left(\frac{ph_i^c}{\omega_1^c}\right). \quad (16)$$

$ph_i^c$is the probability distribution of the intensity levels which is obtained using Eq.4. $\omega_0(th)\; and\; \omega_1(th)$are probability distributions for $C_1\; and\; C_2$. ln(·) stands for the natural logarithm. Similar to the Otsu's method, the entropy-based approach can be extended for multiple threshold values for such case, it is necessary to divide the image into 'k' classes using the similar number of thresholds. Under such conditions, the new objective function is defined as:

$$J\,(\boldsymbol{th}) = max(\textstyle\sum_{i=1}^{k} H_i^c)\; where, c = 1 \quad (17)$$

Where, $\boldsymbol{th} = [th_1, th_2, \ldots, th_{k-1}]$is a vector that contains the multiple thresholds. Each entropy is computed separately with its respective 'th' value, so Eq.18 is expanded for 'k' entropies:

$$H_1^c = \sum_{i=1}^{th_1}\frac{ph_i^c}{\omega_o^c}\ln\left(\frac{ph_i^c}{\omega_o^c}\right),$$

$$H_2^c = \sum_{i=th_{i+1}}^{th_2}\frac{ph_i^c}{\omega_1^c}\ln\left(\frac{ph_i^c}{\omega_1^c}\right),$$

$$H_k^c = \sum_{i=th_{k+1}}^{L}\frac{ph_i^c}{\omega_{k-1}^c}\ln\left(\frac{ph_i^c}{\omega_{k-1}^c}\right). \quad (18)$$

The values of the probability occurrence $(\omega_0^c, \omega_1^c, \ldots\ldots\ldots \omega_{k-1}^c)$of the '$k$' classes are obtained using Eq.13 and the probability distributionph_i^c using Eq.7. Finally, it is necessary to use Eq.3 to separate the pixels into the corresponding classes.

4. STATE OF MATTER SEARCH ALGORITHM

State of matter search is novel and efficient nature inspired evolutory algorithm for solving global optimization problems. The SMS algorithm is based on the simulation of states of matter phenomenon. In SMS, individuals emulate molecules which interact to each other by using evolutionary operations based on the physical principles of the thermal-energy motion mechanism. Such operations allow the increase of the population diversity and avoid the concentration of particles within a local minimum. The proposed approach combines the use of the defined operators with a control strategy that modifies the parameter setting of each operation during the evolution process. In contrast to other approaches that enhance traditional EA (evolutory algorithms) by incorporating some procedures for balancing the exploration–exploitation rate, the proposed algorithm naturally delivers such property as a result of mimicking the states of matter phenomenon. The algorithm is devised by considering each state of matter at one different exploration–exploitation ratio. Thus, the evolutionary process is divided into three stages which emulate the three states of matter: gas, liquid and solid. At each state, molecules (individuals) exhibit different behavior. Beginning from the gas state (pure exploration), the algorithm modifies the intensities of exploration and exploitation until the solid state (pure exploitation) is reached. As a result, the approach can substantially improve the balance between exploration–exploitation, yet preserving the good search capabilities of an evolutionary approach [22].

4.1 SMS implementation procedure

The overall SMS algorithm is comprised of three phases corresponding to the three states of matter: the gas, the liquid and the solid state. Each phase has its own behavior. In the gas phase exploration is intensified whereas in liquid phase a mild transition between exploration and exploitation is executed. Finally, in the solid phase, solutions are refined by emphasizing the exploitation process.

At each phase, the same operations are implemented. However, depending on which phase is referred, they are employed considering a different parameter configuration. The procedure in each phase is shown in algorithm steps for SMS. Such procedure is composed of nine steps and maps the current population Pk to a new population Pk +1. The algorithm receives the current population Pk as input and the configuration parameters α, β, ρ, and H will help to yield the new population Pk +1.

4.2 Steps for implementation of SMS algorithm

Step 1: Initialization of optimization problem and algorithm parameters

Initialize population size (*Pop*), *N*, *α, β, ρ*, *H* for all phases, *D*, *maxit*, *Phase* and limits for threshold levels.

Step 2: Initialization of Population of molecules (Generation of random solution)

The (*P*) is generated randomly; where, elements of *P* represent the sets of decision variables (threshold levels). *P* matrix is represented by:

$$P = \begin{bmatrix} x_1^1 & x_2^1 & \cdots & x_{N-1}^1 & x_N^1 \\ x_1^2 & x_2^2 & \cdots & x_{N-1}^2 & x_N^2 \\ \vdots & \vdots & \vdots & \vdots & \vdots \\ x_1^{Pop-1} & x_2^{Pop-1} & \cdots & x_{N-1}^{Pop-1} & x_N^{Pop-1} \\ x_1^{Pop} & x_2^{Pop} & \cdots & x_{N-1}^{Pop} & x_N^{Pop} \end{bmatrix} \quad (19)$$

$$x_j^i = x_j^{min} + \left(x_j^{max} - x_j^{min}\right) * rand \quad (20)$$

$$[th_1^1\ th_2^1\ th_{N-1}^1\ th_N^1] = P_1 \quad (21)$$

Where, *N* is the number of decision variables (dimension of the problem), x_j^i represents parameter output, i.e., i^{th} population of j^{th} parameter, which is generated randomly between the limits, as x_j^{max} and x_j^{min} are the j^{th} parameter maximum and minimum limits and *rand*() is a random number between 0 and 1.

Step 3: Evaluate the objective function and record the best solution of the population *P*

Calculate the objective value for each initial solution using Eq. 11 and Eq.17. Record the *gbest* solution so far.

$$P \in \{P\} and\ f(P^{best}) = \max\{f(P_1), f(P_1), \ldots\ldots, f(P_{Pop})\} \quad (22)$$

Step 4: Calculate V_{init} (initial velocity of each molecule) and *r* (collision radius)

$$v_{init} = \frac{\sum_{j=1}^{N}\left(x_j^{max} - x_j^{min}\right)}{N} * \beta \qquad r = \frac{\sum_{j=1}^{N}\left(x_j^{max} - x_j^{min}\right)}{N} * \alpha \quad (23)$$

Where, β ϵ [0, 1] and α ϵ [0, 1]

Step 5: Compute new molecules (solutions) by using the direction vector operator Eq. 24

For ($i = 1; i < Pop+1; i{+}{+}$)

$$a_i = {\left(P^{best} - P_i\right)} \Big/ {\|P^{best} - P_i\|}$$

For ($j = 1; j < N+1; j{+}{+}$)

$$dir_{i,j}^{k+1} = dir_{i,j}^{k} * \left(1 - \frac{itr}{maxit}\right) * 0.5 + a_{i,j} \quad (24)$$

$$v_{i,j} = dir_{i,j}^{k+1} * v_{init} \quad (25)$$

$$P_{i,j}^{k+1} = P_{i,j}^{k} + v_{i,j}\ .rand.\rho.\left(x_j^{max} - x_j^{min}\right) \quad (26)$$

End for *j*

End for *i*

Step 6: Solve collisions by using Collision operator Eq. 27

For ($i = 1; i < Pop+1; i{+}{+}$)

For ($j = 1; j < N+1; j{+}{+}$)

$$if\left(\left(\|P_i - P_j\| < r\right) and (i \neq j)\right) \quad (27)$$

$$t = dir_i$$

$$dir_i = dir_j$$

$$dir_j = t$$

End for *if*

End for *j*

End for *i*

Step 7: Generate new random positions by using the random position operator Eq. 28

For ($i = 1; i < Pop+1; i{+}{+}$)

$$if\ (r_m < H) then; where\ r_m \in rand$$

For ($j = 1; j < N+1; j{+}{+}$)

$$P_{i,j}^{k+1} = \begin{cases} x_j^{min} + \left(x_j^{max} - x_j^{min}\right).rand & with\ probability\ H \\ P_{i,j}^{k+1} & with\ probability\ (1-H) \end{cases} \quad (28)$$

End for *j*

End for *if*

End for *i*

Step 8: Initiate change of phase, evaluate the new solution *P* and update *gbest*

Calculate the objective value based on new solution P using Eq. 11 and Eq.17 and select the best solution in new P. If the new solution is better than the previous solution then record the best solution (*gbest*) so far otherwise discard new solution and preserve the previous solution.

Step 9: Stopping criterion

If the maximum number of iterations is reached, computation is terminated. Otherwise, Step 4 to Step 8 is repeated.

The detailed implementation flow chart for SMS algorithm in the context of image enhancement is given in Figure 1.

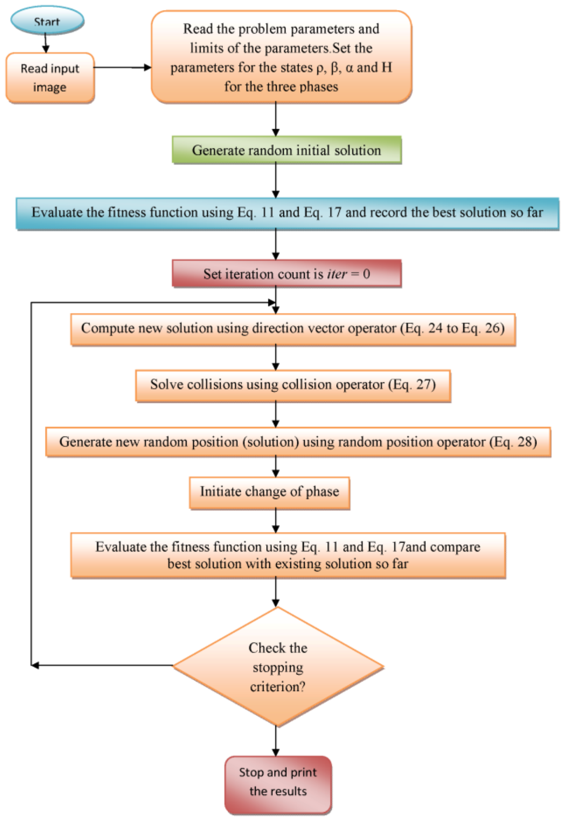

Fig 1: Implementation flow chart for SMS algorithm

5. RESULTS AND DISCUSSIONS

In this section, the proposed strategy taking into account productive outcome is accepted through applying it on segmentation of standard ten test images. Each image of size 256*256, 8-bit gray level. Algorithm parameter determination assumes a noteworthy part of any optimization algorithm as far as execution. Consequently, parameter tuning is essential for streamlining methods before execution. The values doled out for these parameters are chosen by the quantity of trails on the execution of the proposed technique/algorithm. The parameter depiction and doled out values for SMS algorithm are outfitted in Table 1. The proposed technique in light of novel optimization algorithm has been contrasted with well-existing methodologies/algorithms in the literature. All simulations are self-developed MATLAB codes utilizing MATLAB R2010a on an Intel Core i5-2400 Duo 3.10 GHz processor with 4 GB RAM.

To demonstrate the effectiveness of proposed approach, the following two different strategies are taken into account:

1. Maximization of Between Class Variance (Otsu method)
2. Maximization of Entropy (Kapur method)

Table 1 Assigned values for SMS algorithm parameters

Algorithm	Parameter	Description	Assigned value
SMS	*Pop*	Size of population	50
	N	Dimension of the problem	Dependent on ‘k’
	maxit	Maximum number of iterations	1000 for Otsu/ 500 for Kapur
	β	Movement operator	[0.9, 0.5, 0.1]
	α	Collision operator	[0.3, 0.05, 0.0]
	H	Threshold operator	[0.9, 0.2, 0.0]
	ρ	Direction operator	[0.85, 0.35, 0.1]

The aim of the proposed approach is to select best thresholding values and higher objective values with fast convergence. Subjective analysis and comparison of considered strategies for test images like cameraman, lena, baboon, hunter and butterfly has been presented in Figure 2 to Figure 6.

Each Figure gives detailed information about input image, thresholded output image at various optimal thresholds with related convergence characteristics of the proposed approach. From Figure 2 to Figure 6 it is observed that the Kapur-SMS approach was not so good to segment the given input image at threshold level two for all images. Besides, Otsu-SMS has been successful in segmentation of given input image at all considered threshold levels. Table 2 presents the comparison of optimal threshold values obtained by Otsu-SMS approach with various well existing optimization approaches. Table 3 presents the comparison of objective function values obtained by Otsu-SMS approach with existing optimization approaches such as IDSA, HAS, BF and PSO. Form Table 3 it is observed that the values of objective function values obtained by Otsu-SMS is appreciably higher than existing approaches at all threshold levels for all the images and the marginal difference is high in case of number of threshold levels is five. Similarly, Table 4 shows the comparison of optimal threshold values obtained by Kapur-SMS approach with various well existing optimization approaches. Table 5 presents the comparison of objective function values obtained by Kapur-SMS approach with existing optimization approaches such as IDSA, HAS, BF and PSO. In case of Kapur-SMS approach the objective function value at threshold level two is marginal or approximately closer to existing approaches. However, the objective function value obtained by Kapur-SMS at remaining threshold levels is considerably higher than existing methods. Statistical comparison of image quality metric such as PSNR obtained by Otsu-SMS and Kapur-SMS with existing approaches has been furnished in Table 6. From Table 6 it is clear that the PSNR values obtained by Otsu-SMS and Kapur-SMS are superior to existing approaches for all the images. An important factor for the analysis of performance of optimization algorithms is convergence time. A detailed comparison of computational time of various approaches has been presented in Table 7. Form Table 7 it is observed that the average time of convergence of SMS for the optimal multilevel thresholding image segmentation is between 3s to

5s depends on number of threshold levels which is considerably small.

6. CONCLUSIONS

This paper presents a fast and efficient optimization approach for the optimal selection of threshold levels for gray level image segmentation. Two well existing objective functions maximization of entropy and between class variance are considered for the evaluation of proposed approach. Detailed qualitative, quantitative and statistical analysis of results of proposed approach has been presented. Form the obtained results it is concluded that the Otsu-SMS and Kapur-SMS approaches were outperformed than existing methods in terms of quality and convergence. It is noticed that the clarity and information of segmented image increased with increase in number of thresholds levels. Development of hybrid optimization algorithms and novel objective functions may give better results is future scope of the work.

7. REFERENCES

[1] S. Patra, R. Gautam, A. Singla, A novel context sensitive multilevel thresholding for image segmentation, Appl. Soft Comput. J. 23 (2014) 122–127. doi:10.1016/j.asoc.2014.06.016.

[2] D. Oliva, E. Cuevas, G. Pajares, D. Zaldivar, M. Perez-Cisneros, Multilevel thresholding segmentation based on harmony search optimization, J. Appl. Math. 2013 (2013). doi:10.1155/2013/575414.

[3] P. Smith, D.B. Reid, C. Environment, L. Palo, P. Alto, P.L. Smith, A Threshold Selection Method from Gray-Level Histograms, IEEE Trans. Syst. Man. Cybern. 9 (1979) 62–66. doi:10.1109/TSMC.1979.4310076.

[4] J.N. Kapur, P.K. Sahoo, a. K.C. Wong, A new method for gray-level picture thresholding using the entropy of the histogram, Comput. Vision, Graph. Image Process. 29 (1985) 140. doi:10.1016/S0734-189X(85)90156-2.

[5] J. Kittler, J. Illingworth, Minimum error thresholding, Pattern Recognit. 19 (1986) 41–47. doi:10.1016/0031-3203(86)90030-0.

[6] N.R. Pal, S.K. Pal, A review on image segmentation techniques, Pattern Recognit. 26 (1993) 1277–1294. doi:10.1016/0031-3203(93)90135-J

[7] P.. Sahoo, S. Soltani, a. K.. Wong, A survey of thresholding techniques, Comput. Vision, Graph. Image Process. 41 (1988) 233–260. doi:10.1016/0734-189X(88)90022-9.

[8] C.-C. Chang, L.-L. Wang, A fast multilevel thresholding method based on lowpass and highpass filtering, Pattern Recognit. Lett. 18 (1997) 1469–1478. doi:10.1016/S0167-8655(97)00134-7.

[9] Q. Huang, W. Gao, W. Cai, Thresholding technique with adaptive window selection for uneven lighting image, Pattern Recognit. Lett. 26 (2005) 801–808. doi:10.1016/j.patrec.2004.09.035.

[10] P. Smith, D.B. Reid, C. Environment, L. Palo, P. Alto, P.L. Smith, Smith et al. - 1979 - A Tlreshold Selection Method from Gray-Level Histograms, 20 (1979) 62–66. doi:10.1109/TSMC.1979.4310076.

[11] S. Cho, R. Haralick, S. Yi, Improvement of kittler and illingworth's minimum error thresholding, Pattern Recognit. 22 (1989) 609–617. doi:10.1016/0031-3203(89)90029-0.

[12] C.-C. Lai, D.-C. Tseng, A Hybrid Approach Using Gaussian Smoothing and Genetic Algorithm for Multilevel Thresholding, Int. J. Hybrid Intell. Syst. 1 (2004) 143–152. http://content.iospress.com/articles/international-journal-of-hybrid-intelligent-systems/his015.

[13] P.-Y. Yin, A fast scheme for optimal thresholding using genetic algorithms, Signal Processing. 72 (1999) 85–95. doi:10.1016/S0165-1684(98)00167-4.

[14] J. Kennedy, R. Eberhart, Particle swarm optimization, Neural Networks, 1995. Proceedings., IEEE Int. Conf. 4 (1995) 1942–1948 vol.4. doi:10.1109/ICNN.1995.488968.

[15] D. Karaboga, An idea based on Honey Bee Swarm for Numerical Optimization, Tech. Rep. TR06, Erciyes Univ. (2005) 10. doi:citeulike-article-id:6592152.

[16] B. Akay, A study on particle swarm optimization and artificial bee colony algorithms for multilevel thresholding, Appl. Soft Comput. 13 (2012) 3066–3091. doi:10.1016/j.asoc.2012.03.072.

[17] P.D. Sathya, R. Kayalvizhi, Optimal multilevel thresholding using bacterial foraging algorithm, Expert Syst. Appl. 38 (2011) 15549–15564. doi:10.1016/j.eswa.2011.06.004.

[18] P.D. Sathya, R. Kayalvizhi, Modified bacterial foraging algorithm based multilevel thresholding for image segmentation, Eng. Appl. Artif. Intell. 24 (2011) 595–615. doi:10.1016/j.engappai.2010.12.001.

[19] A.K. Bhandari, V.K. Singh, A. Kumar, G.K. Singh, Cuckoo search algorithm and wind driven optimization based study of satellite image segmentation for multilevel thresholding using Kapur's entropy, Expert Syst. Appl. 41 (2014) 3538–3560. doi:10.1016/j.eswa.2013.10.059.

[20] T. Kurban, P. Civicioglu, R. Kurban, E. Besdok, Comparison of evolutionary and swarm based computational techniques for multilevel color image thresholding, Appl. Soft Comput. J. 23 (2014) 128–143. doi:10.1016/j.asoc.2014.05.037.

[21] S. Kotte, P.R. Kumar, S. Kumar, An efficient approach for optimal multilevel thresholding selection for gray scale images based on improved differential search algorithm, Ain Shams Eng. J. (2016). doi:10.1016/j.asej.2016.06.007.

[22] E. Cuevas, An optimization algorithm inspired by the States of Matter that improves the balance between exploration and exploitation, 40 (2014) 256–272.

8. APPENDIX

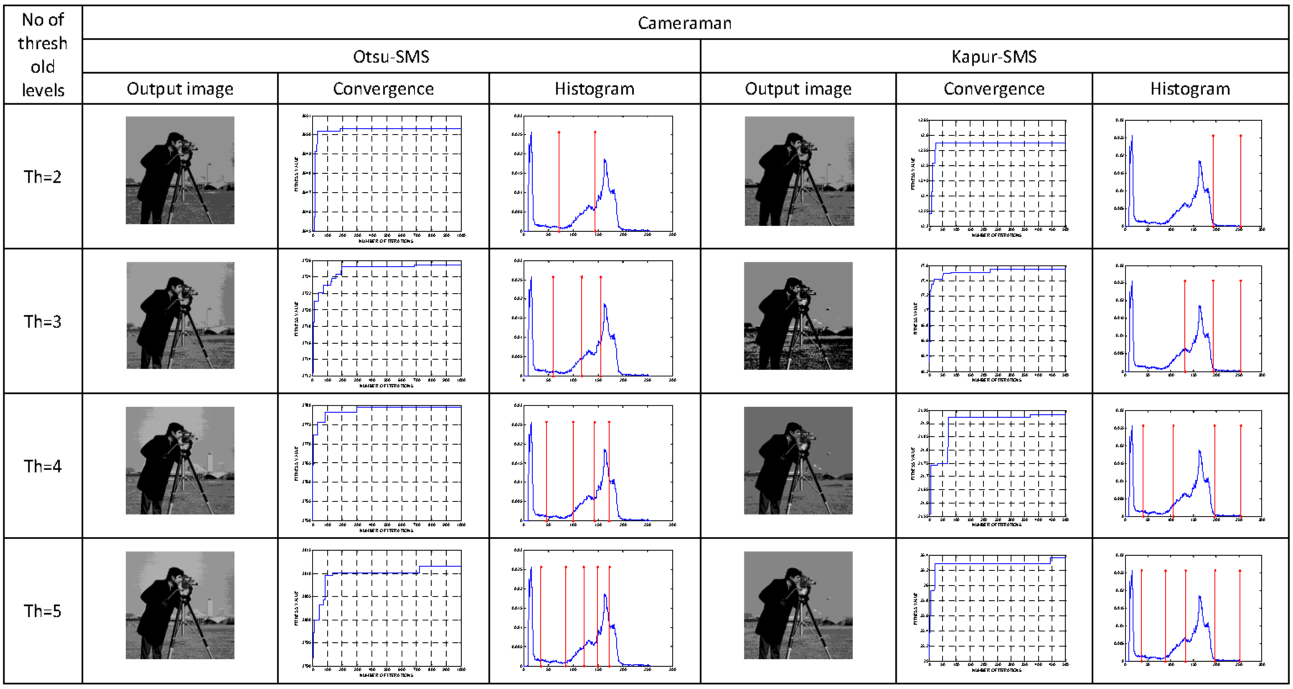

Fig 2: Implementation results of Otsu-SMS and Kapur-SMS over Cameraman Image

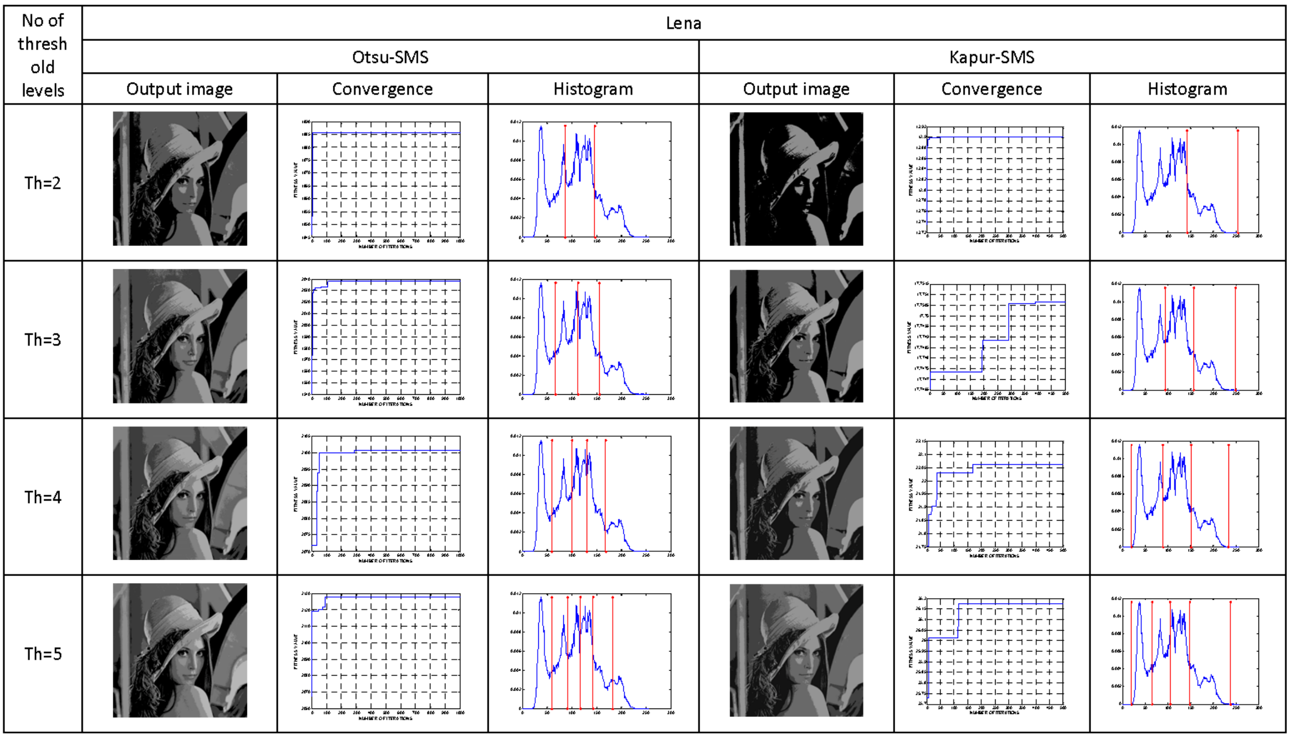

Fig 3: Implementation results of Otsu-SMS and Kapur-SMS over Lena Image

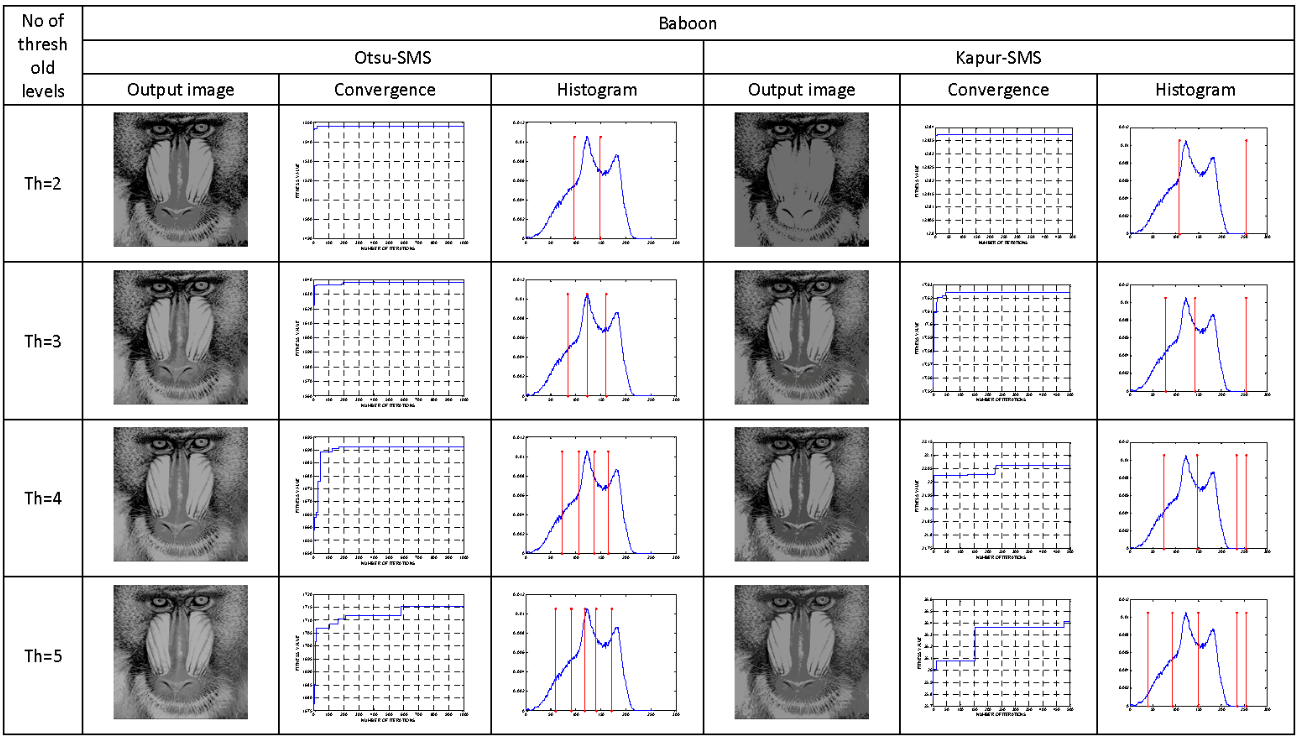

Fig 4: Implementation results of Otsu-SMS and Kapur-SMS over Baboon Image

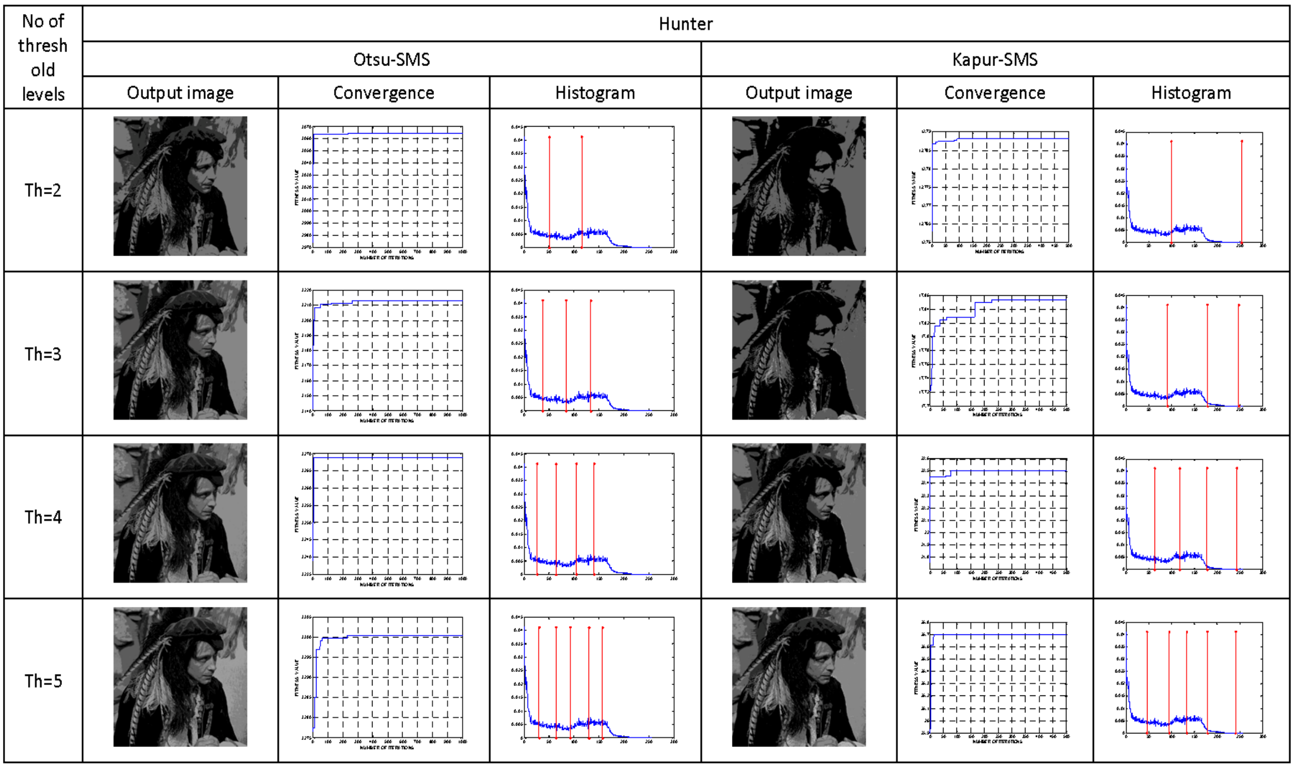

Fig 5: Implementation results of Otsu-SMS and Kapur-SMS over Hunter Image

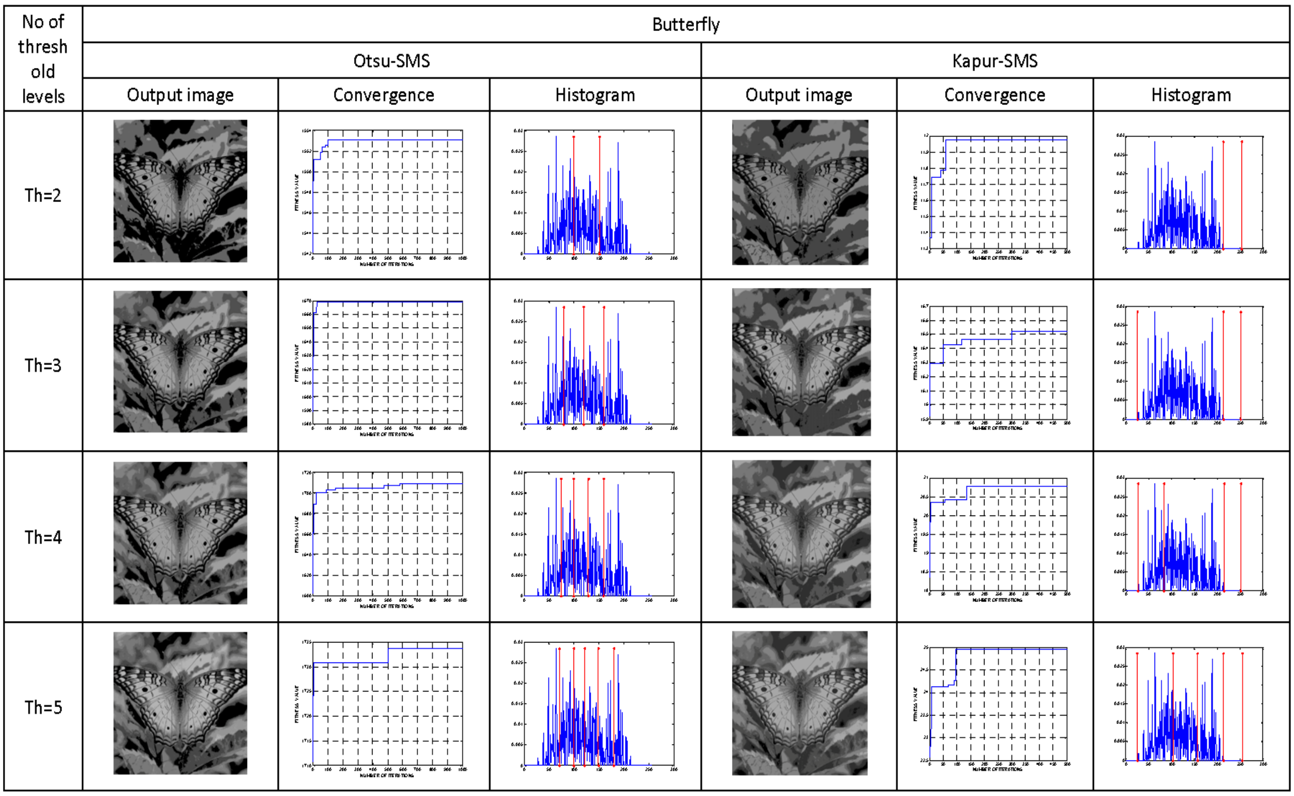

Fig 6: Implementation results of Otsu-SMS and Kapur-SMS over butterfly image

Table 2 Comparison of optimal threshold values obtained by Otsu's method using various optimization algorithms

Input Image Name	k	SMS	IDSA[21]	HSA[2]	BF[17]	PSO[17]
Cameraman	2	71,144	70,144	70, 144	70,143	71,143
	3	59,117,155	59,119,156	59, 119, 156	61,118,155	71,134,166
	4	46, 99,143,172	38,93,140,172	42, 95, 140, 170	48,104,142,170	65,121,147,172
	5	35,85,121,149,173	31,83,135,165,209	36, 82, 122, 149, 173	40,86,125,151,174	45,78,121,146,172
Lena	2	86,146	86,146	91, 150	92,151	94,152
	3	67,112,156	59,115,180	79, 125, 170	79,125,170 7	9,127,170
	4	60,100,130,168	65,114,138,181	73, 112, 144, 179	76,117,151,182	78,112,134,175
	5	59, 91,117,143,182	61,96,128,167,242	71, 107, 134, 158,186	66,92,122,149,183	79,110,140,167,188
Baboon	2	97,149	97,150	97, 149	98,150 9	6,149
	3	84,123,161	73 125 162	85, 125, 161	84,126,159	85,126,166
	4	72,106,137,165	33,84,124,161	71, 105, 136, 167	77,109,139,169	79,105,140,174
	5	60, 91,118,141,172	37,83,111,151,168	66, 97, 123, 147, 173	70,99,127,154,177	74,104,134,161,180
Hunter	2	51,116	51,116	51, 116	51,117	52,116
	3	37,85,134	36,86,135	36, 86, 135	36,86,135	39,86,135
	4	26,64,105,141	30,72,111,146	27, 65, 104, 143	31,80,120,152	36,84,130,157
	5	30,64,92,130,157	36,86,112,135,256	22, 53, 88, 112, 152	31,73,109,141,178	37,85,125,154,177
Butterfly	2	99,151	97,153	99, 151	99,151	99,150
	3	80,119,160	72,100,145	82, 119, 160	78,117,162	79,119,164
	4	74, 99,129,159	74,115,159,201	71, 102, 130, 163	75,105,135,165	80,113,145,177
	5	71,100,121,149,180	63,111,131,150,192	62, 77, 109, 137, 167	76,104,129,154,180	75,106,129,157,180

Table 3 Comparison of Objective function values obtained by Otsu's method using various optimization algorithms

Input Image Name	k	SMS	IDSA[21]	DSA[21]	HSA[2]	BF[17]	PSO[17]	GA[17]
Cameraman	2	3652.878	3651.9	3651.1	3651.9	3609.499	3609.370	3609.076
	3	3728.178	3727.2	3727	3727.4	3682.569	3677.178	3643.215
	4	3793.478	3792.5	3785.1	3782.4	3737.120	3722.644	3710.731
	5	3855.178	3854.2	3834	3813.7	3769.223	3764.957	3755.552
Lena	2	1980.378	1979.4	1967	1964.4	1961.555	1961.414	1960.960
	3	2165.678	2164.7	2141	2131.4	2128.070	2127.777	2126.410
	4	2213.178	2212.2	2199	2194.9	2189.026	2180.686	2173.714
	5	2257.278	2256.3	2228	2218.7	2215.609	2212.555	2196.274
Baboon	2	1552.178	1551.2	1550	1548.1	1548.012	1547.997	1547.658
	3	1669.378	1668.4	1645	1638.3	1637.007	1635.362	1633.522
	4	1703.178	1702.2	1695	1692.1	1690.722	1684.336	1677.705
	5	1755.178	1754.2	1729	1717.5	1716.728	1712.958	1699.390
Hunter	2	3065.178	3064.2	3054.2	3054.2	3064.118	3064.068	3064.015
	3	3214.378	3213.4	3213.4	3213.4	3213.446	3212.058	3211.794
	4	3283.778	3282.8	3271	3269.5	3266.350	3257.176	3231.131
	5	3368.478	3367.5	3345	3308.1	3291.133	3276.317	3244.738
Butterfly	2	1579.178	1578.2	1563	1553.0	1553.073	1553.068	1552.412
	3	1696.378	1695.4	1673	1669.2	1667.280	1665.758	1662.696
	4	1739.678	1738.7	1724	1708.3	1707.099	1702.906	1696.694
	5	1765.278	1764.3	1740	1728.0	1733.031	1730.787	1716.042

Table 4 Comparison of optimal threshold values obtained by Kapur's method using various optimization algorithms

Input Image Name	k	SMS	IDSA[21]	HSA[2]	BF[17]	PSO[17]
Cameraman	2	193,254	128,196	128,196	116,196	115,196
	3	132,193,254	44,103,196	44,103,196	95,139,193	96,138,191
	4	40,106,197,255	43,96,147,198	44,96,146,196	42,96,139,200	77,116,151,202
	5	36,89,133,198,252	24,61,98,147,195	24,60,98,146,196	42,84,115,150,198	64,95,121,156,198
Lena	2	142,254	96,163	96,163	97,164	99,165
	3	94,157,249	23,96,163	23,96,163	88,142,188	86,151,180
	4	20,89,151,233	23,81,127,170	23,80,125,173	74,114,149,184	92,129,162,191
	5	20,65,105,148,238	22,71,108,145,180	23,71,109,144,180	64,95,128,163,194	74,115,145,170,197

Baboon	2	108,255	79,143	79,143	81,144	76,144
	3	78,143,254	79, 143, 231	79,143, 231	53,112,150	72,130,181
	4	74,147,233,255	44, 97, 153, 230	44, 98, 152, 231	39,90,131,168	65,121,153,180
	5	39,93,149,234,255	33, 75, 114, 158,232	33, 74, 114, 159,231	38,79,113,148,180	73,110,142,166,192
Hunter	2	99,255	92,179	92, 179	85,179	83,179
	3	91,179,248	59,117,179	59, 117, 179	57,104,175	85,128,166
	4	63,118,178,244	45, 89, 132, 179	44, 89, 133, 179	50,98,139,180	74,131,174,200
	5	46,95,134,179,241	44,89, 132, 179, 221	44, 89, 133, 179, 222	49,93,137,179,222	90,120,164,190,219
Butterfly	2	213,254	27,213	27, 213	97,144	95,141
	3	26,214,251	27,120, 213	27, 120, 213	75,109,154	63, 96,103,
	4	27,84,215,252	27, 97, 144, 212	27, 96, 144, 213	73,97,127,157	71,113,162,184
	5	25,103,157,214,255	27,82, 118, 151, 212	27, 83, 118, 152, 213	74,97,120,144,167	92,116,142,157,182

Table 5 Comparison of objective values obtained by Kapur's method using various optimization algorithms

Input Image Name	k	SMS	IDSA[21]	HSA[2]	BF[17]	PSO[17]
Cameraman	2	14.5915	14.584	14.584	12.2646	12.2595
	3	17.5130	16.007	16.007	15.2507	15.2110
	4	21.8986	19.686	19.586	18.4066	18.0009
	5	26.2731	23.753	23.553	21.2111	20.9631
Lena	2	12.8999	12.334	12.334	12.3470	12.3459
	3	17.7490	16.955	16.995	15.2206	15.1336
	4	22.0405	18.319	18.089	17.9333	17.8388
	5	26.1341	20.429	20.349	20.6099	20.4427
Baboon	2	12.8372	12.984	12.984	12.2164	12.2134
	3	17.6236	16.745	16.745	15.2114	15.0088
	4	22.0459	18.925	18.815	17.9992	17.5743
	5	26.2779	21.647	21.662	20.7200	20.2245
Hunter	2	12.7879	12.349	12.349	12.3733	12.3708
	3	17.8420	16.838	16.838	15.5533	15.1286
	4	22.4961	19.352	19.218	18.3819	18.0401
	5	26.6963	21.624	21.563	21.2565	20.5339
Butterfly	2	11.9402	10.470	10.470	10.4749	10.4743
	3	16.4652	13.628	13.628	12.7546	12.3130
	4	20.6659	15.425	15.314	14.8777	14.2317
	5	24.7890	17.812	17.756	16.8282	16.3374

Table 6 Comparison of PSNR values obtained in proposed method, Otsu's and Kapur's method using SMS

Input Image Name	k	PSNR (dB)													
		Otsu's Method							Kapur's Method						
		SMS	IDSA[21]		HSA[2]	BF[17]	PSO[17]		SMS	IDSA[21]		HSA[2]	BF[17]	PSO[17]	
Cameraman	2	17.3241	17.2491		17.247	17.048	17.033		14.079	13.626		13.626	11.941	12.259	
	3	20.3023	20.2165		20.211	17.573	19.219		14.913	14.460		14.460	14.827	15.211	
	4	21.6670	21.2508		21.533	20.523	21.254		21.577	21.124		20.153	17.166	18.000	
	5	23.4517	23.3124		23.282	21.369	22.095		21.893	20.84		20.661	19.795	20.963	
Lena	2	15.2389	15.2389		15.401	15.040	15.077		15.091	14.638		14.638	12.334	12.345	
	3	18.2208	17.7239		17.427	17.304	17.276		16.671	16.218		16.218	14.995	15.133	
	4	19.8082	18.8069		18.763	17.920	18.305		19.995	19.542		19.287	17.089	17.838	
	5	20.5840	19.7791		19.443	18.402	18.770		21.667	21.214		21.047	19.549	20.442	
Baboon	2	15.4227	15.4198		15.422	15.304	15.088		16.469	16.016		16.016	12.184	12.213	
	3	17.8546	18.3130		17.709	17.505	17.603		16.469	16.016		16.016	14.745	15.008	
	4	20.1668	20.3641		20.289	18.708	19.233		18.974	18.521		18.485	16.935	17.574	
	5	22.3062	21.9156		21.713	20.203	20.526		20.977	20.524		20.507	19.662	20.224	
Hunter	2	17.8950	17.8950		16.299	17.088	17.932		15.659	15.206		15.206	12.349	12.370	
	3	20.3748	20.3508		18.359	20.045	19.940		18.953	18.500		18.500	14.838	15.128	
	4	22.1772	22.1550		20.737	20.836	21.128		21.567	21.114		21.065	17.218	18.040	
	5	23.4901	21.6472		22.310	21.284	22.026		21.697	21.244		21.086	19.563	20.533	
Butterfly	2	13.9348	13.9610		13.934	13.007	13.092		8.646	8.1930		8.1930	10.470	10.474	
	3	16.9622	17.7078		16.932	15.811	17.261		13.868	13.415		13.415	11.628	12.313	
	4	18.6771	18.9879		19.259	17.104	17.005		17.277	16.824		16.725	13.314	14.231	
	5	22.6969	21.8066		21.450	18.593	18.099		19.987	19.534		19.413	15.756	16.337	

Table 7 Comparison of CPU time (in seconds) for various methods

Input Image Name	k	Otsu's Method					Kapur's Method				
		SMS	IDSA[21]	BF[17]	PSO[17]		SMS	IDSA[21]	BF[17]	PSO[17]	
Cameraman	2	3.0505	2.9345	3.0625	3.4844		4.767	5.8813	7.7813	8.4844	
	3	3.6755	3.5595	3.6875	4.125		4.988	6.372	8.272	9.0625	
	4	4.2224	4.1064	4.2344	4.7406		6.374	6.6938	8.5938	9.125	
	5	4.6599	4.5439	4.6719	5.2656		5.921	7.3969	9.2969	10.1094	
Lena	2	3.2849	3.1689	3.2969	3.5781		5.645	5.3063	7.2063	7.8594	
	3	3.8161	3.7001	3.8281	4.4031		6.114	5.706	7.606	8.3594	
	4	4.2068	4.0908	4.2188	4.75		5.786	6.6	8.5	9.1719	
	5	4.7693	4.6533	4.7813	5.2031		5.767	6.9125	8.8125	9.4063	
Baboon	2	3.2693	3.1533	3.2813	3.8469		5.651	5.725	7.625	8.0016	
	3	3.7849	3.6689	3.7969	4.3125		6.198	6.3824	8.2824	8.7188	
	4	4.2693	4.1533	4.2813	4.9063		6.649	6.8188	8.7188	9.1084	
	5	4.8086	4.6926	4.8206	5.3281		6.874	7.2875	9.1875	9.7813	
Hunter	2	3.2224	3.1064	3.2344	3.8438		4.565	5.4594	7.3594	8	
	3	3.8943	3.7783	3.9063	4.4844		6.279	6.3813	8.2813	8.7031	

	4	4.1755	4.0595	4.1875	4.8125		6.237	6.8344	8.7344	9.0313	
	5	4.8008	4.6848	4.8128	5.3031		7.363	7.725	9.625	10.1406	
Butterfly	2	3.238	3.122	3.25	3.5313		5.127	5.2719	7.1719	7.7188	
	3	3.7068	3.5908	3.7188	4.1875		6.358	5.9906	7.8906	8.5469	
	4	4.1911	4.0751	4.2031	4.8281		6.294	6.5688	8.4688	9	
	5	5.0505	4.9345	5.0625	5.4594		7.150	6.7563	8.6563	9.3813	