

Chapter 2

Image Segmentation Based on Differential Evolution Optimization

Abstract Threshold selection is a critical preprocessing step for image analysis, pattern recognition and computer vision. On the other hand Differential Evolution (DE) is a heuristic method for solving complex optimization problems, yielding promising results. DE is easy to use, keeps a simple structure and holds acceptable convergence properties and robustness. In this chapter, an automatic image multi-threshold approach based on differential evolution optimization is presented. Hereby the segmentation process is considered to be similar to an optimization problem. First, the algorithm approximates the 1-D histogram of the image using a mixture of Gaussian functions whose parameters are calculated using the differential evolution method. Each Gaussian function approximating the histogram represents a pixel class and therefore a threshold point. The resultant approach is not only computationally efficient but also does not require prior assumptions whatsoever about the image. The method is likely to be most useful for applications considering different and perhaps initially unknown image classes. Experimental results demonstrate the algorithm's ability to perform automatic threshold selection while preserving main features from the original image.

2.1 Introduction

Several image processing applications aim to detect and mark remarkable features which in turn might be used to perform high-level tasks. In particular, image segmentation seeks to group pixels within meaningful regions. Commonly, gray levels belonging to the object are substantially different from the gray levels featuring the background. Thresholding is thus a simple but effective tool to isolate objects of interest from the background. Its applications include several classics such as document image analysis, whose goal is to extract printed characters [1, 2], logos, graphical content, or musical scores; also it is used for map processing which aims to locate lines, legends, and characters [3]. It is also used for scene processing, aiming for object detection and marking [4]; Similarly, it has been employed to quality inspection for materials [5, 6], discarding defective parts.

Thresholding selection techniques can be classified into two categories: bi-level and multi-level. In bi-level thresholding, one limit value is chosen to segment an image into two classes: one represents the object and the other represents the background. When an image is composed of several distinct objects, multiple threshold values have to be selected for proper segmentation. This is called multilevel thresholding.

A variety of thresholding approaches have been proposed for image segmentation, including conventional methods [7–10] and intelligent techniques such as in [11, 12]. Extending the algorithm to a multilevel approach may arise some inconveniences: (i) they may have no systematic and analytic solution when the number of classes to be detected increases and (ii) the number of classes is either difficult to be predicted or must be pre-defined. However, this parameter is unknown for many real applications.

In order to solve these problems, an alternative approach using a optimization algorithm based on differential evolution for multilevel thresholding is presented in this chapter. In the traditional multilevel optimal thresholding, the intensity distributions belonging to the object or to the background pixels are assumed to follow some Gaussian probability function; therefore a combination of probability density functions is usually adopted to model these functions. The parameters in the combination function are unknown and the parameter estimation is typically assumed to be a nonlinear optimization problem [13]. The unknown parameters that give the best fit to the processed histogram are determined by using the so-called differential evolution optimization algorithm. DE is a relatively new population-based evolutionary computational model [14]. An important difference among other evolutionary computational techniques, such as genetic algorithms (GA), is that the GA relies on the crossover operator which provides the exchange of information required to build better solutions. DE algorithm fully relies on the mutation operation as its central procedure. DE is well known for its ability to efficiently and adaptively explore large search spaces and has two advantages: (1) DE has shown a faster convergence rate than other evolutionary algorithms on some problems [15], and (2) it has very few parameters to adjust, which makes it particularly easy to implement, as shown by several cases to which DE has been successfully applied to a diverse set of optimization problems [16–18].

2.2 Gaussian Approximation

Assuming an image has L gray levels $[0, \dots, L - 1]$, following a gray level distribution which can be displayed in the form of the histogram $h(g)$. In order to simplify the description, the histogram is normalized and is considered as a probability distribution function:

$$h(g) = \frac{n_g}{N}, \quad h(g) \geq 0, \quad N = \sum_{g=0}^{L-1} n_g, \quad \text{and} \quad \sum_{g=0}^{L-1} h(g) = 1, \quad (2.1)$$

assuming that n_g denotes the number of pixels with gray level g while N is the total number of pixels in the image. The histogram function can be contained into a mix of Gaussian probability functions, yielding:

$$p(x) = \sum_{i=1}^K P_i \cdot p_i(x) = \sum_{i=1}^K \frac{P_i}{\sqrt{2\pi}\sigma_i} \exp \left[-\frac{(x - \mu_i)^2}{2\sigma_i^2} \right], \quad (2.2)$$

considering that P_i is the a priori probability of class i , $p_i(x)$ is the probability distribution function of gray-level random variable x in class i , μ_i and σ_i are the mean and standard deviation of the i th probability distribution function, and K is the number of classes within the image. In addition, the constraint $\sum_{i=1}^K P_i = 1$ must be satisfied. The typical mean square error consideration is used to estimate the $3K$ parameters P_i , μ_i and σ_i , $i = 1, \dots, K$. For example, the mean square error between the composite Gaussian function $p(x_i)$ and the experimental histogram function $h(x_i)$ is defined as follows:

$$E = \frac{1}{n} \sum_{j=1}^n [p(x_j) - h(x_j)]^2 + \omega \cdot \left| \left(\sum_{i=1}^K P_i \right) - 1 \right| \quad (2.3)$$

Assuming an n -point histogram as in [13] and ω being the penalty associated with the constrain $\sum_{i=1}^K P_i = 1$.

In general, the determination of parameters that minimize the square error is not a simple problem. A straightforward method to decrease the partial derivatives of the error function to zero considers a set of simultaneous transcendental equations [13]. An analytical solution is not available due to the non-linear nature of the equations. The algorithm therefore makes use of a numerical procedure following an iterative approach based on the gradient information. However, considering that the gradient descent method may easily get stuck within local minima, the final solution is highly dependent on the initial values. According to our previous experience, the evolution-based approaches actually provide a satisfactory performance in case of image processing problems [18–20]. The DE algorithm is therefore adopted in order to find the parameters and their corresponding threshold values.

2.3 Differential Evolution Algorithms

In recent years, there has been a growing interest in evolutionary algorithms for diverse fields of Science and Engineering. The Differential Evolution algorithm (DE) [14, 15, 21], is a relatively novel optimization technique to solve numerical-optimization problems. The algorithm has successfully been applied to

several sorts of problems as it has claimed a wider acceptance and popularity, following its simplicity, robustness, and good convergence properties [16–18].

The DE algorithm is population based including a simple and direct searching algorithm for globally optimizing multi-modal functions. Just like the Genetic Algorithms (GA), it employs crossover and mutation operators as selection mechanisms. As previously mentioned, an important difference among other evolutionary computational techniques, such as genetic algorithms (GA), is that the GA relies on the crossover operator which provides the exchange of information required to build better solutions. DE algorithm fully relies on the mutation operation as its central procedure.

The DE algorithm also employs a non-uniform crossover, taking child vector parameters from one parent more frequently than from others. The non-uniform crossover operator efficiently shuffles information about successful combinations. This enables the searching to be focused on the most promising area of the solution space. The DE algorithm introduces a novel mutation operation which is not only simple, but also significantly effective. The mutation operation is based on the differences of randomly sampled pairs of solutions within the population. Besides of being simple and capable of globally optimize a multi-modal search spaces, the DE algorithm shows other benefits: it is fast, easy to use, very easily to adapt in case of integer or discrete optimizations. It is quite effective for nonlinear constraint optimization, including penalty functions.

The version of DE algorithm used in this chapter is known as the DE/best/l/exp or “DE1” (see [14, 15]). Classic DE algorithm begins by initializing a population of N_p and an D -dimensional vectors with parameter values which are randomly and uniformly distributed between the pre-specified lower initial parameter bound $x_{j,\text{low}}$ and the upper initial parameter bound $x_{j,\text{high}}$

$$\begin{aligned} x_{j,i,t} &= x_{j,\text{low}} + \text{rand}(0, 1) \cdot (x_{j,\text{high}} - x_{j,\text{low}}); \\ j &= 1, 2, \dots, D; i = 1, 2, \dots, N_p; t = 0. \end{aligned} \quad (2.4)$$

The subscript t is the generation index, while j and i are the parameter and population indexes respectively. Hence, $x_{j,i,t}$ is the j th parameter of the i th population vector in generation t .

To generate a trial solution, DE algorithm first mutates the best solution vector from the current population by adding to the scaled difference of two other vectors from the current population

$$\begin{aligned} \mathbf{v}_{i,t} &= \mathbf{x}_{\text{best},t} + F \cdot (\mathbf{x}_{r_1,t} - \mathbf{x}_{r_2,t}); \\ r_1, r_2 &\in \{1, 2, \dots, N_p\}, \end{aligned} \quad (2.5)$$

with $\mathbf{v}_{i,t}$ being the mutant vector. Vector indexes r_1 and r_2 are randomly selected considering that both are distinct and different from the population index i (i.e., $r_1 \neq r_2 \neq i$). The mutation scale factor F is a positive real number typically less than 1.

Next step considers one or more parameter values of the mutant vector $\mathbf{v}_{i,t}$ to be exponentially crossed to those belonging to the i th population vector $\mathbf{x}_{i,t}$. The result is the trial vector $\mathbf{u}_{i,t}$

$$u_{j,i,t} = \begin{cases} v_{j,i,t}, & \text{if } \text{rand}(0,1) \leq Cr \text{ or } j = j_{\text{rand}}, \\ x_{j,i,t}, & \text{otherwise.} \end{cases} \quad (2.6)$$

with $j_{\text{rand}} \in \{1, 2, \dots, D\}$.

The crossover constant ($0.0 \leq Cr \leq 1.0$) controls the section of parameters belonging to the mutant vector which contributes to the trial vector. In addition, the trial vector always inherits the mutant vector parameter with a random index j_{rand} to ensure that the trial vector differs by at least one parameter from the vector to be compared ($\mathbf{x}_{i,t}$).

Finally, a selection operation is used to improve the solutions. If the cost function of the trial vector is less or equal to the target vector, then the trial vector replaces the target vector on the next generation. Otherwise, the target vector remains in the population for at least one new generation:

$$\mathbf{x}_{i,t+1} = \begin{cases} \mathbf{u}_{i,t}, & \text{if } f(\mathbf{u}_{i,t}) \leq f(\mathbf{x}_{i,t}), \\ \mathbf{x}_{i,t}, & \text{otherwise.} \end{cases} \quad (2.7)$$

here, f represents the cost function. These steps are repeated until a termination criterion is attained or a predetermined generation number is reached.

2.4 Determination of Thresholding Values

The next step is to determine the optimal threshold values. Considering that the data classes are organized such that $\mu_1 < \mu_2 < \dots < \mu_K$, the threshold values are obtained by computing the overall probability error for two adjacent Gaussian functions, following:

$$E(T_i) = P_{i+1} \cdot E_1(T_i) + P_i \cdot E_2(T_i), \quad i = 1, 2, \dots, K-1 \quad (2.8)$$

considering

$$E_1(T_i) = \int_{-\infty}^{T_i} p_{i+1}(x) dx, \quad (2.9)$$

and

$$E_2(T_i) = \int_{T_i}^{\infty} p_i(x) dx, \quad (2.10)$$

$E_1(T_i)$ is the probability of mistakenly classifying the pixels in the $(i + 1)$ th class to the i th class, while $E_2(T_i)$ is the probability of erroneously classifying the pixels in the i th class to the $(i + 1)$ th class. P'_j s are the a priori probabilities within the combined probability density function, and T_i is the threshold value between the i th and the $(i + 1)$ th classes. One T_i value is chosen such as the error $E(T_i)$ is minimized. By differentiating $E(T_i)$ with respect to T_i and equating the result to zero, it is possible to use the following equation to define the optimum threshold value T_i :

$$AT_i^2 + BT_i + C = 0 \quad (2.11)$$

considering

$$\begin{aligned} A &= \sigma_i^2 - \sigma_{i+1}^2 \\ B &= 2 \cdot (\mu_i \sigma_{i+1}^2 - \mu_{i+1} \sigma_i^2) \\ C &= (\sigma_i \mu_{i+1})^2 - (\sigma_{i+1} \mu_i)^2 + 2 \cdot (\sigma_i \sigma_{i+1})^2 \cdot \ln \left(\frac{\sigma_{i+1} P_i}{\sigma_i P_{i+1}} \right) \end{aligned} \quad (2.12)$$

Although the above quadratic equation has two possible solutions, only one of them is feasible (positive and inside the interval).

2.5 Experimental Results

In this section, two experiments are tested to evaluate the performance of the presented algorithm. The first one considers the well-known image of the “The Cameraman” shown in Fig. 2.1a. The corresponding histogram is presented by Fig. 2.1b. The goal is to segment the image in 3 different pixel classes. According to Eq. 2.2, during learning, the DE algorithm adjusts 9 parameters in this test, following the minimization procedure sketched by Eq. 2.3. In this experiment, a population of 90 (N_p) individuals is considered. Each candidate holds 9 dimensions, yielding:

$$I_N = \{P_1^N, \sigma_1^N, \mu_1^N, P_2^N, \sigma_2^N, \mu_2^N, P_3^N, \sigma_3^N, \mu_3^N\}, \quad (2.13)$$

with N representing the individual’s number. The parameters (P, σ, μ) are randomly initialized, but assuming some restrictions to each parameter (for example μ must fall between 0 and 255). Before optimizing the elements of DE, some parameters have to be defined as follows: $F = 0.25$; $Cr = 0.8$, and $\omega = 1.5$.

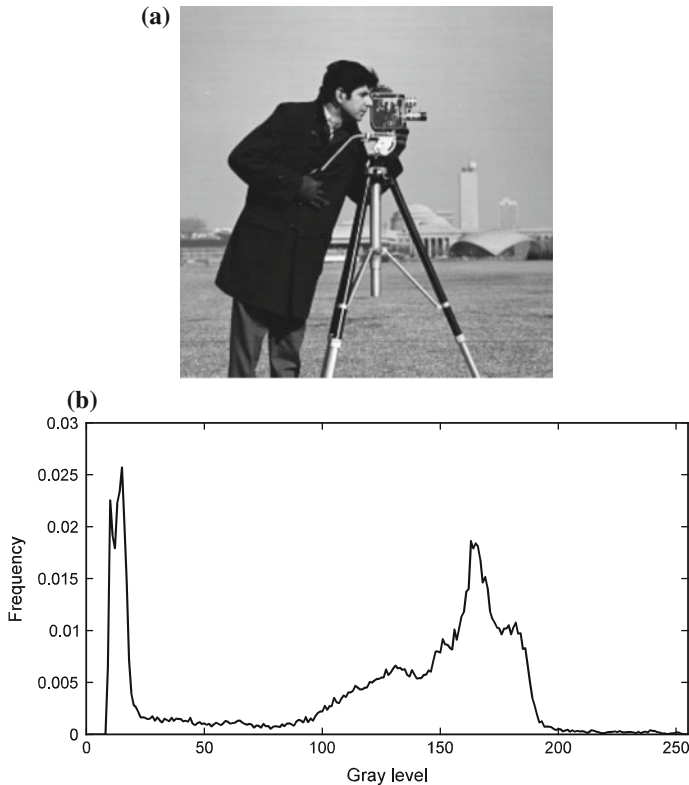


Fig. 2.1 **a** Original “The Cameraman” image, and **b** its correspondent histogram

In theory, the search for optimal values using DE covers all the search space. However, some sets of parameters chosen within the search space might not provide plausible solutions to the problem. Restricting the search space to a feasible region might also be difficult because there are not simple constraints [22]. In order to overcome this inconvenience, a penalty strategy is implemented in the DE algorithm [22, 23]. If a candidate parameter set is not a physically plausible solution, then an exaggerated cost function value is returned. Considering that such value is uncommonly large in comparison to usual cost function values, these “negative” offspring are usually eliminated in the next single generation.

The experiments suggest that after 200 iterations, the DE algorithm has converged to the global minimum. Figure 2.2a shows the obtained Gaussian functions (pixel classes), while Fig. 2.2b shows the combined graph. The layout in Fig. 2.2b suggest an easy combination of the Gaussian functions which approaches to shape of the graph shown in Fig. 2.1b which represents the histogram of the original image. Figure 2.3 shows the segmented image whose threshold values are calculated according to Eqs. 2.11 and 2.12.

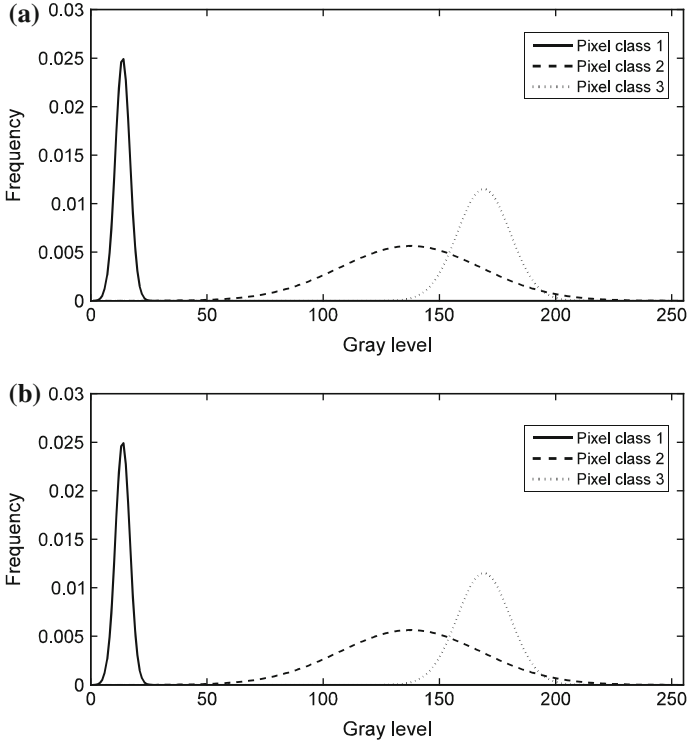


Fig. 2.2 Applying the DE algorithm for 3 classes and its results: **a** Gaussian functions for each class, **b** Mixed Gaussian functions (approach to the original histogram)

An important experiment to test the algorithm's ability to apply the segmentation algorithm with no previous knowledge is now presented. It also considers a higher number of classes by an increase to four instead of three segments. The algorithm follows the minimization presented by Eq. 2.3. Twelve parameters are now considered corresponding to the values of the four Gaussian functions. One population of 120 individuals and 12 dimensions are assumed for this test. Other parameters of the DE algorithm have kept the same values that in the experiment for three classes. Figure 2.4a shows the Gaussian functions after 200 iterations. The combined graph is presented in Fig. 2.4b. It can be seen in Fig. 2.4b the improvement of the new procedure in comparison to the original histogram shown in Fig. 2.1b. Finally, Fig. 2.5 shows the computed threshold values.

The second experiment considers a new image known as “The scene” shown by Fig. 2.6a. The histogram is presented in Fig. 2.6b. Following the first experiment, the image is segmented considering 3 pixel classes. The optimization is performed by the DE algorithm which results in the Gaussian functions shown by Fig. 2.7a. Figure 2.7b presents the combined graph as it results from the addition of the



Fig. 2.3 The image after the segmentation is applied, considering three classes

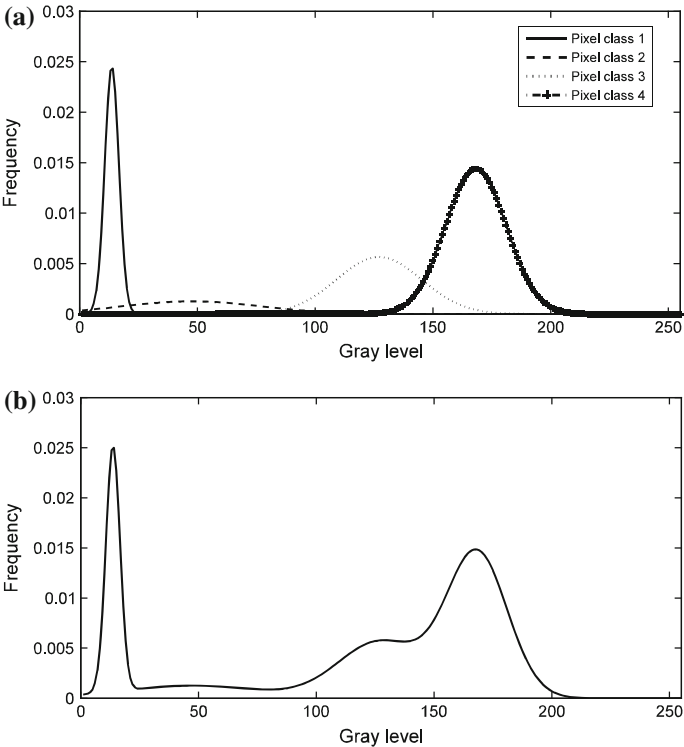


Fig. 2.4 Segmentation of the test image as it was obtained by the DE algorithm considering 4 classes: **a** Gaussian functions for each class. **b** Mixed Gaussian functions approaching the look of the original histogram



Fig. 2.5 The segmented image as it was obtained from considering 4 classes

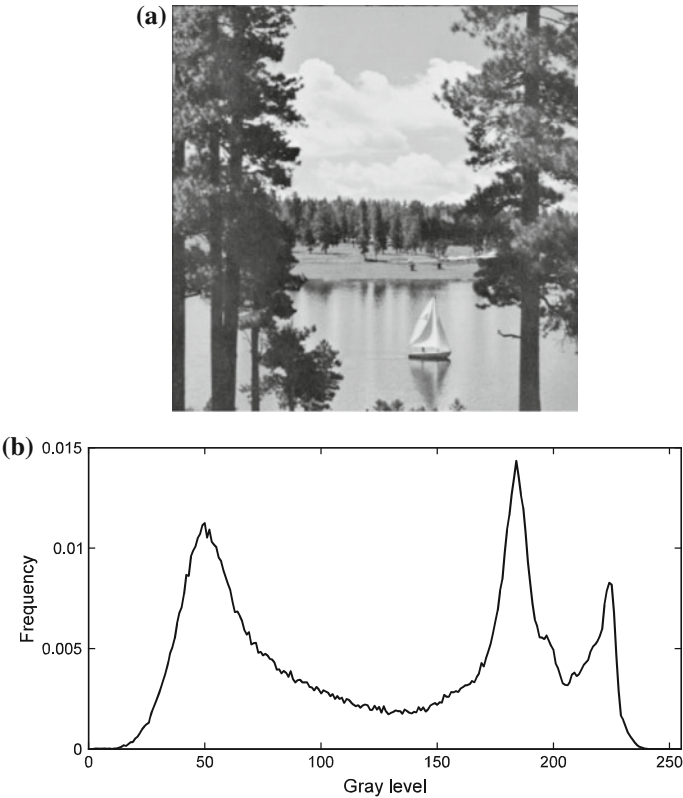


Fig. 2.6 Second experiment data, **a** the original image “The scene”, and **b** its histogram

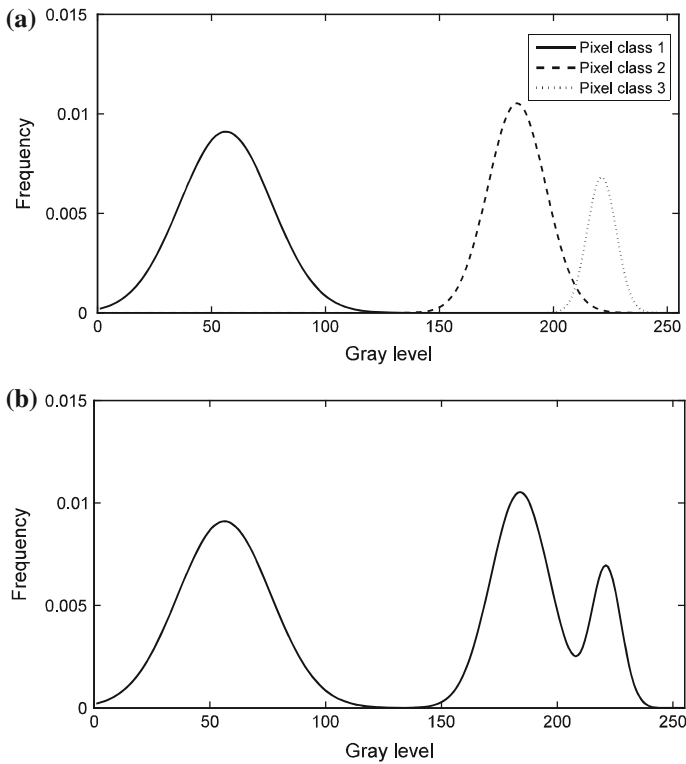


Fig. 2.7 Results obtained by the DE algorithm for 3 classes: **a** Gaussian functions at each class, **b** Combining the Gaussian functions, it approaches the original histogram

Gaussian functions. The experiment considered the parameters used in the DE algorithm for the first experiment.

While Fig. 2.8 shows the segmentation of the image considering 3 classes, the next experiment considers the same image “The scene”, being segmented under 4 classes. After the optimization by the DE algorithm, the combined layout of the 4 Gaussian functions is obtained and shown by Fig. 2.9a. It is also evident that the resulting function approaches the original histogram. Figure 2.9b shows the results from the segmentation algorithm.



Fig. 2.8 “The scene” image considering 3 classes

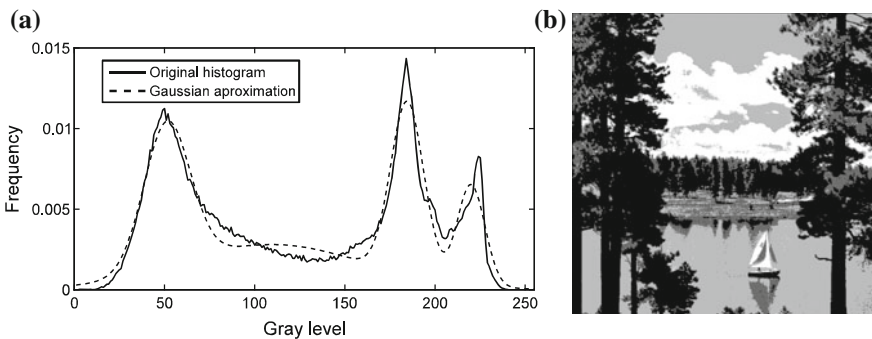


Fig. 2.9 Segmentation of “The Scene” considering the four classes for the DE algorithm: **a** comparison between the original histogram and the Gaussian approach, **b** the segmented image

2.6 Conclusions

In this chapter, an automatic image multi-threshold approach based on differential evolution optimization has been presented. The overall method can also be considered as an optimization algorithm based on differential evolution. The objects and background components within the image are assumed to fit into Gaussian distributions exhibiting non-equal mean and standard deviation. Therefore the histogram can thus be approximated by a mixture of Gaussian functions. The algorithm DE is used to estimate the parameters for the mixture density function as

it seeks to get a minimum square error between the density function and the original histogram. Experimental results show that the presented approach can produce satisfactory results. Further chapter for extending the approach is under development, including a formal comparison to other state-of-the-art image segmentation techniques.

References

1. Abak, T., Baris, U., Sankur, B.: The performance of thresholding algorithms for optical character recognition. In: *Proceedings of International Conference on Document Analytical Recognition 1997*, pp. 697–700 (1997)
2. Kamel, M., Zhao, A.: Extraction of binary character/graphics images from grayscale document images. *Graph. Models Image Process.* **55**(3), 203–217 (1993)
3. Trier, O.D., Jain, A.K.: Goal-directed evaluation of binarization methods. *IEEE Trans. Pattern Anal. Mach. Intell.* **17**(12), 1191–1201 (1995)
4. Bhanu, B.: Automatic target recognition: state of the art survey. *IEEE Trans. Aerosp. Electron. Syst.* **22**, 364–379 (1986)
5. Sezgin, M., Sankur, B.: Comparison of thresholding methods for non-destructive testing applications. In: *IEEE International Conference on Image Processing 2001*, pp. 764–767 (2001)
6. Sezgin, M., Tasaltin, R.: A new dichotomization technique to multilevel thresholding devoted to inspection applications. *Pattern Recogn. Lett.* **21**(2), 151–161 (2000)
7. Guo, R., Pandit, S.M.: Automatic threshold selection based on histogram modes and discriminant criterion. *Mach. Vis. Appl.* **10**, 331–338 (1998)
8. Pal, N.R., Pal, S.K.: A review on image segmentation techniques. *Pattern Recogn.* **26**, 1277–1294 (1993)
9. Shao, P.K., Soltani, S., Wong, A.K.C., Chen, Y.C.: Survey: a survey of thresholding techniques. *Comput. Vis. Graph. Image Process.* **41**, 233–260 (1988)
10. Snyder, W., Bilbro, G., Logenthiran, A., Rajala, S.: Optimal thresholding: a new approach. *Pattern Recogn. Lett.* **11**, 803–810 (1990)
11. Chen, S., Wang, M.: Seeking multi-thresholds directly from support vectors for image segmentation. *Neurocomputing* **67**(4), 335–344 (2005)
12. Chih-Chih, L.: A novel image segmentation approach based on particle swarm optimization. *IEICE Trans. Fundam.* **89**(1), 324–327 (2006)
13. Gonzalez, R.C., Woods, R.E.: *Digital image processing*. Addison Wesley, Reading, MA (1992)
14. Storn, R.M., Price, K.V.: Differential evolution: a simple and efficient heuristic for global optimization over continuous spaces. *J. Global Optim.* **11**, 341–359 (1997)
15. Price, K.V., Storn, R.M., Lampinen, J.: *Differential evolution: a practical approach to global optimization*. Springer, Berlin (2005)
16. Chang, W.D.: Parameter identification of rossler's chaotic system by an evolutionary algorithm. *Chaos, Solutions Fractals* **29**(5), 1047–1053 (2006)
17. Liu, B., Wang, L., Jin, Y.H., Huang, D.X., Tang, F.: Control and synchronization of chaotic systems by differential evolution algorithm. *Chaos, Solutions Fractals* **34**(2), 412–419 (2007)
18. Baştürk, A., Günay, E.: Efficient edge detection in digital images using a cellular neural network optimized by differential evolution algorithm. *Expert Syst. Appl.* **36**(8), 2645–2650 (2009)
19. Lai, C.-C., Tseng, D.-C.: An optimal L-filter for reducing blocking artifacts using genetic algorithms. *Signal Process.* **81**(7), 1525–1535 (2001)

20. Tseng, D.-C., Lai, C.-C.: A genetic algorithm for MRF-based segmentation of multispectral textured images. *Pattern Recogn. Lett.* **20**(14), 1499–1510 (1999)
21. Price, K.V.: Differential evolution: a fast and simple numerical optimizer. In: *IEEE Biennial Conference of the North American Fuzzy Information Processing Society, NAFIPS*, Berkeley, CA, pp. 524–527 (1996)
22. Franco, G., Betti, R., Lus, H.: Identification of structural systems using an evolutionary strategy. *Eng. Mech.* **130**(10), 1125–1139 (2004)
23. Koziel, S., Michalewicz, Z.: Evolutionary algorithms, homomorphous mappings, and constrained parameter optimization. *Evol. Comput.* **7**(1), 19–44 (1999)

Applications of Evolutionary Computation in Image
Processing and Pattern Recognition

Cuevas, E.; Zaldívar, D.; Perez-Cisneros, M.

2016, XV, 274 p. 111 illus., 55 illus. in color., Hardcover

ISBN: 978-3-319-26460-8