Journal of AI and Data Mining
Vol 4, No 1, 2016, 77-83 ***10.5829/idosi.JAIDM.2016.04.01.09***

Noisy images edge detection: Ant colony optimization algorithm

Z. Dorrani[1*] and M. S. Mahmoodi [2]

1. Department of Electrical Engineering, Payame Noor University (PNU), Tehran, Iran.
2. Department of Computer Engineering, Payame Noor University (PNU), Tehran, Iran.

Received 04 January 2015; Accepted 14 October 2015
**Corresponding author: dorrani.z@skpnu.ac.ir(Z. Dorrani).*

Abstract
The edges of an image define the image boundary. When the image is noisy, it does not become easy to identify the edges. Therefore, a method requests to be developed that can identify edges clearly in a noisy image. Many methods have been proposed earlier using filters, transforms and wavelets with Ant colony optimization (ACO) that detect edges. We here used ACO for edge detection of noisy images with Gaussian noise and salt and pepper noise. As the image edge frequencies are close to the noise frequency band, the edge detection using the conventional edge detection methods is challenging. The movement of ants depends on local discrepancy of image's intensity value. The simulation results compared with existing conventional methods and are provided to support the superior performance of ACO algorithm in noisy images edge detection. Canny, Sobel and Prewitt operator have thick, non continuous edges and with less clear image content. But the applied method gives thin and clear edges.

Keywords: *Ant Colony, Edge Detection, Gaussian Noise, Noisy Image, Salt and Pepper Noise.*

1. Introduction

One of the basic issues in image processing and computer vision is identifying the sudden changes of brightness in an image, or edge detection [1-6]. The edges are defined as the boundary between the objects and the background or the boundary between overlapping objects. If the edges are recognized correctly, the location of all objects in the image can be identified exactly and some basic characteristics, such as the surface and the geometry of the objects will be measured easily [7]. There are many methods for extracting and detecting the edges [6]. Most edge detector programs determine the color or the intensity values of the edge pixels. In images without noise, edges are recognized by the gray level changing at a specific pixel [8]. The higher is the gray level changing, the easier is the detection. However, in some images the gray level changes gradually, and no specific pixel can be identified as the edge pixel. Moreover, noise may cause some grey level changes which are not true. In other words, when affected by noise, two pixels with the same grey levels may find different grey levels in the image [9]. Noise is a random phenomena arising from many sources, such as light intensity, type of camera and lens, mobility, temperature, atmospheric effects, and moisture. The random gray level changes of the pixels appear as some discontinuities in the detected edges. Hence, edge detection in noisy images is a nontrivial problem due to the random damage in the image [8].

To reduce noise effects, Gaussians filtering [10], and some algorithms based on Wavelet Transform [11], mathematical morphology [12], neural networks [13], fuzzy networks [14,15] and an enhanced median filter [16] have been proposed.

The ACO algorithm in [1] and [2] and [17] are proposed for edge detection in without noise images, but in this paper, we used the ACO algorithm for edge detection in image with presence noise. So far this algorithm is not used for edge detection of noisy image. However, a majority of edge detector algorithms, proposed so far, are incapable to remove noise effects precisely. Instead, they optimize their performance in terms of noise effects removal [9]. We show that ACO algorithm can detect edges with presence noise in images.

The remainder of the paper is organized as follows. ACO is introduced briefly in section 2, and ACO implementation for the edge detection problem is explained in section 3. The numerical results are demonstrated in section 4 and the paper is concluded in section 5.

2. Ant colony optimization

ACO algorithms are inspired by the natural behavior of ants which find the shortest paths between their nest and food sources using the deposited pheromone on the ground in an iterative process. Similarly, ACO prepares a set of solution components (paths) and updates it based on a pheromone model (weights of the paths) in each iteration. ACO algorithm constructs the solution with a probabilistic mechanism from the components. The pheromone model is a set of values representing a probability distribution over the search space. The pheromone update is performed to increase the probability of constructing good solutions from the components [17]. ACO algorithm definition Using of the variables used in references [18,19].

To implement ACO algorithm, K artificial moving ants in an area consisting of $M_1 \times M_2$ nodes are considered. The ants construct the components by their probabilistic movement based on the transition probability matrix, P^n. The probability of moving from node *i* to node *j* depends on two values associated to the connection of node i to node j: the pheromone value and the heuristic value. The algorithm implementation steps are listed in table 1.

Table 1. ACO algorithm.

step	process
1	Initializing the positions of all K ants and the pheromone matrix $\boldsymbol{\tau}^0$
2	
2-1	Setting the construction-step index n=1:N
2-2	Setting the ant index k=1:K,
3	Moving the kth ant L steps according to an $M_1M_2 \times M_1M_2$ probabilistic transition matrix, P^n
1	Updating the pheromone matrix, $\boldsymbol{\tau}^n$

Accordingly, ACO should perform two main processes iteratively: modifying the probabilistic transition matrix, $p^{(n)}$ and updating pheromone matrix, which are explained in the following. At the n-th step of the algorithm, the k-th ant moves from node i to node j with a probability of:

$$p_{i,j}^{(n)} = \frac{(\tau_{i,j}^{(n-1)})^{\alpha}(\eta_{i.j})^{\beta}}{\sum_{j \in \Omega_i}(\tau_{i,j}^{(n-1)})^{\alpha}(\eta_{i.j})^{\beta}} \quad (1)$$

where, $\tau_{i,j}^{(n-1)}$ is the pheromone value of node i to node j connection; $\eta_{i,j}$ represents the heuristic information of going from node i to node j and is assumed to be the same for each construction-step; Ω_i is the set of neighbor nodes of the ant a_k that is on node i; α and β are positive constants representing the relation between pheromone information and heuristic information. The pheromone matrix needs to be updated twice during the process. First, the pheromone matrix is updated as:

$$\begin{cases} (1-\rho).\tau_{i,j}^{(n-1)} + \rho.\Delta_{i,j}^{(k)} & if\ (i,j)\ belongs\ to \\ & the\ best\ tour \\ \tau_{i,j}^{(n-1)} & otherwise \end{cases} \quad (2)$$

where, ρ is the evaporation rate; $\tau_{i,j}^{(n-1)}$ is the pheromone value of node i to node j connection; $\Delta^{(k)}_{i,j}$ is given by the heuristic matrix. Then, the second update is performed after the movement of all *K* ants within each construction-step as:

$$\tau^{(n)} = (1-\psi).\tau^{(n-1)} + \psi.\tau^{(0)} \quad (3)$$

where, ψ is the pheromone decay coefficient.

3. EDGE detection with ACO

In the edge detection problem, ACO detects the image edges with the ants moving on the image edges and recording the intensity values as estimated pheromone in a matrix which shows the edges information in each situation [1,17].

A number of ants move on a two-dimensional image. Their move with the initial probability on any image pixel is determined with pheromone matrix. This matrix is created according to the definition of edge. In other words, pheromone matrix is created when the difference between two adjacent edges is more than a certain amount. Thus, the ants are led by the changes in the intensity values of edge pixels.

This algorithm performs for the first steps and then in the nth step of construction, one ant being randomly chosen from K total ants and this ant will move over the image for L steps. Ultimately, decision-making Stage is done to determine edges. These steps in order are:

1) Initialization stage

According to the ant system [19], a number of ants are randomly distributed on an image with a size of M1×M2. The primary value of each component of the pheromone matrix, $\tau(0)$, is set to a constant value, τinit.

2) Construction stage

At this stage, a random ant is randomly chosen to perform L movement steps from node (l,m) to a neighboring node (i,j) according to the transition probability [21]

$$p^{(n)}_{(l,m),(i,j)} = \frac{(\tau^{(n-1)}_{i,j})^{\alpha}(\eta_{i,j})^{\beta}}{\sum_{(j,j)\in\Omega_{(i,m)}} (\tau^{(n-1)}_{i,j})^{\alpha}(\eta_{i,j})^{\beta}} \quad (4)$$

where, $\tau_{i,j}^{(n-1)}$ is the pheromone value of node i to node j connection; $\eta_{i,j}$ represents the heuristic information of going from node i to node j and is assumed to be the same for each construction-step; Ω_i is the set of neighbor nodes of the ant a_k that is on node i; α and β are positive constants representing the relation between pheromone information and heuristic information. The heuristic value, η (i,j), should be determined according to the possible position of pixel (i,j) [22], as:

$$\eta_{i,j} = \frac{1}{z} v_c(I_{i,j}) \quad (5)$$

where, $z = \sum_{i=1:M_1} \sum_{j=1:M_1} V_c(I_{i,j})$ is a normalized factor, and $I_{i,j}$ represents the intensity value at position (i,j) in image I; $V_c(I_{i,j})$ is a function of a pixel local group defined by:

$$v_c(I_{i,j}) = f(|I_{i-2,j-1} - I_{i+2,j+1}| + |I_{i-2,j+1} - I_{i+2,j-1}| + |I_{i-1,j-2} - I_{i+1,j+2}| + |I_{i-1,j-1} - I_{i+1,j+1}| + |I_{i-1,j} - I_{i+1,j}| + |I_{i-1,j+1} - I_{i-1,j-1}| + |I_{i-1,j+2} - I_{i-1,j-2}| + |I_{i,j-1} - I_{i,j+1}|). \quad (6)$$

Function f(0) in (6) is determined using the following four functions [23]:

$$f(x) = \lambda x \quad \text{for } x \geq 0 \quad (7)$$

$$f(x) = \lambda x^2 \quad \text{for } x \geq 0 \quad (8)$$

$$f(x) = \sin(\frac{\pi x}{2\lambda}) \quad \text{for } 0 \leq x \leq \lambda \quad (9)$$

$$f(x) = \frac{\pi x \sin(\frac{\pi x}{\lambda})}{\lambda} \quad \text{for } 0 \leq x \leq \lambda \quad (10)$$

where, λ adjusts the shapes of the function (7)-(10). The second issue is to determine the range of the ant's movement. To determine the ant moves range four or eight connection is used.

3) Updating stage

The algorithm performs two updating.

After moving an ant, each component of the pheromone matrix is updated by [20]:

$$\tau^{(n-1)}_{i,j} = \begin{cases} (1-\rho).\tau^{(n-1)}_{i,j} + \rho.\Delta^{(k)}_{i,j}, & \text{If (i,j) is visited by the current } k_{th} \text{ ant} \\ \tau^{(n-1)}_{i,j} & \text{Other wise} \end{cases} \quad (11)$$

where, ρ is the evaporation rate; $\Delta^{(k)}_{i,j}$ is given by the heuristic matrix; $\tau_{i,j}^{(n-1)}$ is the pheromone value of node i to node j connection The second update is performed after moving all ants according to (3).

4) Decision stage

A pixel either belongs or does not belong to the edge. Therefore, the decision process is a binary one. A threshold value, T, with an initial value, $T^{(0)}$ which equals the average amount of pheromone matrix, is chosen. The pixels that in two groups can be classified according to the value less than $T^{(0)}$ or greater than $T^{(0)}$. The initial $T^{(0)}$ is computed as follows:

$$T^{(0)} = \frac{\sum_{i=1:M_2} \tau^{(N)}_{i,j} \sum_{j=1:M_2} \tau^{(N)}_{i,j}}{M_1 M_2} \quad (12)$$

where, the iteration index L=0. Then, Matrix pheromone $\tau^{(N)}$ is divided in two groups using $T^{(L)}$, the first group amounts is less than $T^{(L)}$ and the second group greater than $T^{(L)}$. Mean these two calculate using the following formula that can be [1]:

$$m_L^{(l)} = \frac{\sum_{i=1:M_1} \sum_{j=1:M_2} g^L_{T^{(l)}}(\tau^{(N)}_{i,j})}{\sum_{i=1:M_1} \sum_{j=1:M_2} h^L_{T^{(l)}}(\tau^{(N)}_{i,j})} \quad (13)$$

$$m_u^{(l)} = \frac{\sum_{i=1:M_1} \sum_{j=1:M_2} g^u_{T^{(l)}}(\tau^{(N)}_{i,j})}{\sum_{i=1:M_1} \sum_{j=1:M_2} h^u_{T^{(l)}}(\tau^{(N)}_{i,j})} \quad (14)$$

where,

$$g^L_{T^{(l)}}(x) = \begin{cases} x, & if\ x \leq T^{(l)} \\ o & otherwise. \end{cases} \quad (15)$$

$$g^u_{T^{(l)}}(x) = \begin{cases} x, & if\ x \geq T^{(l)} \\ o & otherwise. \end{cases} \quad (16)$$

$$h^L_{T^{(l)}}(x) = \begin{cases} x, & if\ x \geq T^{(l)} \\ o & otherwise. \end{cases} \quad (17)$$

$$h^u_{T^{(l)}}(x) = \begin{cases} x, & if\ x \geq T^{(l)} \\ o & otherwise. \end{cases} \quad (18)$$

The new threshold value is the average of the two above values. The iteration index is set to l =l +1.

$$T^{(l)} = \frac{m_L^{(l)} + m_u^{(l)}}{2} \quad (19)$$

If $\left|T^{(l)}-T^{(n-1)}\right| > \varepsilon$, where ε is a very small number, the iteration stages stops then a binary decision to determine edge pixel, is made on each pixel location (i,j), to determine whether it is an edge pixel.

$$E_{i.j} = \begin{cases} 1, & if\ \tau_{i,j}^{(N)} \geq T^{(l)} \\ 0 & otherwise \end{cases} \quad (20)$$

Summary of ACO method in figure 1 is shown.

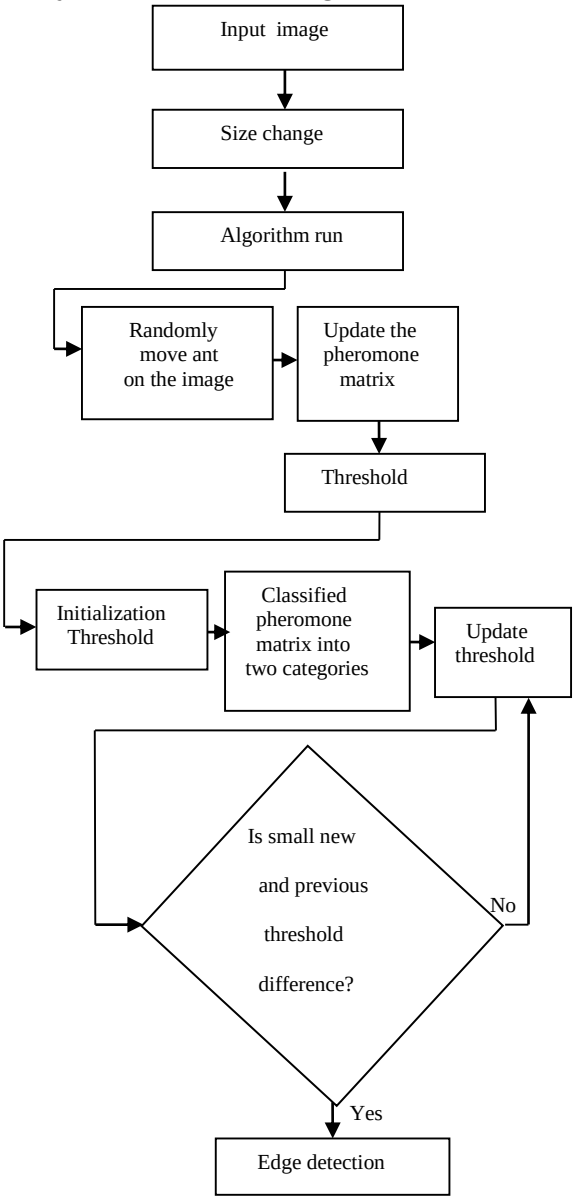


Figure. 1 A review of the performance of ACO algorithm.

4. Simulation results

We evaluate the performance of ACO algorithm in this section. First, a noisy image should be generated. Two types of noise, Gaussian noise, and salt and pepper noise are examined. Algorithm run for image with size $2^n \times 2^n$ and otherwise are resized. The image size is 128 × 128. The initial values of the pheromone matrix components represent the probability of each pixel to be an edge pixel, which is the same for all pixels at the beginning of the iterations. K ants start moving on the edges, hence the probability of being an edge pixel changes. This process is repeated for L steps, and the pixels probabilities change according to the edge intensity values compared to the ones of the neighboring pixels. At the end, the edge pixels are the ones with the highest number of passing ants. The rest of the algorithm parameters are set in such a way that using initial values of reference [20] trial and error method obtained:

k: total number of ants, which here functions [x] shows high right values that are smaller or equal to x.

τ_{init} = 0.0001 initial value of each component of the pheromone matrix.

α = 3: weighting factor of pheromones information in (4).

β = 0.2: weighting factor of the heuristic information in (4).

Ω: the permissible ant's movement range in (4).

λ=1: the adjusting factor of the functions in (7)-(10).

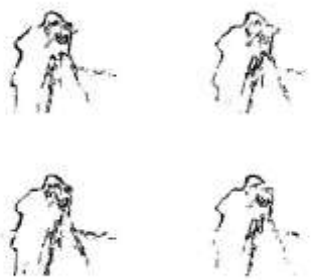
ρ=0.2: the evaporation rate in (11).

L = 40: total number of ant's movement-steps within each construction-step.

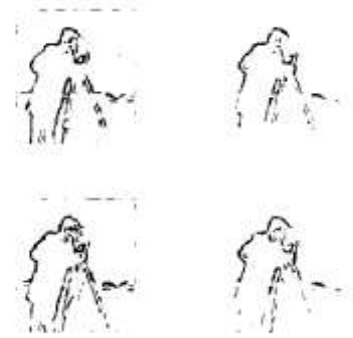
φ = 0.05: the pheromone decay coefficient in (12).

To view the algorithm performance two images are examined.

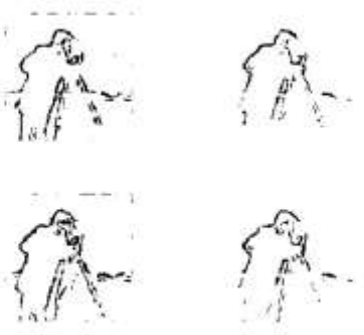
(a)



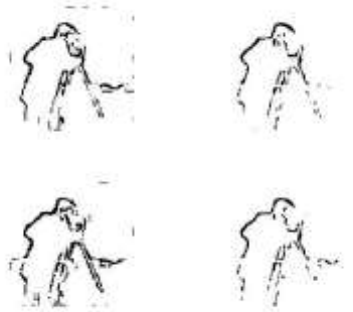
(b)



(c)



(d)

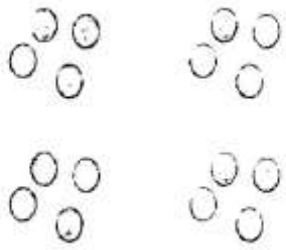


(e)

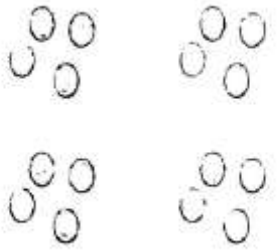
Figure 2. (a) original image (b) edge detection without noise (c) edge detection with salt and pepper noise and intensity= 0.02 (d) edge detection with salt and pepper noise and intensity=0.05 (e) edge detection with Gaussian noise and Intensity=0.05.

Figure 2(a) is the Cameraman image. First, the algorithm is applied to the image without noise. The corresponding results are shown in figure 2(b). Four images obtained according to the four equations in 7-10. Then, figure 2(c) shows the image with salt and pepper noise and intensity=0.02. ACO succeeds in finding the edges. In the next step, noise intensity increases in the image. The computed edges are observed in figure 2(d), and figure 2(g).Next image is Coins. The above steps are related to this image and results are presented in figures 3(a) to (d).

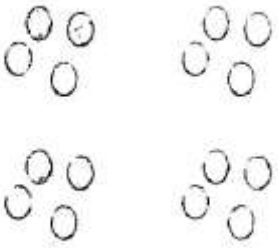
(a)



(b)



(c)



(d)

Figure 3. (a) original image (b) edge detection without noise (c) edge detection with salt and pepper noise and intensity=0.02 (d) edge detection with salt and pepper noise and intensity=0.05.

Figure 3(a) is original image, figure 3(b) is edge detection without noise, figure (c) is edge detection with salt and pepper noise and figure 3(d) is edge detection with Gaussian noise. ACO succeeds in finding the edges. In the next step, edges of same noisy image detection by, Sobel, figure 4(a), Prewitt figure 4(b), and Canny operators figure 4(c). Comparing the eyes of results can see that Canny, Sobel and Prewitt results depended noise effect, but ACO shows better results.

(a)

(b)

(c)

Figure 4. edge detection with salt and pepper noise and intensity= 0.1 (a) edge detection with Sobel operator, (b) edge detection with Prewitt operator, (c) edge detection with Canny operator.

(a) (b)

(c) (d)

Figure 5. The results of edge detection with ACO algorithm in reference [1]. (a-d) the proposed ACO based image edge detection algorithm with the incorporation of the function defined in (7-10).

5. Discussion

The purpose of this paper was to find the edges of the image in noisy images. Many methods were applied for this purpose. However, a majority of edge detector algorithms, proposed so far, are incapable to remove noise effects precisely. This paper finds edges in noisy image using ant colony algorithm. Algorithm finds edges in the presence of white Gaussian noise and salt and pepper noise. After the running algorithm gets four different edges of the four types, experiments were conducted to evaluate the performance of the ACO approach in noisy images.

The results of this algorithm were compared with the results of the algorithm used in reference [1] that the results of this reference shown in figure 5.

The algorithm used in this article is performed only by the presence of white Gaussian noise, while the ACO algorithm can be performed by other noises. After running the algorithm gets four different edges of the four types of relationships, that can use in applications according to type and applied fields. Otherwise, edges can combine using operators such as OR. Thus, the used algorithm can be better and be more complete.

6. Conclusion

ACO algorithm has been used for edge detection of noisy images. Edge detection has been performed in the presence of salt and pepper noise and Gaussian noise which are the dominant noise in most images. While the edges frequencies are

close to the noise frequency bands, ACO succeeds in edge detection. We applied ACO algorithm to different images with different noise intensity of the two types of noise. Comparing to a conventional edge detection algorithm performance, referenced in [1], ACO is more effective.

References

[1] Tian, J., et al. (2008). An ant colony optimization algorithm for image edge detection. IEEE Evolutionary Computation, Hong Kong, China, 2008.

[2] He, Q. & Zhang, Z. (2007). A new edge detection algorithm for image corrupted by White-Gaussian noise. International Journal of Electronics and Communications, vol. 6, no. 8, pp. 546-550.

[3] Khashman, A. (2002). Noise-Dependent optimal Scale in Edge Detection. Industrial Electronics, IEEE International Symposium Proceedings, vol. 2, no. 1, pp. 467-471.

[4] Junxi, S., et al. (2003). A multiscale edge detection algorithm based on wavelet domain vector hidden Markov tree model. Elsevier Journal, vol. 37, no.7, pp. 1315-1324.

[5] Kanga, C. & Wanga, W-J. (2007). A novel edge detection method based on the maximizing objective function. The journal of the pattern recognition, vol. 40, no. 2, pp. 609-618.

[6] Bakhshi, H., et al. (2014). A study on Clustering for Clustering Based Image De-Noising. Journal of Information Systems and Telecommunication, vol. 2, no. 4, pp. 196-204.

[7] Hou, Z. J. & Wei, G. W. (2002). A new approach to edge detection, Elsevier Ltd, vol. 35, no.2, pp. 1559-1570.

[8] Xueqin, S., et al. (2010). Edge Detection for Phytoplankton Cellular Based on Multi-wavelets De-noising. Proceedings of the 2nd International Conference on Computer and Automation Engineering (ICCAE), vol. 2, pp. 190-193, 2010.

[9] He, Jianmin, i., et al. (2009). Canny Edge Detection on a Virtual Hexagonal Image Structure. Conferences on Pervasive Computing (JCPC), Tamsui, Taipei, 2009.

[10] Srivastava, G. K. & Verma, R. (2009). A Novel Wavelet Edge Detection Algorithm for Noisy Images. Ultra Modern International Conference Telecommunications & Workshops, St. Petersburg 2009.

[11] Shih, M. (2005). A wavelet-based multiresolution edge detection and tracking Tseng DC. Image Vision Computr, vol. 23, no.1, pp. 441–451.

[12] Lee, J., et al. (2002). Morphologic edge detection. Journal Robotics and Automation, vol. 3, no. 2, pp. 142–156.

[13] Lesions, R. & Woolfson, M. (2004). Application of region based segmentation and neural network edge detection to skin lesions. Comput Med Imag Graphics, vol. 28, no. 1, pp. 61-80.

[14] Akbari, A. S. & Soraghan, J. J. (2003). Fuzzy-based multiscale edge Detection. Electronics Letters, vol. 39, no. 1, pp. 30-32.

[15] Shafiee, M. & Latif, A. (2014). Modified CLPSO-based fuzzy classification system: Color image segmentation. Journal of AI and Data Mining, vol. 2, no. 2, pp. 167-179.

[16] Arastehfar, S., et al. (2013). An enhanced median filter for removing noise from MR images. Journal of AI and Data Mining, vol. 1, no. 1, pp. 13-17.

[17] Prakash Verma, O. & Madasu Hanmandll, M. (2009). A Novel Approach for Edge Detection using Ant Colon Optimization and Fuzzy Derivative Technique. Advance Computing Conference, Patiala, 2009.

[18] Gill J., et al. (2014). Wavelet with Filter Based Method for Edge Detection Using Ant Colony Optimization. International Journal of Advanced Research in Computer Science and Software Engineering, vol. 4, no. 10, pp. 281-285.

[19] Puneet, R. & Maitreyee, D. (2013). Image Edge Detection using Modified Ant Colony Optimization Algorithm based on Weighted Heuristics. International Journal of Computer Applications, vol. 68, no. 15, pp. 5-9.

[20] Rahebi, J., et al. (2011). Biomedical Image Edge Detection using an Ant Colony Optimization Based on Artificial Neural Networks. International Journal of Engineering Science and Technology (IJEST), vol. 3 no. 2, pp. 8211-8218.

[21] Xiaochen, L. & Suping. F. (2015). A convenient and robust edge detection method based on ant colony optimization. Optics Communications, vol. 353, no. 1, pp. 147–157.

[22] Samit, A. & Dipak Kumar, G., (2014). Edge detection using ACO and F ratio. Springer, vol. 8, no. 1, pp. 625–634.

[23] Charu, G. & Sunanda, G., (2013). Edge Detection of an Image based on Ant Colony Optimization Technique, vol. 2 no. 6, pp. 114-120.

نشریه هوش مصنوعی و داده‌کاوی

دانشگاه شاهرود

لبه‌یابی تصاویر نویزی با استفاده از الگوریتم بهینه‌سازی اجتماع مورچگان

زهره درانی[1]* و مریم سادات محمودی[2]

[1] گروه مهندسی برق، دانشگاه پیام نور، تهران، ایران.

[2] گروه مهندسی کامپیوتر، دانشگاه پیام نور، تهران، ایران.

ارسال ۲۰۱۵/۰۱/۰۴؛ پذیرش ۲۰۱۵/۱۰/۱۴

چکیده:

لبه‌ها در یک تصویر، مرز آن تصویر تعریف می‌شوند. زمانی‌که تصویر نویزی است، به راحتی لبه‌ها قابل تشخیص نیست. بنابراین، روش پیشرفته‌ای مورد نیاز است تا در یک تصویر نویزی بتواند لبه‌ها را به درستی تشخیص دهد. پیش از این روش‌های بسیاری با استفاده از فیلترها، انتقال و موجک با الگوریتم بهینه‌سازی اجتماع مورچگان مطرح شده است. ما در اینجا، الگوریتم بهینه‌سازی اجتماع مورچگان را برای آشکارسازی لبه در تصاویر نویزی با نویز گوسی و نمک و فلفل استفاده کردیم. از آنجایی که فرکانس‌های لبه خیلی نزدیک به باند فرکانسی نویز است، لبه‌یابی با استفاده از روش‌های لبه-یابی معمول مشکل است. حرکت مورچه‌ها بستگی به اختلاف محلی مقدار شدت تصویر دارد. نتایج شبیه‌سازی در مقایسه با روش‌های معمولی موجود و ارائه شده، عملکرد بهتر الگوریتم بهینه‌سازی اجتماع مورچگان را در تشخیص لبه تصاویر نویزی نشان می‌دهد. عملگرهای کنی، سوبل و پرویت، لبه‌های ضخیم، ناپیوسته و محتوای تصویر بدون وضوح را ارائه می‌دهند. اما روش بکار برده شده، لبه‌های نازک و روشن می‌دهد.

کلمات کلیدی: اجتماع مورچگان، لبه‌یابی، نویز گوسی، تصاویر نویزی، نویز نمک و فلفل.