Applied Soft Computing 11 (2011) 4883–4893

Contents lists available at ScienceDirect

Applied Soft Computing

journal homepage: www.elsevier.com/locate/asoc

An ant-inspired algorithm for detection of image edge features

S. Ali Etemad[a,*], Tony White[b]

[a] Department of Systems and Computer Engineering, Carleton University, Ottawa, Ontario K1S5B6, Canada
[b] Department of Computer Science, Carleton University, Ottawa, Ontario K1S5B6, Canada

ARTICLE INFO

Article history:
Received 21 November 2010
Received in revised form 14 May 2011
Accepted 14 June 2011
Available online 24 June 2011

Keywords:
Ant colony systems
Feature extraction
Image edge analysis

ABSTRACT

This paper presents a technique inspired by swarm methodologies such as ant colony algorithms for processing simple and complicated images. It is shown that the proposed technique for image processing is capable of performing feature extraction for edge detection and segmentation, even in the presence of noise. Our proposed approach, Ant-based Correlation for Edge Detection (ACED), is tested on different samples and the results are compared to typical established non-swarm-based methods. The comparative analysis highlights the advantages of the proposed method which generates less distortion when noise is added to the test images. Both qualitative and quantitative evaluations support the claim, confirming the significance of our swarm-based method for image feature extraction and segmentation.

1. Introduction

Image processing and machine vision have for long been a vital element in various fields of technology. While automated and artificially intelligent systems in many cases require a significant amount of precise and robust image processing capabilities, robust techniques are yet to be proposed and demonstrated for many ongoing problems. Different techniques ranging from simple filtering procedures to more complicated methods like SIFT [1] aid in solving many practical problems for industrial and research applications. In this regard, more dynamic algorithms, despite their promising performance in many other fields, have not been widely explored. This could be in some part due to the existing gap between researchers in the two fields of machine vision and soft computing.

Swarm intelligence is a branch of artificial intelligence which provides dynamic, adaptive, and flexible solutions using self-organized decentralized algorithms [2]. While swarm intelligence maintains stochastic properties for most of its procedures, it manages to provide robust solutions in many domains [3,4,5]. The applications of swarm intelligence for practical research and even industrial problems have grown substantially in recent years.

In this paper, we approach the problem of image feature extraction using swarm-based image processing. We propose a method capable of extracting image features for edge detection. The method is employed and tested on different images, as well as the benchmark test on Lena. Furthermore, sensitivity of the algorithm to the different parameters used in the method is explored. The effect of noise on the proposed method is also tested, and finally we compare our results to other well-established methods from the research literature.

The two main contributions of this work are: first, demonstrating that a stochastic, distributed, agent-based, and swarm-based method can perform an image processing task such as feature extraction with high accuracy, outperforming traditional methods, even in the presence of noise, and second, developing a new algorithm for image edge extraction which can be tuned using different parameters for satisfying performance in the presence of noise.

In Section 2 of this paper we review some of the related work in this area. In Section 3 an overview of the problem central to this paper is presented. A brief definition of the problem is first described in Section 3.1, and an overview of ant-based swarm intelligence is reviewed in Section 3.2. Subsequently, the proposed model is described in Section 4 and details of the methodology are described throughout this section. Section 5 explains the post-processing procedures required subsequent to execution of the main algorithm, as well as the methods employed for evaluation of the proposed feature extraction system. In Section 6 the experimental setup and respective results for validation of the proposed method are described and presented, and accordingly, respective discussions are provided. Finally, in Section 7 conclusions and final remarks are presented.

2. Related work

In the field of image processing, the use of soft computing, and ant-based swarm intelligence in particular, do not have a long

* Corresponding author.
E-mail addresses: etemad@sce.carleton.ca (S.A. Etemad), arpwhite@scs.carleton.ca (T. White).

1568-4946/$ – see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.asoc.2011.06.011

history. Liu et al. [6] have proposed an agent-based technique for extraction of edges in grayscale images. Their proposed method revolves around autonomous agents which perform robustly for some tasks such as edge detection.

In [7], Benatcha et al. have employed ants for image segmentation. The algorithm is based on the social behavior of ants when cleaning their nest. This method seems to perform with reasonable accuracy for the one example provided in the paper. However, further evaluation of the procedure, especially for more complex scenarios, seems necessary prior to concrete judgment.

Zhuang et al. have proposed another ant-based image feature extraction algorithm [8] in which they utilize the Ant Colony System (ACS) for edge extraction. The proposed method in this paper, which seems to be built on an earlier work by Zhuang [9], is capable of segmenting simple and semi-complex images.

Similar to [8,9], in [10,11] Ouadfel and Batouche have employed ACS for image segmentation. The authors utilize MRF (Markov Random Fields) alongside ACS for detection of edges. The proposed algorithm is tested on MR images (Magnetic Resonance images, in this case, brain phantoms) and the results are compared with two other methods based upon genetic algorithms and simulated annealing. Segmentation using all three techniques maintained a similarity measure (Jaccard measure) in the range of 80–90%.

Lakehal in [12] has employed ant-based swarm intelligence for detection of centers of objects in an image. This method must be employed subsequent to an agent-based edge detection method such as [13], and therefore cannot act alone in terms of feature extraction.

Mirzayans et al. [14] have utilized a swarm-based method for recognizing objects; namely, square, rectangle, cross, triangle, and circle. The method is demonstrated as being fast and efficient, even when different levels of noise have been introduced to the images. The applicability of the technique to real world image object recognition, however, requires further investigation.

Keshtkar and Gueaieb have used a cellular automata based method not entirely dissimilar from the ant-based techniques described earlier, this time for segmentation of dental radiographs [15].

In [16], Du et al. have developed an algorithm similar to basic ACO (Ant Colony Optimization) applied to the TSP (Travelling Salesman Problem). Comparison is with K-means segmentation and results have been only evaluated qualitatively and a full specification of the experimental setup is not provided. The approach of the paper is to create a perceptual graph that represents the relationships between adjacent points. The ants perform a number of moves before a path assessment is performed based upon the gray-scale variance for the neighborhood of the path points.

Wan and Wang [17] have described an algorithm, strongly influenced by basic ACS, which is meant to separate background and foreground components of an image. The proposed algorithm was compared to the MRF (Markov Random Fields) algorithm and demonstrated to be superior.

In [18], Wang and Zhu have used an Alife technique for image feature extraction by use of an algorithm in which agents have a lifetime and reproduce. The central idea of this work is that large numbers of agents live close to image features and are likely to generate large numbers of offspring agents in feature-rich regions of the image. Similar to our work, a two-class approach is proposed: one class of agents remains at pixels considered to be part of a feature and the second class is mobile within the image environment. The results are qualitatively evaluated but the effect of noise on the proposed algorithm is not studied.

While [19,20] use ant- or swarm-like techniques applied in the image recognition domain, the problems that they solve are quite different from the edge detection problem of interest in this paper. Specifically, [19] has as its focus the capture of the field gradients in the associated images and is not identifying features per se, while [20] assumes a number of features and uses an ACO algorithm to decide whether the features are present. Furthermore, [19] provides no qualitative comparisons with traditional feature detection techniques.

Particle Swarm Optimization (PSO), which is another population-based technique belonging to the class of swarm intelligence, has also been employed for image processing. In [21], grayscale image segmentation is carried out and tested using PSO in the presence of salt and pepper noise. PSO is utilized for classification of humans in infrared images [22]. Finally, scene matching, which tends to search for, and map, similar objects in two images is carried out using PSO in [23].

While not a swarm-based technique, Genetic Algorithms (GA) have also been used for edge detection. A GA-based technique has been employed by Gudmundsson et al. on MRI images, CT scans, and ultrasound images [24]. They have compared the performance to Simulated Annealing (SA) where the outcome was improved by 18%.

Based on the provided review, it is notable that the utilization of swarm methods for image processing is a rather new topic of research which has recently been receiving increasing attention. While several simple algorithms, mostly for edge detection and segmentation, have been implemented based on swarm algorithms, a strong algorithm for extraction of features using ants, and more importantly, a comprehensive evaluation of the method and its comparison with well established non-swarm-based methods especially in the presence of noise, is missing. It is this problem that motivates the algorithms proposed in this paper.

3. Background

3.1. Problem overview

In the field of image processing, detecting and locating specific features in images is of great importance. These features can vary from the edges of distinct sections of the image, to more complicated features such as those used for image matching, or even specialized features defined based on the application at hand (i.e., detection of eyes in human face images).

Detection and extraction of different features might require different approaches. While many different techniques for dealing with the diverse problems have been proposed regarding feature extraction throughout the history of image processing, no specific *one for all* solution exists. For edge detection and segmentation, different methods such as Canny [25], Sobel [26], Thresholding [27], Prewitt [28], and other techniques are widely used based on the application. Feature extraction, subsequent to some form of transform being applied to the image, requires a different approach to the problem. Since the transforms can widely vary from affine transforms ($x \rightarrow Ax+b$) to addition of noise and even more complicated transformations, advanced approaches are required. In this area, practical methods often fail to perform the required task and more sophisticated procedures are required.

3.2. Ant colony systems

Swarm intelligence is, as Bonabeau et al. describe it, the process of solving problems using models of collective behaviors of social creatures such as ants, bees, or even bird flocks or schools of fish [2]. Each member of the swarm society acts autonomously and in a self-organizing manner, consequently resulting in adaptive solutions. Such characteristics are closely related to cellular automata which makes it suitable for distributed problem solving approaches.

ACS is a technique within swarm intelligence that employs the characteristics of ants. ACS was proposed by Dorigo et al. [29] and used for finding the optimal paths in a graph. The technique was first applied to solving problems such as the TSP. ACS is based on the property of ants which leave pheromone while searching for food, and other ants follow the pheromone trails. Each ant on a given path can move from node i to j with the probability of $p_{i,j}$ given by Eq. (1).

$$p_{i,j} = \frac{(\tau_{i,j}^{\alpha})(\eta_{i,j}^{\beta})}{\sum_k (\tau_{i,k}^{\alpha})(\eta_{i,k}^{\beta})} \tag{1}$$

In Eq. (1), $\tau_{i,j}$ is the amount of pheromone between nodes i and j, while $\eta_{i,j}$ is the desirability of choosing the path. The parameter α is the influence coefficient for τ, and β is the influence coefficient for η. At each time step the pheromone value is updated based on Eq. (2).

$$\tau_{i,j} = (1-\rho)\tau_{i,j} + \lambda \tag{2}$$

Evaporation of pheromone is modeled by the first term of Eq. (2) where ρ is the pheromone evaporation rate. The second term in Eq. (2), λ, denotes the laying of fresh pheromone by ants upon crossing the path from i to j. The value of λ is dependent on how successful an ant is at solving the problem in consideration.

This general model of path reinforcement has been shown to be able to solve NP hard problems from several domains, including scheduling, routing, and bioinformatics [2].

4. Methodology

The goal of this paper is to design and implement an ant-inspired algorithm such that the ants can perform the image processing task of extracting specific features for edge detection. In this section we will first define the feature space and employed neighborhood, procedures of pheromone accumulation ($\eta_{i,j}$) and evaporation (ρ), and the algorithm used for extraction of features from the image. The algorithm involves several parameters, the description and setting of which is provided in this section.

The proposed model is based on the fact that an image is composed of a number of pixels, creating a map of cells. While RGB images are composed of three channels, a grayscale image is basically a 2D map which is a suitable environment for foraging of ants. Therefore, as the first step of the algorithm, the image is converted to grayscale.

A neighborhood is defined for each pixel which identifies where the ants are permitted to move next. An important fact to take into account is that an ant may end up in a repetitive situation, circling among few pixels. Selecting the Von Neumann neighborhood significantly decreases the chances of such a case. A Von Neumann neighborhood is commonly used in cellular automata models. The neighborhood is illustrated in Fig. 1.

Although using the Von Neummann neighborhood would prevent an agent at (i,j) from moving to cell (i + 1, j + 1) directly, this transition is still possible using two iterations by making one horizontal and one vertical transition. This means that the mobility of the ants is not decreased and an agent starting from any given pixel can reach any other pixel through a sufficient number of transitions.

Pheromone is a decisive component in ant colony algorithms. We define two types of pheromone for the problem studied in this paper. Each ant lays a trail of pheromone type-I as it forages through the 2D map. The value of λ (laid pheromone) is typically set to 1 and ρ (evaporation rate) is adjusted accordingly (ρ < 1). At the end of each iteration of the process, each ant lays pheromone with the intensity of λ and moves on to one of the neighboring cells. Also at the end of each iteration, all the pheromone type-I in the world is attenuated by a factor of $1-\rho$ (Eq. (2)). Since there is no preference for any specific transition over others, α is set to 1 and β is set to 0. We call the global pheromone matrix Λ.

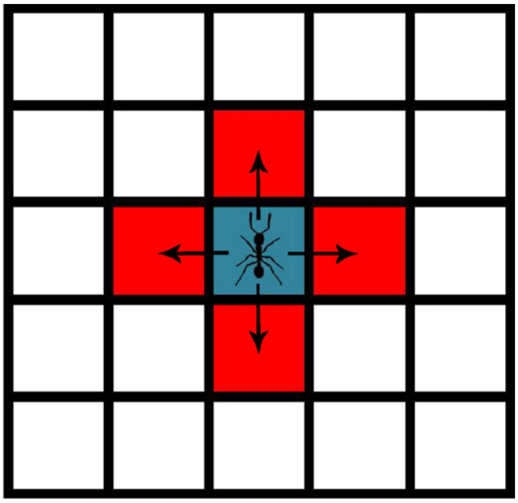

Fig. 1. The neighborhood used by the model. Each ant is only permitted to choose its next cell from this neighborhood (red cells), and therefore a direct transition from (i,j) to (i + 1,j + 1) is prohibited. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of the article.)

Each ant is assigned a short term memory called μ, which it uses to remember its last place in the world that it visited. The memory is required so that the following constraint can be applied:

Constraint: An ant is not permitted to directly return to its previous cell. A return transition, however, is possible through a combination of transitions.

This constraint will ensure that agents do not get stuck in a forward-backward loop and aids in maximizing the mobility of the ants. In order to accomplish this, the previous location of an agent is always stored in μ, and when a successful transition is carried out, the old value in μ is replaced with the latest previous location.

Applying the aforementioned constraint significantly increases the mobility of the agents. However, if each ant is influenced by any type of pheromone in the environment, it can still be easily entangled in a loop of pheromone laid by it or other agents. To avoid this situation, we define pheromone type-II. This type of pheromone is the component responsible for the decision making process of the ants. The stimulus is calculated using pheromone type-II and the two types of pheromone are independent and do not combine. To avoid confusion, when the phrase "pheromone" is used, we are referring to type-I pheromone, and the phrase "stimulus" is used for pheromone type-II.

Fig. 2 illustrates 3 ants foraging through a blank 100 by 100 image for 50 iterations and starting from random pixels. For this trial λ = 1 and ρ = 0. In this image, the color bar refers to pheromone density after normalization.

Besides the pheromone which the ants lay while foraging, the image must contain pheromone-like characteristics to guide the ants independent of λ. To accomplish this, the gradient of the image IM is calculated using Eq. (3).

$$\nabla IM = \frac{\partial IM}{\partial x}\hat{i} + \frac{\partial IM}{\partial y}\hat{j} \tag{3}$$

The global stimulus S, is calculated by Eq. (4) which is the Euclidean norm of the image gradient.

$$S = \left\| \nabla IM \right\| \tag{4}$$

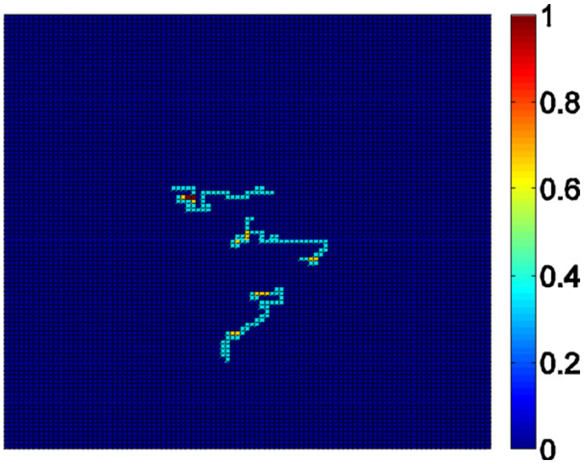

Fig. 2. Normalized pheromone (type-I) accumulation of 3 ants foraging in an empty space (no initial pheromone) for 50 iterations with $\lambda = 1$ and $\rho = 0$. Initiation has taken place at random pixels. The color bar illustrates the pheromone density.

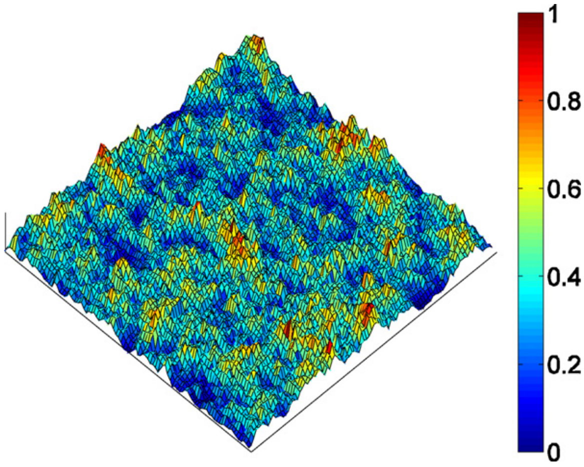

Fig. 4. 1000 ants foraging for 500 iterations in a 100 × 100 blank image and with evaporation rate of $\rho = 0.1$. The two types of pheromone are not combinable.

In this equation, the norm is calculated by:

$$\left\|\nabla IM\right\| = \sqrt{\left(\frac{\partial IM}{\partial x}\right)^2 + \left(\frac{\partial IM}{\partial y}\right)^2} \qquad (5)$$

The stimulus calculated by Eq. (4) is utilized for deciding from among the possible transitions for each agent during each iteration. Eq. (1) and the constraint are used in making this decision.

Unlike many cellular automata worlds where a wrapped environment is used, the space at hand here is an image, and therefore the concept of a wrapped environment would be illogical. This implies that for cells on the edge of the entire image, when calculating the stimulus, fewer neighbors will be involved and the algorithm must be modified accordingly.

By defining two separate types of pheromone, we are effectively characterizing a static and a dynamic field. The static field is determined by the image itself, and the dynamic field is defined and modified through the pheromone left by the ants and evaporated over time. If the two types of pheromone were allowed to mix, if a number of ants (which possess a degree of stochastic behavior) accidentally started following the pheromone deposited by one another, they would be trapped in small clusters of pheromone which eventually they could not escape from. Fig. 3 shows this effect where 1000 ants are allowed to apply Eqs. (1) and (2) to the pheromone type-I left by the agents. In this figure even though a blank image is used, the pheromone is not distributed uniformly. This image shows how most of the pheromone is accumulated on few pixels. In this trial, $\rho = 0.1$ and $\lambda = 1$ (for simplicity, λ is always assumed to be 1 unless stated otherwise).

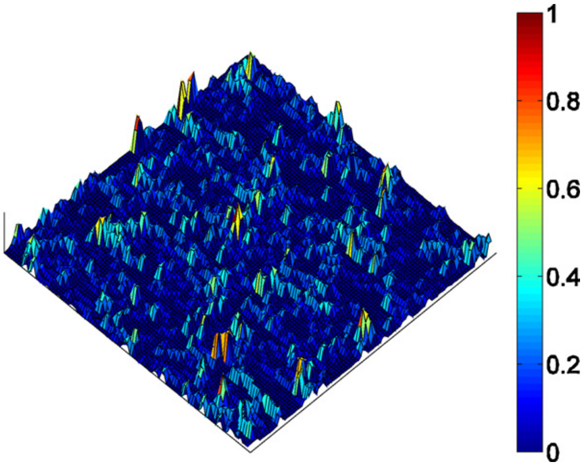

Fig. 3. 1000 ants foraging for 500 iterations in a 100 × 100 blank image and with evaporation rate of $\rho = 0.1$. The two types of pheromone are allowed to combine.

Fig. 4 illustrates a similar situation except for the fact that the proposed two-pheromone approach is used. This image indicates that the difference between high and low peaks is not as extreme as Fig. 3 where much of the pheromone was deposited on a limited number of pixels. Inspection shows that both Figs. 3 and 4 have the same average value of pheromone/pixel (or equal total sum of pheromone). For a scenario like Fig. 3 in which the two types of pheromone are allowed to combine, however, the standard deviation is more than 4 times greater than a scenario like Fig. 4 where the two fields are strictly distinct. Since there are no features in the image, it is desirable that pheromone which is intended to be deposited with greater intensity on feature pixels, be distributed evenly and with a smaller deviation.

The result of Eq. (3) can be positive or negative for each term based on the values of different pixels. If the ants always follow pheromone type-II towards higher density areas, the pixels with the highest densities of gradient within the image would become the goal pixels which the ants would try to reach. This is important to avoid and is taken into account by utilizing the Euclidean norm of the gradient (Eq. (5)) which always returns a positive value.

Once the global stimulus S is derived, for each ant at (i,j) the local stimulus $S_{i,j}$ is calculated. To do this, the function σ is defined which returns the stimulus of all the valid neighboring cells of the ant; that is, taking into account the constraint.

$$S_{i,j} = \sigma(i, j, i', j') = S((neighbors(i, j)) \wedge \neg(i', j')) \qquad (6)$$

In Eq. (6), *neighbors*(i,j) are all the neighbors of cell (i,j) and (i',j') is the previous location of the ant stored in ant memory, μ. The returned stimulus values are utilized using Eq. (1) in making a movement decision.

The overall procedure used for extraction of features is presented by Algorithm 1 in pseudo code.

The Λ matrix returned by Algorithm 1 after the desired number of iterations holds the final distribution of pheromone laid by the agents throughout the 2D map. T is the threshold value described later in Section 5. It is once again emphasized on the fact that the ants, for the entire duration of foraging, are only influenced by pheromone type-II. The criterion for stopping the algorithm is solely based on the number of iterations (max_iterations) which is defined by the user. This number can vary based on the number of features in the image (complexity of the scene). Further details regarding the impact of different iteration values is presented later in Section 6.1.

Algorithm 1. Feature Extraction Process

```
Convert image to grayscale, save as $IM$
Tune $\rho$, $\lambda$, and $T$ based on Section 5
Determine number and initial position of each ant based on Section 6
Select number of iterations (max_iterations)
Calculate $\nabla IM$ using Eq. (3)
Calculate the global stimulus matrix using Eq. (5)
for iteration < max_iterations do
    for all ants do
        Calculate local stimulus using Eq. (6)
        Move ant using Eq. (1)
      Lay pheromone type-I
    end for
    Update $\Lambda$ using Eq. (2)
    iteration = iteration + 1
end for
```

5. Post-processing and evaluation

In this section we first describe the post-processing procedures which need to be carried out for the pheromone-filled image to be interpreted precisely in terms of edge features. It should be noted that these routines are not compulsory, but carried out merely to modify the features so that they can be compared to traditional edge detection techniques which we use to assess the performance of our proposed method. After the post-processing subsection we describe the methods used to evaluate the ACED technique. It should be noted that with the overwhelming processing time associated with the ant-based algorithm, post-processing time is negligible in comparison.

5.1. Post-processing

Most traditional methods of image feature extraction work on grayscale images and return binary features, i.e., a pixel is either a feature point or it is not. Nevertheless, our method results in a fuzzy feature set. After normalization of the pheromone values, the amount of pheromone on each pixel varies between 0 and 1, representing the possibility of that specific pixel being a feature. This property will result in more flexible outputs and results in more flexibility in accepting or rejecting possible features. To be able to evaluate the features with regard to other edge detection methods, a threshold value, *T*, is introduced into the procedure.

The threshold value determines the minimum amount of normalized pheromone required for a pixel to be accepted as an extracted feature. By determining a threshold value close to 1 the level of confidence is increased, yet fewer features will be detected. A value closer to 0 will result in extraction of more features; however, the features will be less reliable and incorrect features might be included in the final set of features. Various methods for automatically selecting suitable threshold values in image processing applications have been proposed [30]. In this research, however, finding the optimal threshold is not a determining factor in illustrating the capabilities of the proposed feature detection algorithm and so *T* is selected manually. Particular automatic methods can also be employed which can result in similar performance.

Further investigation shows that the borders and edges detected by use of this method tend to be slightly thicker than most convolution-based methods of edge detection. Based on different applications, this property can represent a drawback or an asset, yet, to evaluate ACED with respect to the other techniques, multiple layers of the same edge must be reduced to one layer only. To solve this problem, a *thinning* algorithm is carried out as post-processing of the results [31]. Guo and Hall's method [32] is a fast and accurate means of thinning and therefore selected for this purpose. The procedure for this technique is based on dividing the image into two different subfields. The division of the image into the two subfields takes place through a checkerboard-like pattern. For a feature pixel to be deleted, the following two conditions are necessary but not sufficient:

Condition 1. *where $x_1, x_2, \ldots, x_8$ are all the neighboring pixels numbered in counter-clockwise format (starting from the right pixel) and b_i is defined by*

$$b_i = \begin{cases} 1 & \text{if } x_{2i-1} = 0 \text{ and } [x_{2i} = 0 \text{ or } x_{2i+1} = 0] \\ 0 & \text{otherwise} \end{cases}$$

The condition holds true if:

$$\sum_{i=1}^{4} b_i = 1$$

Condition 2. *True if:*

$$2 \le \min \left\{ \sum_{k=1}^{4} x_{2k-1} \vee x_{2k}, \sum_{k=1}^{4} x_{2k} \vee x_{2k+1} \right\} \le 3$$

Conditions 3 and 4 are also necessary for a feature pixel to be removed, the former for the first sub-iteration and latter for the second.

Condition 3. *True if:* $(x_2 \vee x_3 \vee \bar{x}_8) \wedge x_1 = 0$

Condition 4. *True if:* $(x_6 \vee x_7 \vee \bar{x}_4) \wedge x_5 = 0$

For each pixel, the two sub-iterations are carried out and the complete process is repeated on the whole image until the image does not show further modifications.

5.2. Evaluation

Evaluating the results of image feature extraction techniques has long been a topic of research itself. Several different methods have been proposed and employed for assessing the quality and precision of features obtained using various edge detection methods [33]. As the first step towards such an assessment, the *ground truth* must be selected or generated. The ground truth can be manually defined or obtained by means of another method. Once this is done, the set of features in question is compared to the set defined as the ground truth. This comparison can be done through a variety of different measures. In this research we use True Positive/False Positive (TP/FP) based metrics [34,35] and Peak Signal to Noise Ratio (PSNR) [35], which have shown to be effective and informative for evaluating edge detectors, as well as typical subjective evaluation for determining the effectiveness of our proposed method.

Four traditional edge detection methods have been selected to test the proposed method against. Specifically, they are: Canny, Sobel, Prewitt, and Log. The thresholds for these techniques have been selected using the popular measure of mean of the norm of the gradients [36]. When dealing with basic images with no noise, the results of ACED will be evaluated based on all the other four techniques. By doing so, each of the four techniques are considered as ground truth and the ACED method is evaluated accordingly. TP and TN are calculated for each case to determine whether the proposed method shows a performance similar to the traditional methods or not, and if so, which technique does it show to be the most similar to. In the second stage, when salt and pepper noise is added to the images, each of the five methods (4 traditional methods and ACED) before the addition of noise are assigned as the ground truth and once noise is added to the test images all 5 methods are carried

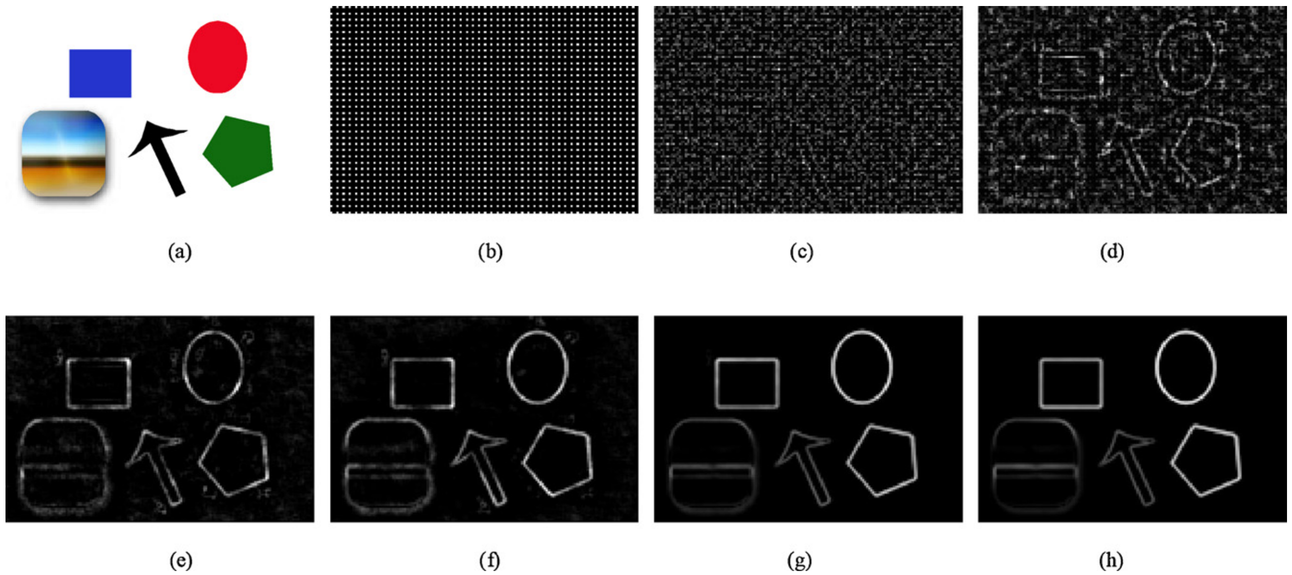

Fig. 5. Feature extraction for different number of iterations with 1650 ants and $\rho = 0.1$. (a): original image, (b): 1 iteration, (c): 10 iterations, (d): 20 iterations, (e): 50 iterations, (f): 100 iterations, (g): 1000 iterations, and (h): 5000 iterations.

out again. Using the extracted features before and after addition of noise, TP/FP values, a Score measure, and PSNR values for each technique are computed. The Score is calculated using Eq. (7) where *TP* denotes the number of true positive features and *TN* denotes the number of true negative features. The Score represents the number of pixels correct as measured by the ground truth. By Eq. (7) we are implying that each pixel, regardless of the fact that it may be an edge feature or not, has the same value in our evaluation, and the goal is to calculate how many pixels have been labelled correctly. PSNR is calculated using Eqs. (8) and (9). In Eq. (8) *IM* is the original image and naturally for a normalized greyscale image the maximum value would be 1. Eq. (9) illustrates the calculation of the Mean Squared Error (*MSE*) where *IM′* is the image after addition of noise. The images have a resolution of *m* by *n* pixels.

$$Score = TP + TN \tag{7}$$

$$PSNR = 10 \cdot \log_{10}\left(\frac{MAX^2(IM)}{MSE}\right) \tag{8}$$

$$MSE = \frac{1}{m \cdot n}\sum_{i=1}^{m-1}\sum_{j=1}^{n-1}[IM(i,j) - IM'(i,j)]^2 \tag{9}$$

Through the course of this study TP implies the pixels which have been determined as feature pixels in both the ground truth and the test image. FP represents the pixels which are not features in the ground truth but are incorrectly determined as features in the test image. TN denotes non-feature pixels of the ground truth which have been identified correctly as non-feature pixels in the test image. Finally, FN is non-feature pixels of the ground truth incorrectly labelled as feature pixels in the test image.

6. Experimental results and discussion

In this section, the proposed method and the algorithms described earlier are evaluated using an implementation in Matlab R2009a and executed on a 2.00 GHz Intel Core Duo CPU system with 2.00 GB of RAM. A simple synthesized image, as well as the benchmark test image, Lena, are used for testing the overall performance and the different parameters of the algorithm. We have also compared our results with four other well-known edge detection techniques: Canny, Sobel, Prewitt, and Log.

Prior to testing the ACED method and evaluating the results, an important factor to consider is the initial distribution of ants (see line 3 of Algorithm 1). A variety of different methods can be used; for instance, a random uniform distribution may seem to be a logical and effective choice. A Random distribution, however, will require a large enough set of ants such that the likelihood of an ant being present within a reasonable distance of a feature would be very high. Even with a very large set of agents, the distribution will vary for different trial runs and instances might occur where an important section of the image, which contains numerous features, would be ignored by the agents. Increasing the number of algorithm iterations could, in some cases, compensate for a poor initial distribution by providing more foraging time for the ants. The ants, given a large enough number of iterations, would have the chance of visiting the furthest feature pixels.

A more conservative approach to the initial distribution problem, however, would be safer and logical. It seems quite reasonable to distribute the ants such that at least one agent is guaranteed to be present on every $i \times j$ window of pixels. While placing an ant on every pixel might seem desirable, decreasing the ant resolution by assigning $i = j = 3$ yields equivalent system performance, while increasing the speed by a factor of 9. Therefore, in this research, the ants are distributed accordingly and such a distribution is used unless stated otherwise. This matter is investigated in more detail further on in this section.

In the remainder of this section of the paper the overall functionality of ACED is first illustrated by using a synthesized image shown in Fig. 5(a). This test image is designed to contain different colors, different geometrical shapes, a non-geometrical figure (arrow), and even a textured figure composed of different colors (the figure in the bottom left corner). This image is much simpler and contains fewer features when compared to Lena; therefore, it is easier to observe the changes in the performance of ACED as different parameters of the algorithm are tested and altered. The main evaluation of the algorithm and its performance in the presence of noise is then carried out using Lena.

6.1. Performance and parameters

The ACED algorithm is illustrated for different iteration numbers in Fig. 5(b)–(h). It is quite clear that pheromone accumulation

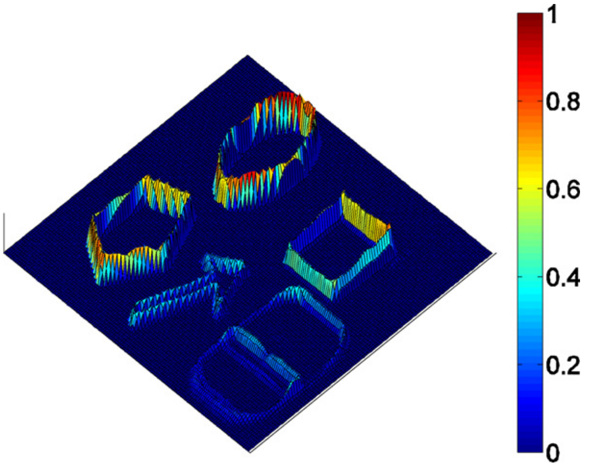

Fig. 6. Normalized pheromone accumulation for 1650 ants in Fig. 5(a) for 1000 iterations and $\rho = 0.1$.

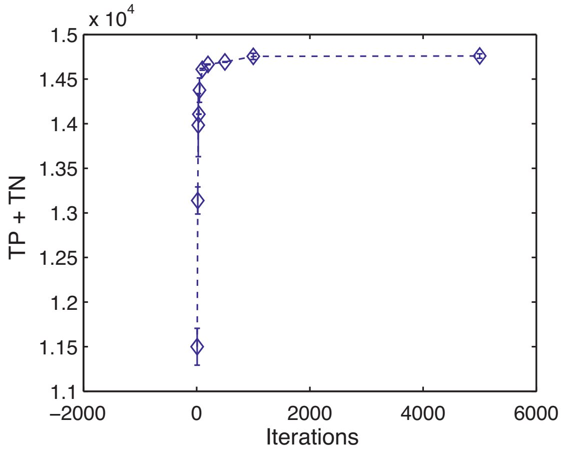

Fig. 7. Extracted features vs. iterations.

tends to merge towards the edges of the items in the image. Fig. 5(b) shows the outcome of the algorithm after a single iteration. The initial distribution of the agents can be seen in this image. As the number of iterations increases in Fig. 5(c) and (d), for 5 and 10 iterations, the extracted features become more solid and less pheromone is accumulated in areas with no features. In Fig. 5(d) the items are almost recognizable. By increasing the number of iterations to 50 in Fig. 5(e) the items are outlined almost completely. Some pheromone, however, is still present in feature-less sections of the image and all the real features are not extracted. Fig. 5(f) illustrates the output for 100 iterations. The pheromone is almost only present on the edges of the different objects of the image. The image shows few imperfections where some pheromone is present on non-edge sections of the image. Increasing the number of iterations to 1000 as shown in Fig. 5(g) results in an almost perfect edge detection except for the thick edges which will be dealt with later based on the method described in Section 5.1. In this image, all the edges are highlighted precisely and no pheromone is accumulated on feature-less pixels. Proceeding beyond a certain number of iterations does not increase the number of extracted features as the segmentation seems perfect. By increasing the number of iterations to 5000 no significant difference is observed in the output as illustrated by Fig. 5(h).

Fig. 6 presents the pheromone accumulation in 3D for Fig. 5(g). Based on this image a threshold of $T = 0.1$ is assigned for post processing to extract the weak edges of the textured rectangle in the image.

Earlier in this section the influence of the distribution of ants as well as number of foraging iterations were discussed. It was argued that increasing the foraging duration of the ants (iterations of the algorithm) will enhance the outcome. It was also discussed that distributing the ants such that all pixels are within a reasonable distance of the agents will again improve the outcome. The latter conclusion entails another property regarding the algorithm: increasing the number of ants will enhance the outcome. To test and evaluate these two ideas (number of iterations and number of ants) a ground truth set of features is required. Subjective evaluation indicates that the Sobel edge detector performs best with respect to the other three traditional edge detection techniques and therefore is assigned as the ground truth.

Fig. 7 illustrates the Score (Eq. (7)) versus the number of iterations after thresholding and thinning. It is evident that increasing the number of iterations increases the Score value. While this increase in the Score value is extreme for shorter foraging durations, there is no significant improvement beyond 1000 iterations; i.e., the marginal utility of added iterations is extremely low. The 1000 iteration mark may not be the optimum duration for all images; however, it does indicate that a Score-based termination criterion for the algorithm is possible, even desirable. Fig. 7 simply shows that more foraging time will most likely result in the discovery of more features; however, a threshold does exist beyond which there is marginal additional feature discovery. Subjective evaluation demonstrates that for images used in this paper, 1000 iterations are sufficient. To obtain the results for this figure each test was run 3 times. The data points are the average of results and the bars represent the standard deviations. More complex and larger images will require more foraging iterations as the ants will require more time to converge towards extraction of the final features. The stopping condition, as mentioned earlier, is defined by the number of iterations, pre-selected by the user. Based on Fig. 7, however, automatic conditions such as Score reaching equilibrium could be used. This was not done in the experiments reported in this paper. The complexity of the algorithm is O(number-of-ants × max-iterations). The number-of-ants is computed by dividing the $m \times n$ pixel image by 9.

Fig. 8 illustrates the feature extraction of the test image for a varying number of agents and pheromone evaporation rates. It is evident that by increasing the number of ants, the segmented sections of the image become more solid and distinct as shown in Fig. 8(f) where a precise segmentation is achieved. Fig. 8(g) illustrates the case where a very low pheromone evaporation rate is used. It can be observed that there is a large amount of left over pheromone in the environment, implying that incorrect features have been extracted. In Fig. 8(h), where ρ is increased to 0.7, many of the features have not been extracted by the agents. This is due to that fact that a high evaporation rate will prevent high densities of pheromone from accumulating.

Fig. 9 presents the Score versus the number of ants in which a similar trend to that observed in Fig. 7 is noticed. The Score increases as the number of ants increases. This improvement in performance is more extreme for smaller populations of ants. By increasing the number of agents beyond a certain point, insignificant improvement in the Score occurs. The optimum number of ants in this paper, as mentioned earlier, occurs when there are enough ants so that one agent starts its foraging process in any given 3 × 3 window of pixels in the image (1650 ants for Fig. 5(a)). Fig 9 confirms that a population of slightly over 1500 is optimal. The output is calculated using the same 3 sample approach used for Fig. 7.

The influence of the threshold, T, is illustrated by Fig. 10(a)–(c) which show the output for Fig. 8(f) after applying a threshold with 3

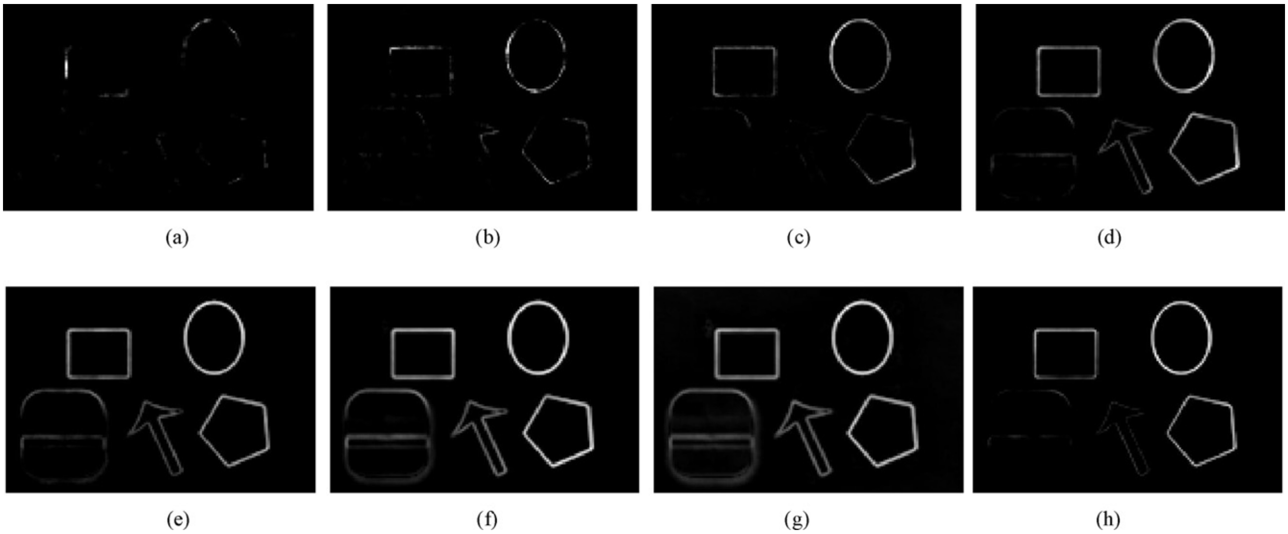

Fig. 8. Feature extraction for different number ants after 1000 iterations: (a): 35 ants & $\rho = 0.1$, (b): 70 ants & $\rho = 0.1$, (c): 150 ants & $\rho = 0.1$, (d): 294 ants & $\rho = 0.1$, (e): 600 ants & $\rho = 0.1$, (f): 1650 ants & $\rho = 0.1$, (g): 1650 ants & $\rho = 0.01$, (h): 1650 ants & $\rho = 0.7$.

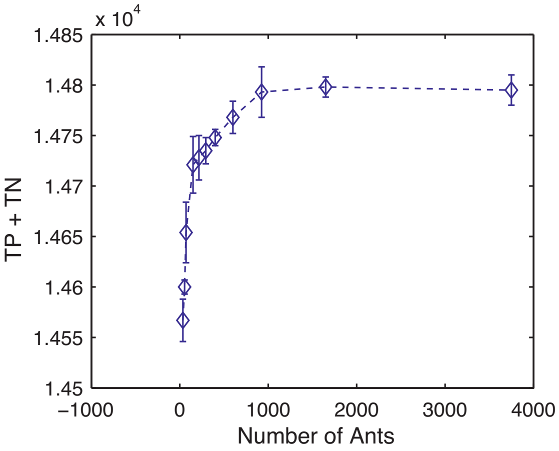

Fig. 9. Extracted features vs. number of ants.

different values. As expected, and discussed earlier, higher thresholds produce fewer, yet more confident, features while lower values for *T* result in more features which can be incorrect. In this research, *T* is always set to 0.1.

Finally, the performance of ACED is evaluated with respect to the 4 other methods using Lena as illustrated by Fig. 11(a). The experimental parameters used are as follows: number-of-ants = 1650 uniformly distributed according to the windowing mechanism described earlier, $\rho = 0.1$, $T = 0.1$, and max iterations = 1000. Fig. 11(b)–(f) highlight the performance of all 5 techniques. Subjectively, it appears that ACED is superior to the other 4 methods. While Sobel and Prewitt have missed out on some of the features which one would select manually, Canny and Log have extracted an excessive number of features. ACED, however, maintains a balance between extracting too few and too many features. Table 1 presents the TP and TN values for all 5 methods with respect to one another. All 5 techniques have been assigned as the ground truth, allowing the remaining 4 to be tested based on them. This is done because there is no *correct* ground truth. To populate the table, ACED is carried out 5 times and the average values are used. The remaining methods are evaluated only once since there is no stochastic property associated with them. The values in table are in the form of TP/TN except in the case of the diagonal elements that represent the feature and non-feature pixels computed by the respective method. The small standard deviations shown in the last row confirm an accurate convergence towards a uniform set of features. The table shows how the 5 edge detectors act differently in terms of TP, TN and Score (sum of TP and TN). The table confirms the fact that ACED maintains a middle ground in terms of TP and TN with respect to Sobel/Prewitt and Canny/Log. Also it can be seen that in terms of TP, ACED is most similar to Canny, and in terms of TN, it is most similar to Sobel.

While the Sum – average column of Table 1 clearly demonstrates that the ACED method has the best balanced behavior when compared to the other 4 methods, to better illustrate the performance of the proposed method, the results in Table 1 have been graphed in Figs. 12 and 13. Fig. 12 presents the TP versus TN cross-comparisons (CC) of each of the four methods with respect to one another and

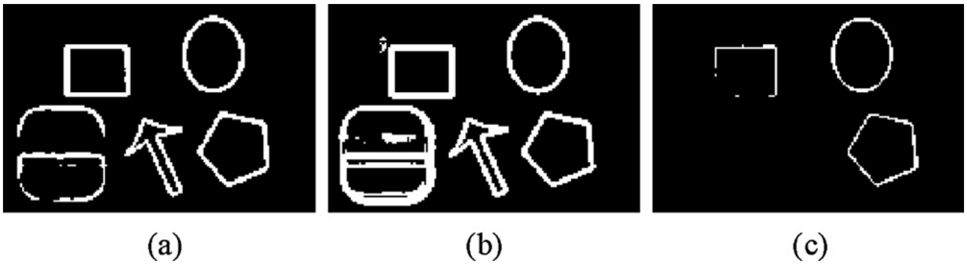

Fig. 10. Different threshold values: (a): $T = 0.1$, (b): $T = 0.01$, (c): $T = 0.5$.

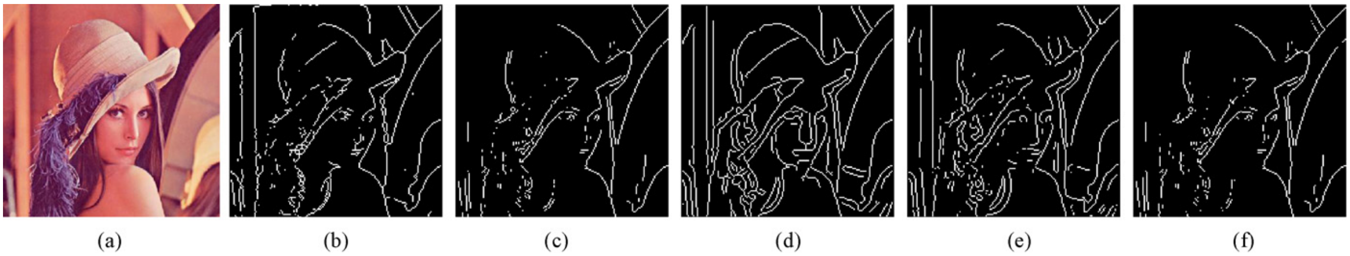

Fig. 11. Edge detection using different techniques (a): Lena, (b): ACED, (c): Sobel, (d): Canny, (e): Log, and (f): Prewitt.

Table 1
Feature extraction performance.

	ACED	Sobel	Canny	Prewitt	Log	Sum	Sum – average
ACED	1681/24888	705/24451	951/23386	684/24441	459/23361	2799/95639	−19/−315
Sobel		1142/25427	751/23725	1078/25374	412/23853	2946/97403	128/1449
Canny			2453/24116	757/23742	839/22969	3298/93822	480/−2132
Prewitt				1131/25438	409/23861	2928/98140	110/2186
Log					1986/24583	2119/94766	−699/−1188
SD	15/59	2/85	31/46	12/86	2/76		

The values are in the form of TP/TN.

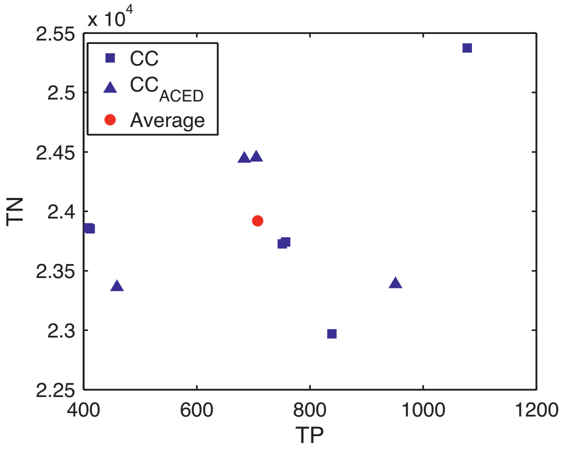

Fig. 12. CC values for Canny, Sobel, Prewitt, Log, and ACED.

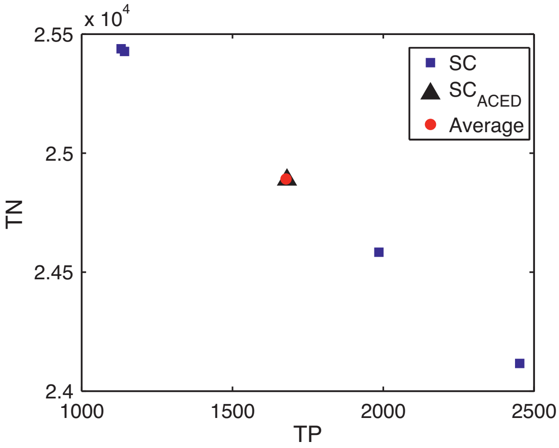

Fig. 13. SC values for Canny, Sobel, Prewitt, Log, and ACED.

with respect to ACED. Fig. 13 highlights the self-comparisons (SC) of the four methods as well as ACED. These figures further show the balanced performance of ACED in extraction of TP and TN features of the image. Specifically, Fig. 13 shows that the average of the TP/TN SC values for the 5 methods are almost exactly those computed by the ACED method. Given the resolution of the graph, the average and ACED points appear to be coincident.

6.2. Performance in the presence of noise

Noise is a common element present in many image processing procedures. In this section we will test the performance of the proposed method in the presence of salt and pepper noise.

Fig. 14(a) presents the original image, Fig. 11(a), after conversion to grayscale and adding salt and pepper noise. Fig. 14(b)–(f) illustrates the performance of all 5 methods in such situations. The parameters used in these experiments were the same as those used in the previous section.

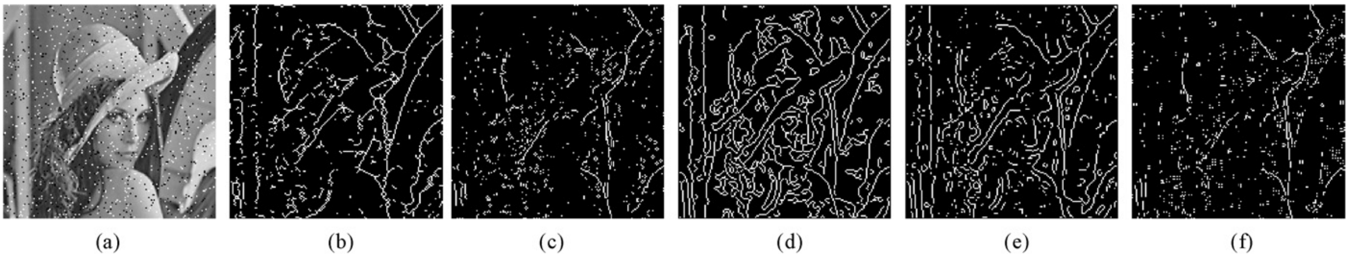

Fig. 14. Edge detection in the presence of salt and pepper noise using different techniques (a): Lena, (b): ACED, (c): Sobel, (d): Canny, (e): Log, and (f): Prewitt.

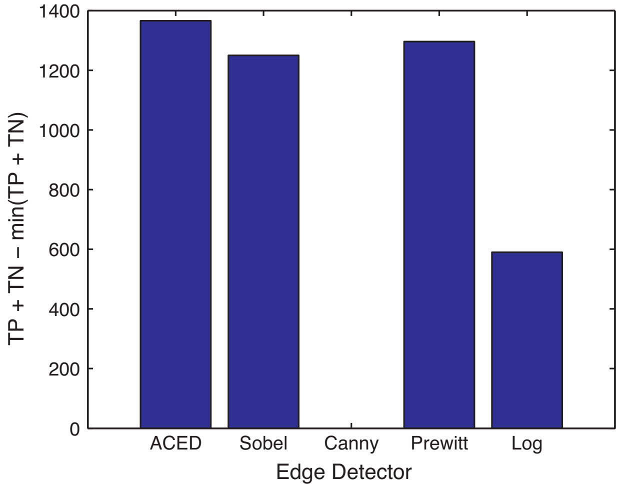

Fig. 15. Evaluation of edge detectors for noisy images.

The score of each edge detector is calculated. To do so, the TP and TN values are measured for each technique prior to and after the addition of noise. An average value over 5 test runs has been used for ACED. A standard deviation of 19 for TP and 65 for TN shows reasonable convergence. Fig. 15 shows the performance of the 5 edge detectors in the presence of noise where, once again, ACED tends to maintain a balance between too many and too few features. The overall performance is superior to all 4 conventional methods. The reason that $TP+TN-min(TP+TN)$ is assigned to the vertical axis is to illustrate the different performance characteristics more clearly. Since the sum, $TP+TN$, is a value of the order of 10^4, reducing all values by the $min(TP+TN)$ highlights the different values of $TP+TN$ for all edge detectors. This is why the measure for Canny appears as zero as it has the lowest value of the sum.

The PSNR values of the five methods were also calculated using Eq. (8). An average value of 13.03 dB for ACED compared to 12.67 dB, 9.95 dB, 12.81 dB, and 11.03 dB for Sobel, Canny, Prewitt, and Log respectively shows the strong performance of the ACED algorithm compared to these traditional techniques in the presence of noise. Using these values, ACED ranks first among the 5 methods tested on the noisy Lena image.

7. Conclusion

In this paper a swarm-based technique for extraction of image features and segmentation was proposed. The method was inspired by ant-based algorithms. The ants perform their foraging using a stochastic process which is based on the amount of pheromone in the environment.

For this algorithm we defined two types of pheromone: one which is laid by the agents (type-I) and its accumulation indicates the presence of features to be extracted, and the other (type-II) which is determined by the Euclidean norm of the image gradient that represents a static field.

Through numerous experiments, it is determined that increasing the iterations of the algorithm does improve the feature extraction capabilities of the proposed method, but up to a limit. Increasing the number of ants, however, can significantly influence the outcome. The initial distribution of the agents is also of great importance. By ensuring that at least one ant is present within any given window inside the image, the chances of robust extraction of features are greatly increased.

We also tested our proposed method for situations where salt and pepper noise is added to the image. Comparisons with Sobel, Canny, Prewitt, and Log edge detectors illustrated the advantages of our approach where it outperformed these methods in terms of Score (Eq. (7)) and PSNR (Eq. (8)).

Despite the desirable feature extraction performance of our proposed technique it should be noted that the computational time required for such algorithms are generally quite high. We would therefore recommend that our algorithm be employed where accurate feature extraction is critical to the application at hand as opposed to real-time performance. Furthermore, our method was executed using MATLAB which is relatively inefficient for iterative computations such as those performed in this paper. For future work, the algorithm will be implemented on a GPU using CUDA (NVIDIA's parallel computing architecture) which will significantly enhance the speed of the process. Similar implementations of ACO on GPUs have observed 100× speedup. Finally, parallel computing in MATLAB could also be explored.

References

[1] D.G. Lowe, Object recognition from local scale-invariant features, in: Proc. 7th IEEE Int. Conf. Computer Vision 2, 1999, pp. 1150–1157.
[2] E. Bonabeau, M. Dorigo, G. Theraulaz, Swarm Intelligence: From Natural to Artificial Systems, first ed., Oxford University Press, USA, 1999.
[3] R. Schoonderwoerd, J.L. Bruten, O.E. Holland, L.J.M. Rothkrantz, Ant-based load balancing in telecommunications networks, Adaptive Behavior 5 (2) (1996) 169–207.
[4] D. Teodorović, Swarm intelligence systems for transportation engineering: principles and applications, Transportation Research C: Emerging Technologies 16 (6) (2008) 651–667.
[5] A. Forestiero, C. Mastroianni, A swarm algorithm for a self-structured P2P information system, IEEE Transactions on Evolutionary Computation 13 (4) (2009) 681–694.
[6] J. Liu, Y.Y. Tang, Y.C. Cao, An evolutionary autonomous agents approach to image feature extraction, IEEE Transactions on Evolutionary Computation 1 (2) (1997) 141–158.
[7] K. Benatcha, M. Koudil, N. Benkhelat, Y. Boukir, ISA An algorithm for image segmentation using ants, in: IEEE Int. Symp. Industrial Electronics, 2008, pp. 2503–2507.
[8] X. Zhuang, G. Yang, H. Zhu, A model of image feature extraction inspired by ant swarm system, in: 4th International Conference on Natural Computation 7, 2008, pp. 553–557.
[9] X. Zhuang, Image segmentation by ant swarm – a case study of digital signal processing with biological mechanism of intelligence, in: 11th IEEE Workshop Digital Signal Processing, 2004, pp. 143–146.
[10] S. Ouadfel, M. Batouche, MRF-based image segmentation using ant colony system, Electronic Letters on Computer Vision and Image Analysis 2 (2) (2003) 12–24.
[11] S. Ouadfel, M. Batouche, Ant colony system with local search for Markov random field image segmentation, in: Proc. 2003 Int. Conf. Image Processing, 2003, pp. 133–136.
[12] E. Lakehal, A swarm intelligence based approach for image feature extraction, in: Int. Conf. Multimedia Computing and Systems, 2009, pp. 31–35.
[13] R.N. Khushaba, A. Al-Ani, A. Alsukker, A. Al-Jumaily, A combined ant colony and differential evolution feature selection algorithm, Ant Colony Optimization and Swarm Intelligence 5217 (2008) 1–12.
[14] T. Mirzayans, N. Parimi, P. Pilarski, C. Backhouse, L. Wyard-Scott, P. Musilek, A swarm-based system for object recognition, Neural Network World 15 (3) (2005) 243–255.
[15] F. Keshtkar, W. Gueaieb, Segmentation of dental radiographs using a swarm intelligence approach, in: Canadian Conf. Electrical and Computer Engineering, 2006, pp. 328–331.
[16] J. Du, L. Zhou, X. Xu, L. Yang, Z. Shi, Image feature extraction with the artificial ant colony, in: 2010 International Conference on Intelligent Computation Technology and Automation 2, 2010, pp. 466–469.
[17] Q. Wan, Y. Wang, Detecting moving objects by ant colony system in a MAP-MRF framework, in: 2010 International Conference on E-Product E-Service and E-Entertainment, 2010, pp. 1–4.
[18] Y. Wang, M. Zhu, Joint transform correlator based on joint image feature extraction using swarm intelligence method, in: International Conference on Mechatronics and Automation, 2009, pp. 4964–4969.
[19] V. Ramos, F. Almeida, Artificial ant colonies in digital image habitats – a mass behaviour effect study on pattern recognition, in: Proceedings of Int. Workshop on Ant Algorithms (From Ant Colonies to Artificial Ants), 2000, pp. 113–116.
[20] Z. Ye, W. Liu, H. Chen, E. Zhao, A novel feature selection approach based on swarm intelligence, in: International Workshop on Intelligent Systems and Applications, 2009, pp. 1–4.
[21] L. Zheng, Q. Pan, G. Li, J. Liang, Improvement of grayscale image segmentation based on PSO algorithm, in: 4th Int. Conf. Computer Sciences and Convergence Information Technology, 2009, pp. 442–446.

[22] Y. Owechko, S. Medasani, N. Srinivasa, Classifier swarms for human detection in infrared imagery, in: Computer Vision and Pattern Recognition Workshop, 2004.

[23] O. Sjahputera, J.M. Keller, Particle swarm over scene matching, in: Proc. 2005 IEEE Swarm Intelligence Symposium, 2005, pp. 108–115.

[24] M. Gudmundsson, E.A. El-Kwae, M.R. Kabuka, Edge detection in medical images using a genetic algorithm, IEEE Transactions on Medical Imaging 17 (3) (1998) 469–474.

[25] J. Canny, A computational approach to edge detection, IEEE Transactions on Pattern Analysis and Machine Intelligence 8 (1986) 679–714.

[26] E. Trucco, A. Verri, Introductory Techniques for 3-D Computer Vision, first ed., Prentice Hall, 1998.

[27] R.C. Gonzalez, R.E. Woods, Digital Image Processing, Addison-Wesley, 1992.

[28] W. Chen, X.X. Chen, L. Zhou, Multifractal analysis of edge images detected with classical edge detector, in: International Conference on Measuring Technology and Mechatronics Automation, 2, 2010, pp. 725–727.

[29] M. Dorigo, V. Maniezzo, A. Colorni, Ant system: optimization by a colony of cooperating agents, IEEE Transactions on Systems, Man, and Cybernetics: B 26 (1) (1996) 29–41.

[30] M. Sezgin, B. Sankur, Survey over image thresholding techniques and quantitative performance evaluation, Journal of Electronic Imaging 13 (1) (2004) 146–168.

[31] L. Lam, S.-W. Lee, C.Y. Suen, Thinning methodologies-a comprehensive survey, IEEE Transactions on Pattern Analysis and Machine Intelligence 14 (9) (1992) 869–885.

[32] Z. Guo, R.W. Hall, Parallel thinning with two-subiteration algorithms, Communications of the ACM 32 (3) (1989) 359–373.

[33] N.L. Fernandez-Garcıa, A. Carmona-Poyato, R. Medina-Carnicer, F.J. Madrid-Cuevas, Automatic generation of consensus ground truth for the comparison of edge detection techniques, Image and Vision Computing 26 (2008) 496–511.

[34] J. Gauch, C.-W. Hsia, A comparison of three color image segmentation algorithms in four color spaces, Visual Communications and Image Processing 1818 (1992) 1168–1181.

[35] S.-Y. Zhu, K.N. Plataniotis, A.N. Venetsanopoulos, Comprehensive analysis of edge detection in color image processing, Optical Engineering 38 (4) (1999) 612–625.

[36] W.K. Pratt, Digital Image Processing, Wiley, Hoboken, United States of America, 2007.

S. Ali Etemad received his M.A.Sc. in Electrical and Computer Engineering from Carleton University, Ottawa, Canada, in 2009. His area of research was on machine learning, artificial intelligence, pattern recognition, and image and video processing focused on human motion analysis. He is currently working towards his Ph.D. at the same program where he works as a research and teaching assistant.

Tony White received his M.A. in Theoretical Physics from Cambridge University in 1981. He subsequently received an M.C.S. in Computer Science from Carleton University in 1993 and a Ph.D. in 2000, also from Carleton. With over 20 years in private industry, he is now an associate professor of Computer Science at Carleton, where his interests include Complex Systems and Swarm Intelligence. He has published over 100 papers and is the co-author of 8 patents with others pending.