

Ant Colony Optimization for Image Edge Detection

ANNA VERONICA C. BATERINA, CARLOS M. OPPUS
Department of Electronics and Communications Engineering
Ateneo de Manila University
Katipunan Avenue, Loyola Heights, Quezon City
PHILIPPINES
nbaterina@gmail.com, coppus@ateneo.edu

Abstract: - Ant colony optimization is a population-based metaheuristic that mimics the foraging behavior of ants to find approximate solutions to difficult optimization problems. This paper presents an ACO-based technique for image edge detection. The proposed method establishes a pheromone matrix that represents the edge information at each pixel based on the routes formed by ants dispatched on the image. The movement of the ants is guided by the local variation of the image's intensity values. The proposed ACO-based approach takes advantage of the improvements introduced in ant colony system, one of the main extensions to the original ant system. Experimental results show the success of the technique in extracting edges from a digital image.

Key-Words: - ant colony optimization, image edge detection, swarm algorithm

1 Introduction

Ant colony optimization (ACO) is a nature-inspired optimization algorithm [1], [2], that is motivated by the natural foraging behavior of ant species. Ants deposit pheromone on the ground to mark paths between a food source and their colony, which should be followed by other members of the colony. Over time, pheromone trails evaporate. The longer it takes for an ant to travel down the path and back again, the more time the pheromones have to evaporate. Shorter – and thus, favorable – paths get marched over faster and receive greater compensation for pheromone evaporation. Pheromone densities remain high on shorter paths because pheromone is laid down faster. This positive feedback mechanism eventually leads the ants to follow the shorter paths. It is this natural phenomenon that inspired the development of the ACO metaheuristic. Dorigo et al. [3] proposed the first ACO algorithm, ant system (AS) [1]-[3]. Since then, extensions to AS have been developed, such as ant colony system (ACS) [1], [2] and MAX-MIN ant system (MMAS) [1], [2]. ACO has been used to solve a wide variety of optimization problems.

In this paper, an ACO-based method for image edge detection is proposed. Edge detection aims to localize the boundaries of objects in an image and is a basis for image analysis and machine vision. In the proposed ACO-based approach, artificial ants are distributed over an image and moved based on the local variation of the image's intensity values. The movement of ants establishes a pheromone matrix that represents the edge information at each pixel location in the image. The method makes use of the improvements introduced in ACS, a variant ACO which is based on the original AS.

2 Image Edge Detection

Image edge detection refers to the extraction of the edges in a digital image. It is a process whose aim is to identify points in an image where discontinuities or sharp changes in intensity occur. This process is crucial to understanding the content of an image and has its applications in image analysis and machine vision. It is usually applied in initial stages of computer vision applications.

The purpose of detecting sharp changes in image intensity is to capture significant events and changes in the physical properties of the world. Under general assumptions about the image formation process, discontinuities in intensity usually correspond to discontinuities in depth, discontinuities in surface orientation, changes in material properties, and variations in scene illumination. A representation of an image in terms of its edges is compact because it uses a set of one-dimensional curves instead of a two-dimensional pattern. Hence, edges have been used as main features in many computer vision algorithms.

Conventional approaches to edge detection are computationally expensive because each set of operations is conducted for each pixel. In conventional approaches, the computation time quickly increases with the size of the image. An ACO-based approach has the potential of overcoming the limitations of conventional methods. Furthermore, it can readily be parallelized, which makes the algorithm easily adaptable for distributed systems.

Several ACO-based approaches to the edge detection problem have been proposed [4]-[7]. Previously reported ACO-based approaches to image edge detection, to the best of the author's knowledge, all use a decision rule

that is based on AS. AS is the first ACO algorithm. Since its development, a number of extensions have emerged. One of the successful ones is ACS. This paper presents a technique derived from distinguishing features of ACS. One of the significant aspects of ACS is the form of decision rule used, the pseudorandom proportional rule. The approach presented in this paper uses such rule in the tour construction process.

3 Proposed Edge Detection Method

This section provides a theoretical discussion on the ant colony optimization metaheuristic and ant colony system, one of the main extensions to AS. The theoretical discussion is followed by a discussion on the proposed ACO-based image edge detection technique.

3.1 Ant Colony Optimization

ACO is a probabilistic technique for finding optimal paths in fully connected graphs through a guided search, by making use of the pheromone information. This technique can be used to solve any computational problem that can be reduced to finding good paths on a weighted graph. In an ACO algorithm, ants move through a search space, the graph, which consists of nodes and edges. The movement of the ants is probabilistically dictated by the transition probabilities. The transition probability reflects the likelihood that an ant will move from a given node to another. This value is influenced by the heuristic information and the pheromone information. The heuristic information is solely dependent on the instance of the problem. Pheromone values are used and updated during the search. Fig. 1 shows a pseudocode of the general procedure in an ACO metaheuristic.

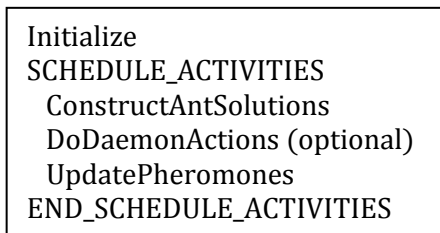


Fig. 1. ACO metaheuristic

The initialization step is performed at the beginning. In this step, the necessary initialization procedures, such as setting the parameters and assigning the initial pheromone values, are performed.

The SCHEDULE_ACTIVITIES construct regulates the activation of three algorithmic components: (1) the construction of the solutions, (2) the optional daemon

actions that improve these solutions, and (3) the update of the pheromone values. This construct is repeated until the termination criterion is met. An execution of the construct is considered an iteration.

ConstructAntSolutions. In a construction process, a set of artificial ants construct solutions from a finite set of solution components from a fully connected graph that represents the problem to be solved. A construction process contains a certain number of construction steps. Ants traverse the graph until each has made the target number of construction steps. The solution construction process starts with an empty partial solution, which is extended at each construction step by adding a solution component. The solution component is chosen from a set of nodes neighboring the current position in the graph. The choice of solution components is done probabilistically. The exact decision rule for choosing the solution components varies across different ACO variants. The most common decision rule is the one used in the original AS. On the n^{th} construction process, the k^{th} ant moves from node i to node j according to the transition probability $p_{ij}^{(n)}$, the probability that an ant will move from node i to node j (i.e., an ant in node i will move to node j). The AS decision rule is based on the transition probability given by

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

where $\tau_{ij}^{(n-1)}$ is the quantity of pheromone on the edge from node i to node j ; η_{ij} is the heuristic information of the edge from node i to node j ; Ω_i is the neighborhood nodes for the ant given that it is at node i ; α and β are constants that control the influence of the pheromone and heuristic information, respectively, to the transition probability. $\sum_{j \in \Omega_i} (\tau_{ij}^{(n-1)})^\alpha (\eta_{ij})^\beta$ is a normalization factor to limit the values of $p_{ij}^{(n)}$ within $[0,1]$.

DoDaemonActions. Once solutions have been constructed, there might be a need to perform additional actions before updating the pheromone values. Such actions, usually called daemon actions, are those that cannot be performed by a single ant. Normally, these are problem specific or centralized actions to improve the solution or search process.

UpdatePheromones. After each construction process and after the daemon actions have been performed, the pheromone values are updated. The goal of the pheromone update is to increase the pheromone values associated with good solutions and decrease those associated with bad ones. This is normally done by decreasing all the pheromone values (evaporation) and increasing the pheromone values associated with the

good solutions (deposit). Pheromone evaporation implements a form of forgetting, which prevents premature convergence to sub-optimal solutions and favors the exploration of new areas in the graph. The exact way by which the pheromone values are updated varies across different ACO variants. The AS pheromone update follows the equation

$$\tau_{i,j}^{(n)} = (1 - \rho) \cdot \tau_{i,j}^{(n-1)} + \sum_{k=1}^K \Delta\tau_{i,j}^{(k)} \quad (2)$$

where $\rho \in (0,1]$ is the pheromone evaporation rate; K is the number of ants; $\Delta\tau_{i,j}^{(k)}$ is the quantity of pheromone laid on edge (i,j) by the k^{th} ant:

$$\Delta\tau_{i,j}^{(k)} = \begin{cases} \frac{1}{L_k}, & \text{if ant } k \text{ used edge } (i,j) \\ 0, & \text{otherwise} \end{cases} \quad (3)$$

where L_k is the tour length of the k^{th} ant. The tour length is determined according to some user-defined rule. The rule depends on the nature of the problem to be solved, but it must always be such that desirable routes have smaller tour lengths. In general, the tour length is a function of the heuristic information associated with the edges belonging to the tour.

ACS is the first major improvement to AS. ACS has three significant differences from AS. First, it uses a more aggressive decision rule, the so-called pseudorandom proportional rule, which strengthens the exploitation of the search experience accumulated by the ants. Second, pheromone evaporation and deposit are done only on edges belonging to the best-so-far tour, as opposed to AS where pheromone evaporation is done on all edges and pheromone deposit is done on edges belonging to any solution constructed in the current iteration. Third, each time an ant uses an edge to move from one node to another, it removes some pheromone from that edge to increase the exploration of other areas. The process of removing pheromones from edges as they are crossed is called local pheromone update. The local update counterbalances the effect of the greedy decision rule, which favors the exploitation of the pheromone information.

3.1.1 ACS Tour Construction

In the pseudorandom proportional rule, the transition probability depends on a random variable q that is uniformly distributed over $[0,1]$ and a parameter q_0 . If $q \leq q_0$, then the transition that maximizes $\tau_{i,j}\eta_{i,j}^\beta$ is chosen; otherwise, the AS probabilistic decision rule (Eq. 1), with $\alpha = 1$, is used. The value of q_0 determines the degree of exploration of the ants: with probability q_0 , the ant chooses the transition with the highest

$\tau_{i,j}\eta_{i,j}^\beta$, while with probability $(1 - q_0)$, it performs a biased exploration of the edges. The balance between biased exploration and pheromone exploitation can be tweaked by adjusting the value of q_0 .

3.1.2 ACS Global Pheromone Update

The global pheromone update is performed only on the best-so-far solution according to the equation

$$\tau_{i,j}^{(n)} = (1 - \rho) \cdot \tau_{i,j}^{(n-1)} + \rho \cdot \Delta\tau_{i,j}^{(k_{bs})} \quad (4)$$

where $\Delta\tau_{i,j}^{(k_{bs})}$ is the amount of pheromone deposited by the ant that produced the best-so-far-solution, which is normally

$$\Delta\tau_{i,j}^{(k_{bs})} = \begin{cases} \frac{1}{L_{k_{bs}}}, & \text{if ant } k_{bs} \text{ used edge } (i,j) \\ 0, & \text{otherwise} \end{cases} \quad (5)$$

where L_k is the tour length associated with the best-so-far solution.

Another thing that makes the global update in ACS different from that in AS is that in ACS, the pheromone deposited is decreased with a factor of ρ , the evaporation rate, which results to a new pheromone value that is a weighted average between the old value and the amount deposited in the current iteration.

3.1.1 ACS Local Pheromone Update

Local pheromone update is interleaved with the tour construction process and applies each time and immediately after an ant traverses an edge during the construction process. After each construction step, an ant updates the pheromone value associated with the last edge that it has traversed based on the equation

$$\tau_{i,j}^{(n)} = (1 - \varphi) \cdot \tau_{i,j}^{(n)} + \varphi \cdot \tau_0 \quad (6)$$

where $\varphi \in (0,1]$ is the pheromone decay coefficient and τ_0 is the initial pheromone value.

Local pheromone update diversifies the search by having the effect of decreasing the desirability of edges that have already been traversed.

3.2 ACO-based Image Edge Detection

In the proposed method, ants move on a two-dimensional image – stepping from one pixel to another – to construct a pheromone matrix, from which the edge information is determined to extract the edges of the image. The movement of the ants is steered by the local variation of the image's pixel intensity values.

In the model used, each pixel in the image represents both a node and an edge in the graph. A pixel represents a node because locations in the graph are associated with pixel locations – ants move from one pixel to another. At the same time, it also represents an edge because the heuristic information is determined from the local variation of the image's intensity values and hence, is associated with a pixel location in the image. The components of the pheromone and transition matrices are associated with pixels in the image.

The algorithm consists of three main steps. The first is the initialization process. The second is the iterative construction-and-update process, where the goal is to construct the final pheromone matrix. The construction-and-update process is performed several times, once per iteration. The final step is the decision process, where the edges are identified based on the final pheromone values.

3.2.1 Initialization Process

In the initialization process, each of the K ants is assigned a random position in the $M_1 \times M_2$ image. The initial value of each element in the pheromone matrix is set to a constant τ_{init} , which is small but non-zero. Also, the heuristic information matrix is constructed based on the local variation of the intensity values. The heuristic information is determined during initialization since it is dependent only on the pixel values of the image, thus, constant.

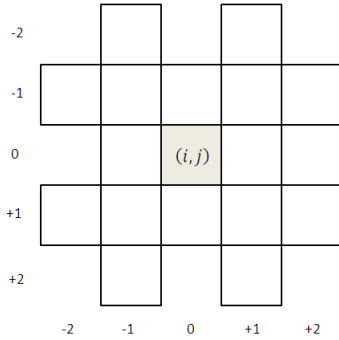


Fig. 2. A local configuration for computing the intensity variation at (i, j) .

The heuristic information at pixel (i, j) is determined by the local statistics at that position:

$$\eta_{i,j} = \frac{V_c(I_{i,j})}{\sum_{i=1}^{M_1} \sum_{j=1}^{M_2} V_c(I_{i,j})} \quad (7)$$

where $I_{i,j}$ is the intensity value of the pixel at (i, j) .

$V_c(I_{i,j})$ is a function that operates on the local group of pixels (Fig. 2) around the pixel (i, j) . It depends on the

variation of the intensity values on the local group, and is given by

$$V_c(I_{i,j}) = |I_{i-2,j-1} - I_{i+2,j+1}| + |I_{i-2,j+1} - I_{i+2,j-1}| + |I_{i-1,j-2} - I_{i+1,j+2}| + |I_{i-1,j+2} - I_{i+1,j-2}| + |I_{i-1,j-1} - I_{i+1,j+1}| + |I_{i-1,j+1} - I_{i+1,j-1}| + |I_{i-1,j} - I_{i+1,j}| + |I_{i,j-1} - I_{i,j+1}| \quad (8)$$

$\sum_{i=1}^{M_1} \sum_{j=1}^{M_2} V_c(I_{i,j})$ is a normalization factor.

3.2.2 Iterative Construction and Update Process

On every iteration, each ant moves across the image, from one pixel to the next, until it has made L construction steps (a construction step consists of a single movement from one pixel to another). An ant moves from the pixel (i_0, j_0) to an adjacent pixel (i, j) according to the pseudorandom proportional rule. The transition probability for the biased exploration is given by

$$p_{(i_0, j_0), (i, j)}^{(n)} = \frac{(\tau_{i,j}^{(n-1)})^\alpha (\eta_{i,j})^\beta}{\sum_{(i,j) \in \Omega_{(i_0, j_0)}} (\tau_{i,j}^{(n-1)})^\alpha (\eta_{i,j})^\beta} \quad (9)$$

where $\tau_{i,j}^{(n-1)}$ is the pheromone value for pixel (i, j) , $\Omega_{(i_0, j_0)}$ is the neighborhood pixels of pixel (i_0, j_0) , $\eta_{i,j}$ is the heuristic information at pixel (i, j) . The constants $\alpha = 1$ and β control the influence of the pheromone and heuristic information, respectively.

Each time an ant visits a pixel, it immediately performs a local update on the associated pheromone. The amount of pheromone on the pixel (i, j) on the n^{th} iteration, $\tau_{i,j}^{(n)}$, is updated based on the equation for ACS local pheromone update:

$$\tau_{i,j}^{(n)} = (1 - \varphi) \cdot \tau_{i,j}^{(n-1)} + \varphi \cdot \tau_{init} \quad (10)$$

where $\varphi \in (0, 1]$ is the pheromone decay coefficient and τ_{init} is the initial pheromone value. Local pheromone updates are interleaved with the solution construction process; the pheromone values change within the iteration.

The permissible range of movement of the ants is obtained from the 8-connectivity neighborhood (Fig. 3). An ant can move to any adjacent pixel. But, this is restricted by the condition that an ant moves only to a node that it has not recently visited. This is to prevent the ants from visiting the same set of nodes repeatedly. In order to keep track of the recently visited nodes, each ant has a memory.

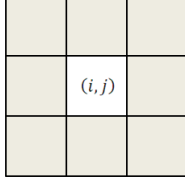


Fig. 3. Neighborhood pixels (shaded) of the pixel (i, j) : 8-connectivity neighborhood.

After all the ants finish the construction process, global pheromone update is performed on pixels that have been visited by at least one ant:

$$\tau_{i,j}^{(n)} = (1 - \rho) \cdot \tau_{i,j}^{(n-1)} + \rho \cdot \sum_{k=1}^K \Delta\tau_{i,j}^{(k)} \quad (11)$$

where $\Delta\tau_{i,j}^{(k)}$ is the amount of pheromone deposited by the k^{th} ant on pixel (i, j) . $\Delta\tau_{i,j}^{(k)}$ is assigned to be equal to $\eta_{i,j}$. Its reciprocal, $\frac{1}{\eta_{i,j}}$, can be interpreted as the tour length. This definition of the tour length satisfies the requirement that desirable routes have smaller tour lengths. Desirable routes are those that pass along pixels with higher local variation in intensity (i.e., larger $\eta_{i,j}$ or smaller $\frac{1}{\eta_{i,j}}$). Pheromones for unvisited pixels remain unchanged.

Global pheromone update for the proposed method does not exactly follow the ACS approach. This is because some details of the ACS approach do not suit the nature of the proposed edge-detection technique. One of the first problems ACO was made to solve is the traveling salesman problem (TSP). The nature of the ACO-based approach to TSP is different from the nature of the ACO-based edge detection technique described in this paper.

There are two main differences in the global pheromone update. First, there is no selection of a best-so-far tour; all visited pixels are updated. In ACS, only the solution components belonging to the best-so-far solution is updated. Having a best-so-far solution makes sense for the ACO-based approach to TSP because each ant creates a tour that is a possible complete solution to the problem. In the ACO-based edge detection approach, however, an individual ant does not aim to produce a complete possible solution to the problem (i.e., a complete trace of the image edges). Instead, the goal of each ant is to produce only a partial edge trace in the image. The collected action of the ants produces a pheromone matrix, which can be used to extract a complete edge trace. With this, it is not appropriate to select a best-so-far solution during the construction process. Therefore, all edges that have been visited by at least a one ant undergo a global pheromone update.

The second difference is in the function for the amount of pheromone deposited. In ACS, the amount of pheromone deposited on edges within a single tour is equal for all the edges and is a function of the cost of the tour as a whole. In the proposed method, the amount of pheromone deposited is a function only of the heuristic information in the specific pixel; the amount of pheromone deposited is not necessarily the same for all edges within the same tour.

3.2.3 Decision Process

The final pheromone matrix is used to classify each pixel either as an edge or a non-edge. The decision is made by applying a threshold on the final pheromone matrix $\tau^{(N)}$. The threshold value is computed based on the method described in [8], also known as the Otsu thresholding technique.

```

Do initialization procedures
for each iteration n = 1:N do
  for each construction_step l = 1:L do
    for each ant k = 1:K do
      Select and go to next pixel
      Update pixel's pheromone (local)
    end
  end
  Update visited pixels' pheromones (global)
end

```

Fig. 4. ACO-based image edge detection

Fig. 4 shows a pseudocode of the proposed method.

4 Experimental Results

Experiments were conducted using several test images. The proposed ACO-based edge detection method was implemented using C++. The program is run on a PC with an Intel Core 2 Duo 1.5 GHz CPU and 2 GB RAM. The execution time for a 256×256 image, with $N = 20$, $L = 40$, and $K = 256$, is around 3 seconds. Two test images, *Peppers* and *Mandrill*, are shown in Fig. 5.



Fig. 5. Test images *Peppers* and *Mandrill*.

Fig. 6 and Fig. 7 show the extracted edges of the two images at different values of q_0 . Fig. 8 and Fig. 9 show the extracted edges of the two images at different values of N . For unvaried parameters, the values used were $\alpha = 1.0$ (ACS constraint), $\beta = 1.0$, $\rho = 0.1$, $\varphi = 0.05$, $q_0 = 0.75$, $\tau_{\text{init}} = 0.0000001$, $K = \lfloor \sqrt{M_1 \times M_2} \rfloor$, $N = 10$, $L = 40$.



Fig. 6. Extracted edges of the 256×256 image *Peppers* at different values of q_0 : a. 0; b. 0.25; c. 0.5; d. 1.

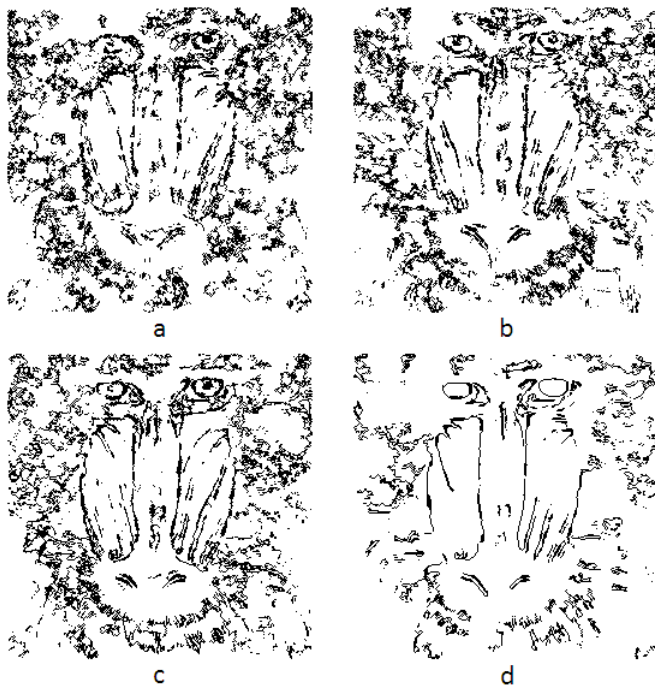


Fig. 7. Extracted edges of the 256×256 image *Mandril* at different values of q_0 : a. 0; b. 0.25; c. 0.5; d. 1.

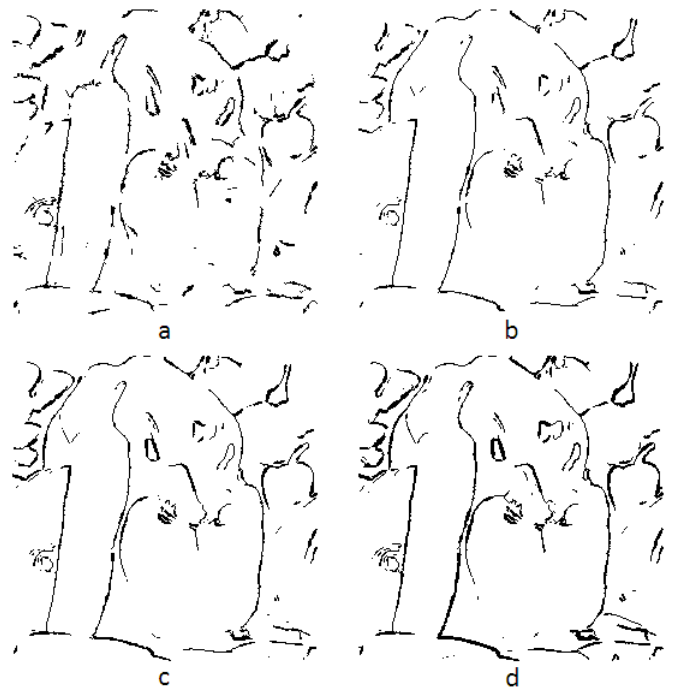


Fig. 8. Extracted edges of the 256×256 image *Peppers* at different values of N : a. 1; b. 5; c. 10; d. 20.

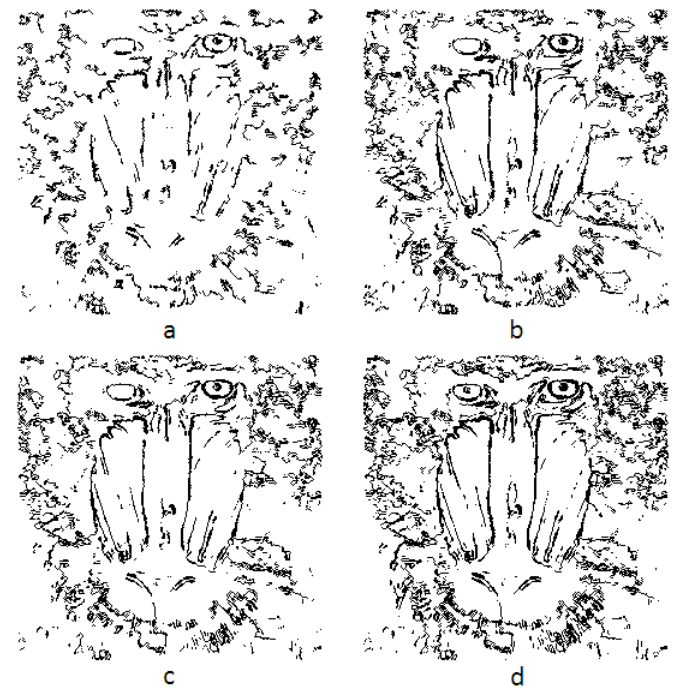


Fig. 9. Extracted edges of the 256×256 image *Mandril* at different values of N : a. 1; b. 5; c. 10; d. 20.

5 Conclusion

An ACO-based image edge detection method that takes advantage of the improvements introduced in ACS has been successfully implemented. The proposed method produced acceptable results within reasonable amounts of time.

References:

- [1] M. Dorigo and T. Stutzle, *Ant Colony Optimization*, Cambridge: MIT Press, 2004.
- [2] M. Dorigo, V. Maniezzo, and A. Colorni, Ant System: Optimization by a colony of cooperating agents, *IEEE Transactions on Systems, Man and Cybernetics*, Part B, Vol.26, 1996, pp. 29-41.
- [3] M. Dorigo. (2007) Ant colony optimization, *Scholarpedia*, 2(3):1461. [Online]. Available: http://www.scholarpedia.org/article/Ant_colony_optimization.
- [4] A. Rezaee, Extracting Edge of Images with Ant Colony, *Journal of Electrical Engineering*, Vol.59, No.1, 2008, pp. 57-59.
- [5] J. Tian, W. Yu, and S. Xie, An Ant Colony Optimization Algorithm for Image Edge Detection, *IEEE Congress on Evolutionary Computation*, 2008.
- [6] H. Nezamabadi-pour, S. Saryazdi, and E. Rashedi, Edge Detection Using Ant Algorithms, *Soft Computing*, Vol.10, 2006, pp. 623-628.
- [7] X. Zhuang, Edge Feature Extraction in Digital Images with the Ant Colony System, *IEEE International Conference in Computational Intelligence for Measurement Systems and Applications*, 2004.
- [8] N. Otsu, A Threshold Selection Method from Gray-level Histograms, *IEEE Transactions on Systems, Man and Cybernetics*, Vol.9, No.1, 1979, pp. 62-66.