# Image Edge Detection Using Ant Colony Optimization

Anna Veronica Baterina and Carlos Oppus

***Abstract*—**Ant colony optimization (ACO) is a population-based metaheuristic that mimics the foraging behavior of ants to find approximate solutions to difficult optimization problems. It can be used to find good solutions to combinatorial optimization problems that can be transformed into the problem of finding good paths through a weighted construction graph. In this paper, an edge detection technique that is based on ACO is presented. The proposed method establishes a pheromone matrix that represents the edge information at each pixel based on the routes formed by the ants dispatched on the image. The movement of the ants is guided by the local variation in the image's intensity values. The proposed ACO-based edge detection method 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.

***Keywords*—**ant colony optimization, image edge detection, swarm algorithm.

## I. 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], [2], [3]. Since then, extensions to AS have been developed. One of the successful ones is ant colony system (ACS) [1], [2], [4]. 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.

Manuscript received January 15, 2010. This work was funded by the Philippine Department of Science and Technology under the Engineering Research and Development for Technology program and the Ateneo de Manila University.

A. V. Baterina is taking up a Master's degree in Electronics Engineering in Ateneo de Manila University, Katipunan Avenue, Loyola Heights, Quezon City, 1108 Philippines (phone: +632-911-7715; e-mail: nbaterina@gmail.com).

C. Oppus is with the Department of Electronics, Computer, and Communications Engineering, Ateneo de Manila University, Katipunan Avenue, Loyola Heights, Quezon City, 1108 Philippines (e-mail: coppus@ateneo.edu).

## II. 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.

Edge detection aims to localize the boundaries of objects in an image and is a basis for many image analysis and machine vision applications. 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 [5]-[9]. Previously reported ACO-based approaches to image edge detection, to the best of the authors' knowledge, all use a decision rule that is based on AS. This paper presents a technique that is derived from improvements introduced in ACS, one of the main extensions to AS. 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.

## III. Proposed Edge Detection Method

This section provides a theoretical discussion on the ant colony optimization metaheuristic and ACS, the first major improvement to AS. The theoretical discussion is followed by a discussion on the proposed ACO-based image edge detection technique.

### *A. Ant Colony Optimization and ACS*

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.

```
Initialize
SCHEDULE_ACTIVITIES
  ConstructAntSolutions
  DoDaemonActions (optional)
  UpdatePheromones
END_SCHEDULE_ACTIVITIES
```

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 nth construction process, the kth ant moves from node i to node j according to the transition probability $p_{i,j}^{(n)}$, 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

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

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

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

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 kth ant:

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

where $L_k$ is the tour length of the kth 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 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.

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 $q_0$.

2. ACS Global Pheromone Update

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

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

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

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

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. 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

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

where $\varphi \in (0,1]$ is the pheromone decay coefficient; $\tau_0$ is the initial pheromone value.

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

*B. ACO-based Image Edge Detection*

Image edge detection can be thought of as a problem of identifying the pixels in an image that correspond to edges. A w × h two-dimensional digital image can be represented as a two-dimensional matrix with the image pixels as its elements (Fig. 2).

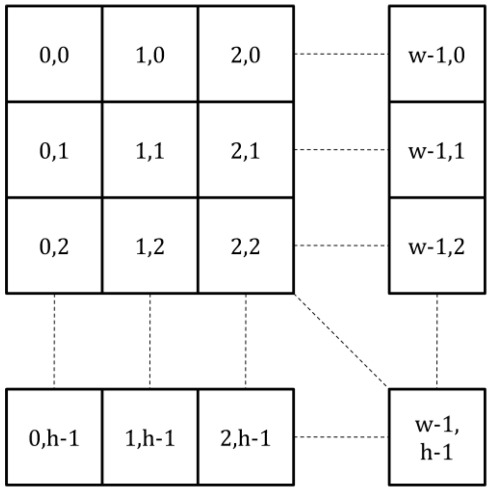

Fig. 2 Matrix representation of an image

The graph is defined as follows. The components of the graph are the pixels of the image. The connections of the graph connect adjacent components or pixels together. The construction graph representation of an image is shown in Fig. 3. An 8-connectivity pixel configuration (Fig. 4) is used: a pixel is connected to every pixel that touches one of its edges or corners. Ants traverse the graph by moving from one pixel to another, through their connections. An ant cannot move to a pixel if it is not connected to the pixel where the ant is currently located. This means that an ant can move only to an adjacent pixel.

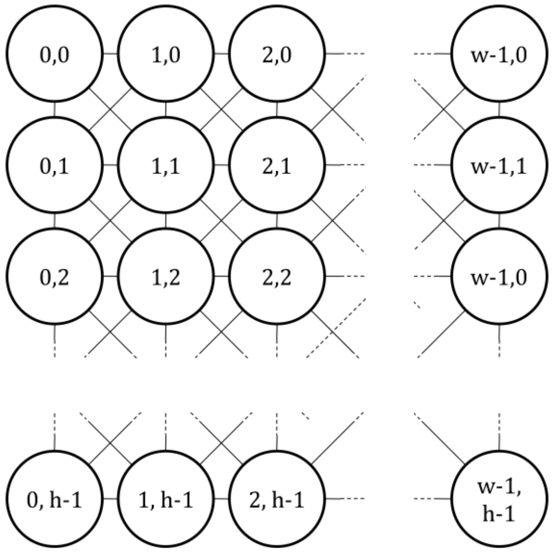

Fig. 3 Graph representation of an image

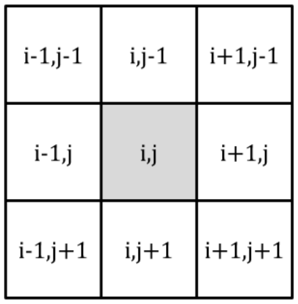

Fig. 4 8-connectivity configuration for pixel $(i, j)$

Artificial ants are distributed over the image and move from one pixel to another. The movement of the ants is steered by the local variation of the pixel intensity values. The goal of the ants' movement is to construct a final pheromone matrix that reflects the edge information. Each element in the pheromone matrix corresponds to a pixel in the image and indicates whether a pixel is an edge or not.

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.

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 $\tau_{init}$, 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.

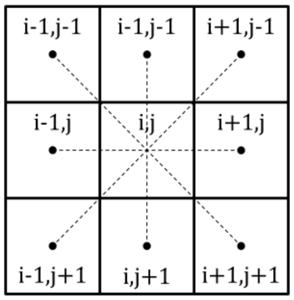

Fig. 5 A local configuration for computing the intensity variation at $(i, j)$

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

$$\eta_{i.j} = \frac{V_c(I_{i,j})}{V_{max}} \tag{7}$$

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. 5) around the pixel $(i, j)$. It depends on the variation of the intensity values on the local group, and is given by

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

$V_{max}$ is the maximum intensity variation in the whole image and serves as a normalization factor.

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

$$p^{(n)}_{(i_0,j_0),(i,j)} = \frac{\left(\tau^{(n-1)}_{i,j}\right)^{\alpha} (\eta_{i,j})^{\beta}}{\sum_{(i,j)\in\Omega_{(i_0,j_0)}} \left(\tau^{(n-1)}_{i,j}\right)^{\alpha} (\eta_{i,j})^{\beta}} \tag{9}$$

where $\tau^{(n-1)}_{i.j}$ 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 $\beta$ control the influence of the pheromone and the 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, $\tau^{(n)}_{i,j}$, is updated based on the equation for ACS local pheromone update:

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

where $\varphi \in (0,1]$ is the pheromone decay coefficient; $\tau_{init}$ 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. 4). 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.

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

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

where $\Delta\tau^{(k)}_{i,j}$ is the amount of pheromone deposited by the $k^{th}$

ant on pixel $(i, j)$. The deposited amount of pheromone $\Delta\tau_{i,j}^{(k)}$ is equal to the average of the heuristic information associated with the pixels that belong to the tour of the $k^{th}$ ant if pixel $(i, j)$ was visited by the $k^{th}$ ant in its current tour; 0 otherwise. Its reciprocal 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. 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.

The difference lies in the selection of the tours to be used in the update. 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 complete possible 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 collective interaction 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 one ant undergo a global pheromone update.

### 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 [10], 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. 6 ACO-based image edge detection

Fig. 6 shows a pseudocode of the proposed method.

## IV. Experimental Results

Experiments were conducted using canonical test images to observe the effect of the parameter $q_0$ on the result and to compare the edges produced using AS with those produced using ACS.

Fig. 7 shows four test images: *Lena*, *Mandril*, *Peppers*, and *Pirate*. All the canonical test images presented in this chapter have a size of 256 × 256 pixels.

The parameters of the algorithm are:

- $\tau_{init}$: initial pheromone value
- N: number of iterations
- L: number of construction steps
- K: number of ants
- $q_0$: parameter for controlling the degree of exploration of the ants
- $\alpha$: parameter for controlling influence of pheromone trail (fixed to 1 for ACS)
- $\beta$: parameter for controlling influence of heuristic information
- $\varphi$: pheromone decay coefficient
- $\rho$: pheromone evaporation coefficient

In the experiments, the fixed parameters were assigned values that have been found to produce good results.

Their values used in the experiments are:

- $\tau_{init} = 0.1$
- $N = 10$
- $L = 40$
- $K = 512$ (256 × 256 image)
- $q_0$ (varies)
- $\alpha = 1$ (ACS)
- $\beta = 1$
- $\varphi = 0.05$
- $\rho = 0.1$

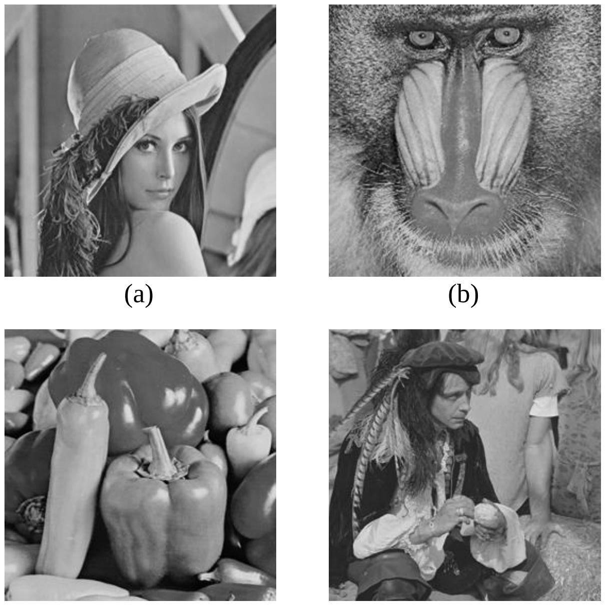
(a) (b)
(c) (d)

Fig. 7 (a) *Lena*, (b) *Mandril*, (c) *Peppers*, (d) *Pirate*

### *A. Effect of Parameter $q_0$*

Fig. 8-11 show the extracted edges of the test images *Lena*, *Mandril*, *Peppers*, and *Pirate*, respectively, at different values of $q_0$. Increasing the value of $q_0$ results to smoother edges. However, it is not good to set $q_0$ to a very high value because it causes some significant features to be missed, as clearly shown when $q_0$ is 1. Evidently, it is also not good to set $q_0$ to 0. To take advantage of the ACS decision rule, $q_0$ must have a value between, but not equal to, 0 and 1. At 0, the edges are barely distinguishable. At 1, the random exploration of the ants is completely removed and important features of the image are missed. The range of good values for $q_0$ depends on the nature of the image. In general, higher values of $q_0$ are suitable for images that contain less details while lower values are suitable for those that contain more details.

### *B. ACS Edges vs. ACS Edges*

A version of the algorithm that uses ant system was implemented and tested using the same test images. The results produced with AS and ACS were compared, at different values of the ACS parameter $q_0$.

Fig. 12-15 show that ACS can produce better results. For *Lena*, *Mandril*, and *Peppers*, a significant improvement is already visible at $q_0 = 0.4$. For *Pirate*, although the quality of the ACS edges is not as good, the edges extracted with ACS are more defined than those with AS. Even at relatively lower values of $q_0$, say 0.2, the edges produced by ACS are, in general, more defined.

## V. Conclusion

An ACO-based image edge detection algorithm that takes advantage of the improvements introduced in ACS has been successfully developed and tested. Experimental results show the feasibility of the approach in identifying edges in an image. With suitable parameter values, the algorithm was able to successfully identify edges in the canonical test images. It must be noted that the appropriate parameter values depend on the nature of the image, and thus, may vary per application.

As a continuation of this research, it is recommended to further examine how the quality of the extracted edges is affected by the parameter values and the functions for obtaining the heuristic information, for quantifying the quality of a solution, and for computing how much pheromone to deposit. In a study on a simplified ACO algorithm [11], it was shown that the basic properties of ACO are critical to the success of the algorithm, especially when solving more complex problems.

In recent studies, techniques that could enhance the performance of ACS have been explored. In [12], ants are assigned different pheromone sensitivity levels, which makes some ants more sensitive to pheromone than the others. In [13], multiple ant colonies with new communication strategies were employed. The proposed ACS method for edge detection could be extended and possibly be improved by making use of such techniques.

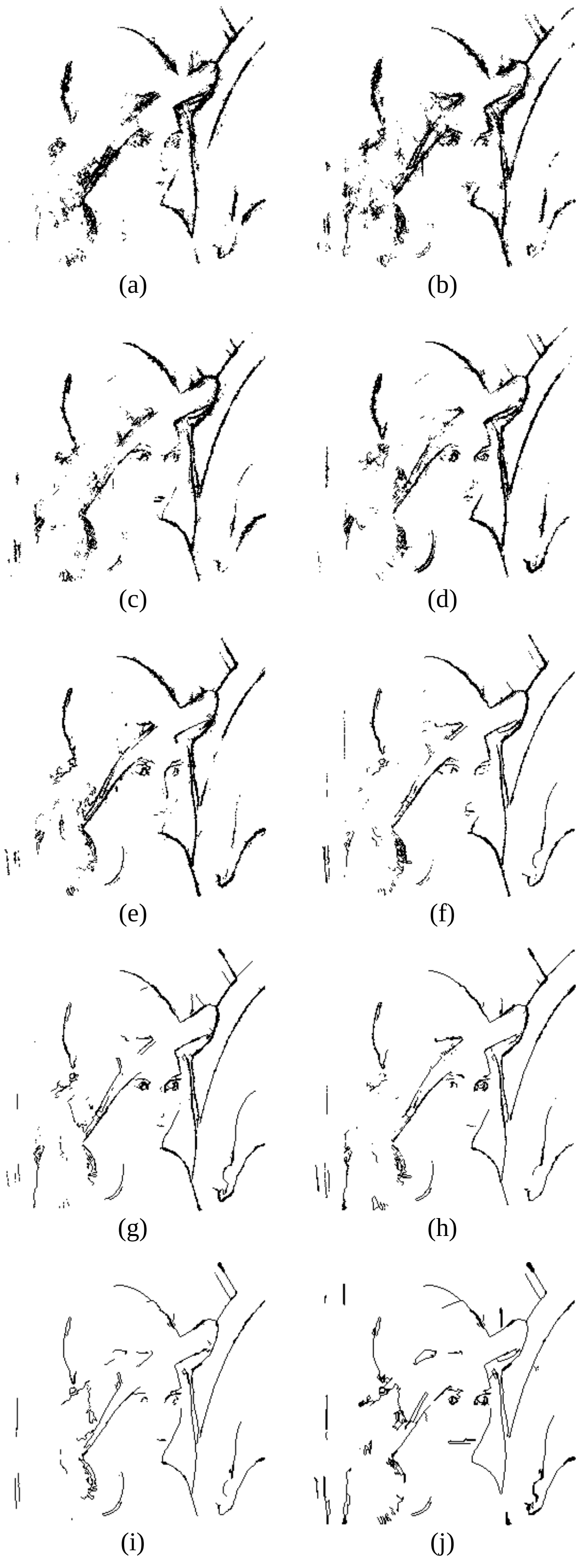

Fig. 8 Edges for *Lena* at different values of $q_0$: (a) 0.0, (b) 0.1, (c) 0.2, (d) 0.3, (e) 0.4, (f) 0.6, (g) 0.7, (h) 0.8, (i) 0.9, (j) 1.0

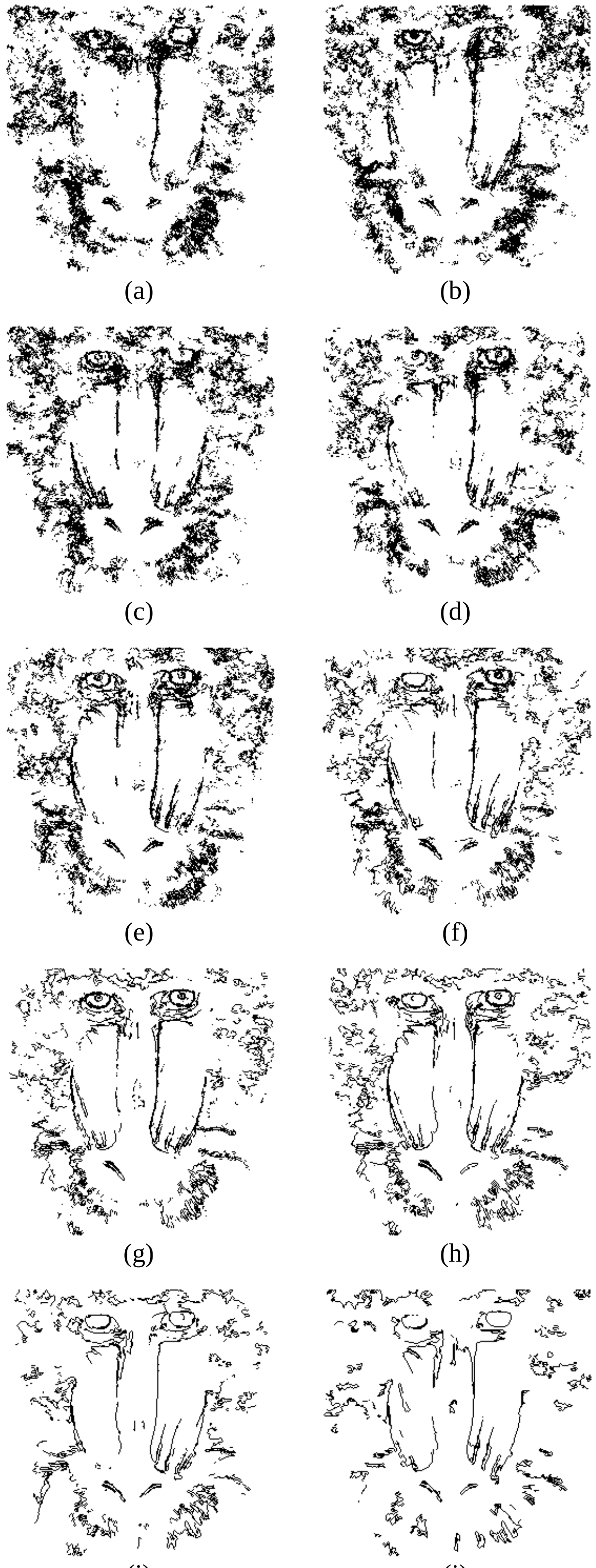

Fig. 9 Edges for *Mandril* at different values of $q_0$:
(a) 0.0, (b) 0.1, (c) 0.2, (d) 0.3, (e) 0.4,
(f) 0.6, (g) 0.7, (h) 0.8, (i) 0.9, (j) 1.0

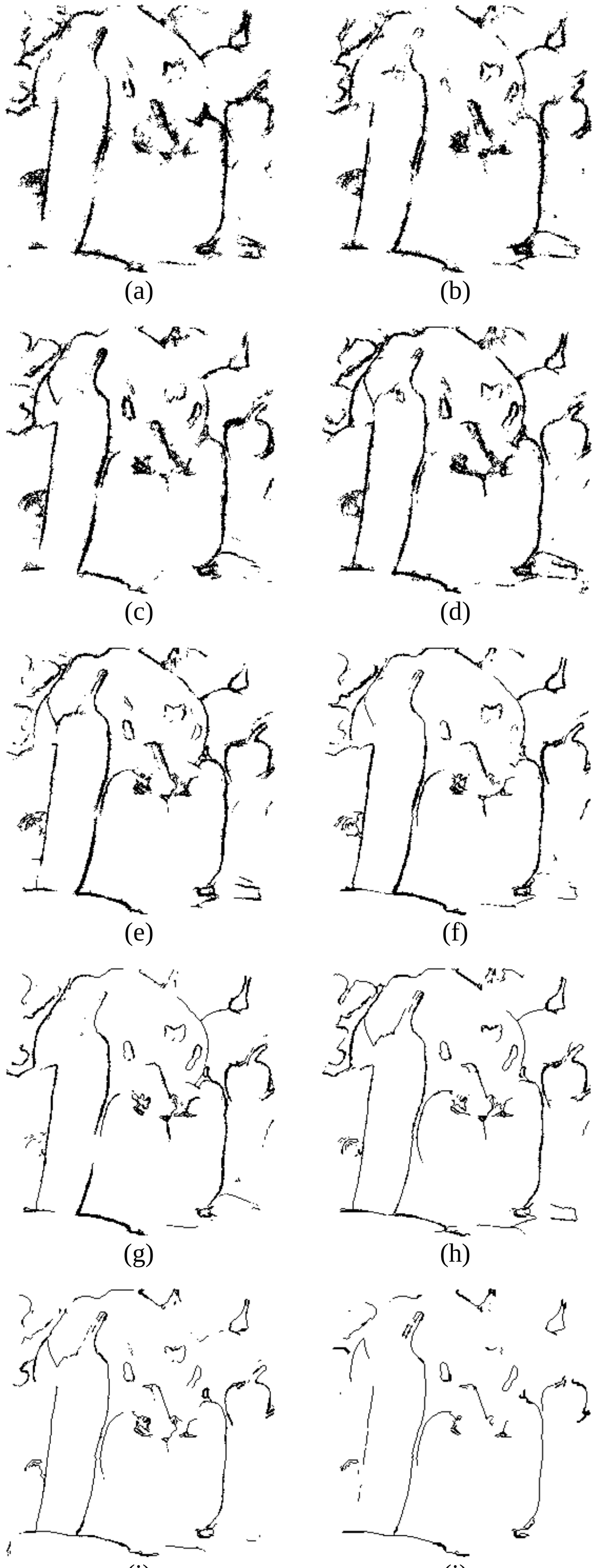

Fig. 10 Edges for *Peppers* at different values of $q_0$:
(a) 0.0, (b) 0.1, (c) 0.2, (d) 0.3, (e) 0.4,
(f) 0.6, (g) 0.7, (h) 0.8, (i) 0.9, (j) 1.0

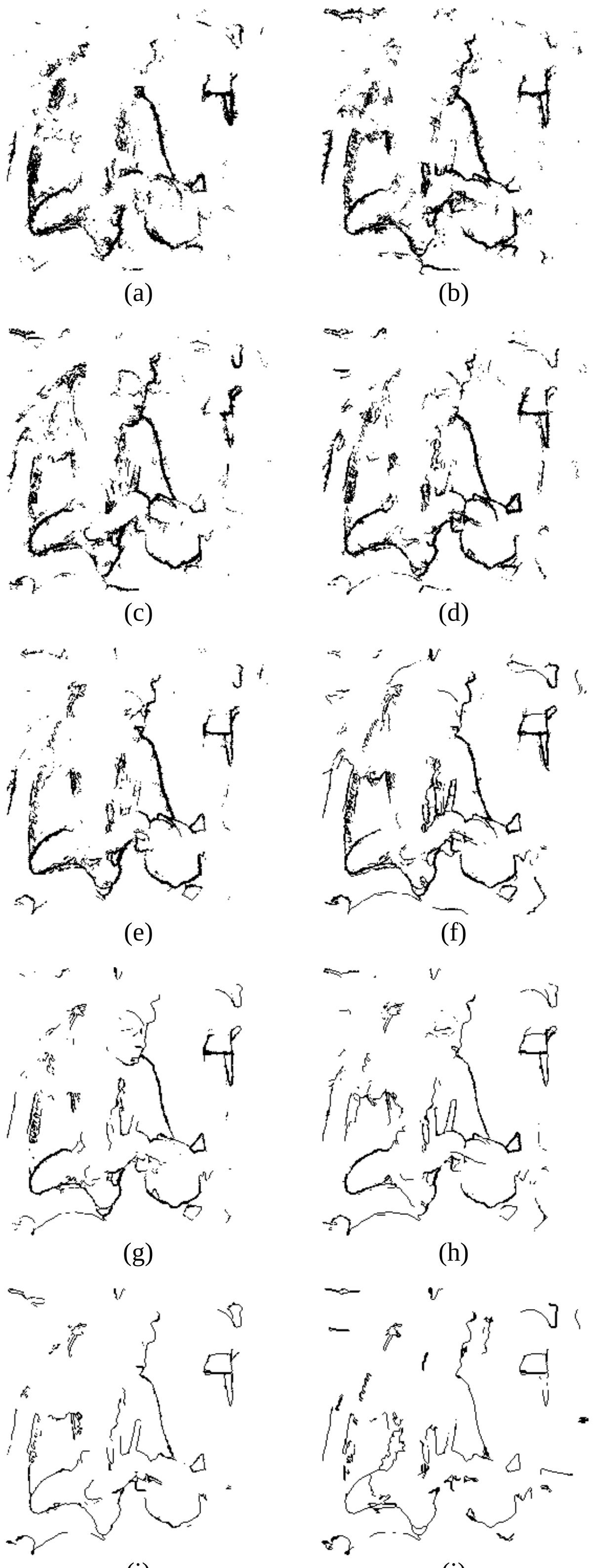

Fig. 11 Edges for *Pirate* at different values of $q_0$:
(a) 0.0, (b) 0.1, (c) 0.2, (d) 0.3, (e) 0.4,
(f) 0.6, (g) 0.7, (h) 0.8, (i) 0.9, (f) 1.0

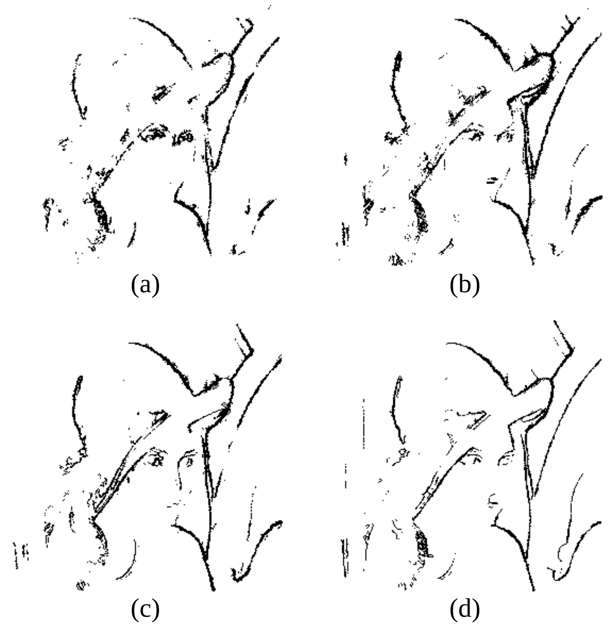

Fig. 12 Comparison between AS and ACS edges at different values of $q_0$ for *Lena*: (a) AS, (b) ACS 0.2, (c) ACS 0.4, (d) ACS 0.6

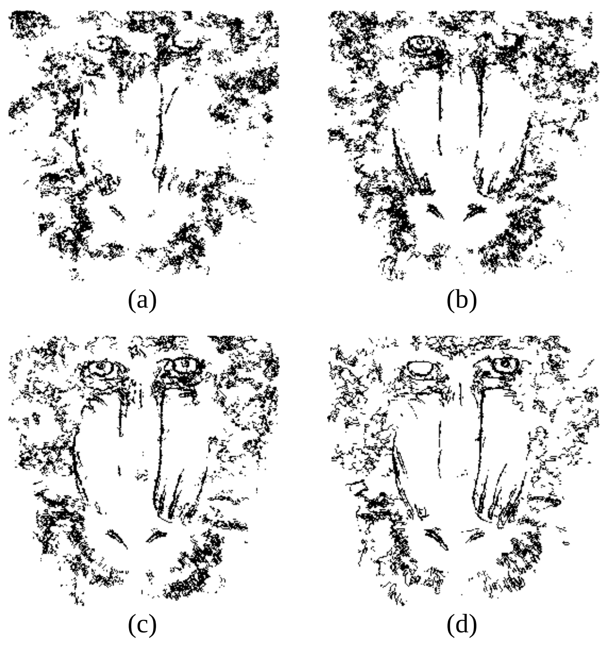

Fig. 13 Comparison between AS and ACS edges at different values of $q_0$ for *Mandril*: (a) AS, (b) ACS 0.2, (c) ACS 0.4, (d) ACS 0.6

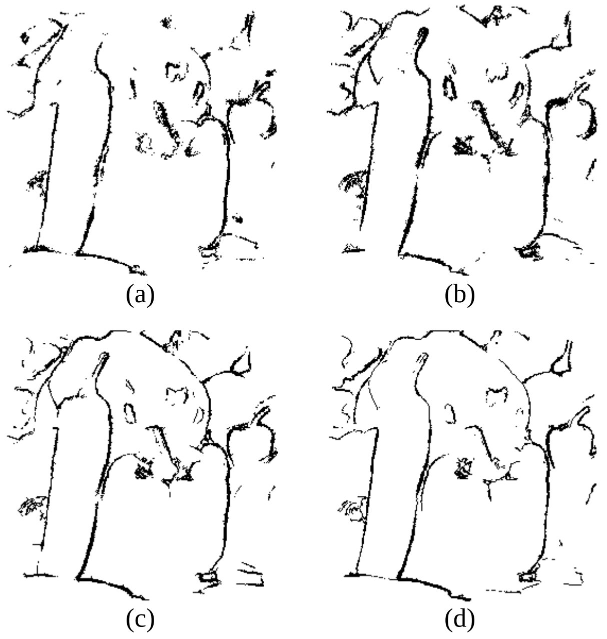

Fig. 14 Comparison between AS and ACS edges at different values of $q_0$ for *Peppers*: (a) AS, (b) ACS 0.2, (c) ACS 0.4, (d) ACS 0.6

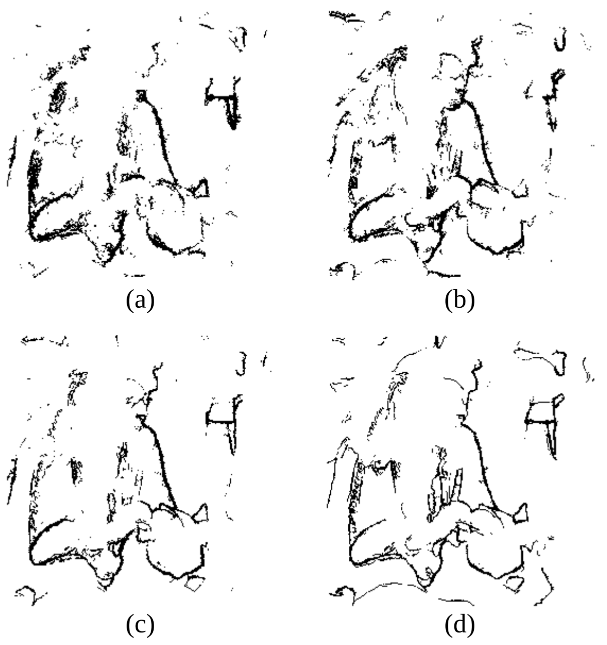

Fig. 15 Comparison between AS and ACS edges at different values of $q_0$ for *Pirate*: (a) AS, (b) ACS 0.2, (c) ACS 0.4, (d) ACS 0.6

## REFERENCES

[1] M. Dorigo and T. Stützle, *Ant Colony Optimization*, Cambridge: MIT Press, 2004.

[2] M. Dorigo. (2007) "Ant Colony Optimization," *Scholarpedia*, 2(3):1461. [Online]. Available: http://www.scholarpedia.org/article/Ant_colony_optimization.

[3] 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, pp. 29-41, 1996.

[4] M. Dorigo and L. M. Gambardella, "Ant Colony System: A Cooperative Learning Approach to the Traveling Salesman Problem," *IEEE Transactions on Evolutionary Computation*, vol. 1, pp. 53-66, 1997.

[5] A. Rezaee, "Extracting Edge of Images with Ant Colony," *Journal of Electrical Engineering*, vol. 59, no.1, pp. 57-59, 2008.

[6] J. Tian, W. Yu, and S. Xie, "An Ant Colony Optimization Algorithm for Image Edge Detection," *IEEE Congress on Evolutionary Computation*, 2008.

[7] H. Nezamabadi-pour, S. Saryazdi, and E. Rashedi, "Edge Detection Using Ant Algorithms," *Soft Computing*, vol. 10, pp. 623-628, 2006.

[8] X. Zhuang and N. E. Mastorakis, "Edge Detection Based on the Collective Intelligence of Artificial Swarms," *Proceedings of the 4th WSEAS International Conference on Electronic, Signal Processing, and Control*, 2005.

[9] 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.

[10] N. Otsu, "A Threshold Selection Method from Gray-level Histograms," *IEEE Transactions on Systems, Man and Cybernetics*, vol. 9, no. 1, pp. 62-66, 1979.

[11] M. Dorigo and T. Stützle, "An Experimental Study of the Simple Ant Colony Optimization Algorithm," *Proceedings of the WSES International Conference on Evolutionary Computation*, 2001.

[12] C. Chira, D. Dumitrescu, and C. Pintea, "Sensitive Ant Model for Combinatorial Optimization," *Proceedings of the 12th WSEAS International Conference on Computers*, 2008.

[13] I. Ellabib and O. Basir, "A Preliminary Study for Multiple Ant Colony System with New Communication Strategies," *Proceedings of the 9th WSEAS International Conference on Communications*, 2005.