Edge detection and Prioritization in Blurred images using Ant Colony Optimization Technique

Shweta Agarwal
M.tech scholar @ JIET, Jodhpur

Abstract

This paper defines edge detection and prioritization in blurred images. Edge Detection has been center of concern in image processing, as this help to distinguish object from the system and draw boundaries. A large number of powerful and effective techniques exist for edge detection in digital images. But sometimes our image is not clear, it is blurred. All edge detection algorithms are not able to process such blurred images. So by using ant colony optimization technique it is possible to detect and prioritize edges in blurred images. Our algorithm does not consider image de-blurring hence eliminating any chances of data loss. We know that a blur image will produce multiple edges in an area of concern, few among these edges will be non-prominent and less useful, and few others will be more prominent, strong and useful.

1. Introduction

Edge detection is a fundamental tool in image processing and computer vision in the areas of feature detection and feature extraction. The main aim of edge detection is to identifying points in a digital image at which the image brightness changes sharply or more formally. Image edge detection deals with extracting edges in an image by identifying pixels where the intensity variation is high. It is a fundamental tool used in most image processing applications to obtain information from the frames as a precursor step to feature extraction and object segmentation. This process detects outlines of an object and boundaries between objects and the background in the image. The edge is the set of the pixel, whose surrounding gray is rapidly changing. The internal characteristics of the edge-dividing area are the same, while different areas have different characteristics. The edge is the basic characteristics of the image. There is a lot of information of the image in the edge. Edge detection is to extract the characteristics of discrete parts by the difference in the image characteristics of the object, and then to determine the image area according to the closed edge.

2. Edge models

The edges obtained from natural images are usually not at all ideal step edges. Instead they are normally affected by one or several of the following effects:

- Focal blur caused by a finite depth-of-field and finite point spread function.
- Penumbral blur caused by shadows created by light sources of non-zero radius.
- shading at a smooth object

A number of researchers have used a Gaussian smoothed step edge (an error function) as the simplest extension of the ideal step edge model for modeling the effects of edge blur in practical applications [3][4]. Thus, a one-dimensional image f which has exactly one edge placed at $x = 0$ may be modeled as:

$$\mathrm{f(x)} = (\mathrm{Ir} - \mathrm{Il})/2(\mathrm{erf}\left(\frac{\mathrm{x}}{\sqrt{2\sigma}}\right) + 1) + \mathrm{Il}$$

At the left side of the edge, the intensity is

$$Il = \lim_{x\to-\infty} f(x),$$

The edge it is

$$Ir = \lim_{x\to\infty} f(x).$$

The scale parameter σ is called the blur scale of the edge.

3. Approaches of edge detection

There are many methods for edge detection, but most of them can be grouped into two categories, search-based and zero-crossing based. The search-based methods detect edges by first computing a measure of edge strength, usually a first-order derivative expression such as the gradient magnitude, and then searching for local directional maxima of the gradient magnitude using a computed estimate of the local orientation of the edge, usually the gradient direction. The zero-crossing based methods search for zero crossings in a second-order derivative expression computed from the image in order to find edges, usually the zero-crossings of the Laplacian or the zero-crossings of a non-linear differential expression. As a pre-processing step to edge detection, a smoothing stage, typically Gaussian smoothing, is almost always applied.

- Edge detection based on gradient operator.

The edge is the place where image gray value is changing rapidly, so the method based on the derivation of the gradient operator is most widely used. The classical gradient operators are Sobel operator [1], Prewitt operator [2], Roberts operator, Laplacian operator.

- Edge detection based on the optimum operator.

The gradient of the image edge is the maximum value, that is, the inflection point of the gray image is the edge. From the mathematical point of view, inflection point of the second derivative of the function is 0. Detecting this point, whose second derivative is 0 is a way of edge detection, for example, Marr-Hildreth operator[3], Canny operator[4,5].

- Multiscale edge detection.

Wavelet transform is particularly suitable for signal mutation detection and edge detection. Rosenfeld [6] suggested a combined consideration on the edge detected by multiple dimensions operator; Marr advocated applying multiple scales of different operators, and put forward some combination rules.

- Edge detection based on ant colony optimization:

(ACO) is a nature-inspired optimization algorithm [1], [2], motivated by the natural phenomenon that ants deposit pheromone on the ground in order to mark some favourable path

- Some other methods.

The adaptive smooth filter method. The iterative computation of the smoothing filtering sharpens the signal edge. And then to detect the edge can get a high positioning accuracy. There are also methods based on integral transform and based on tensor.

4. Ant Colony Optimization technique

Ant colony optimization (ACO) is a class of optimization algorithms modeled based on the actions of an ant colony. ACO methods are useful in problems that need to find paths to goals. Artificial 'ants'—simulation agents—locate optimal solutions by moving through a parameter space representing all possible solutions. Real ants lay down pheromones directing each other to resources while exploring their environment. The simulated 'ants' similarly record their positions and the quality of their solutions, so that in later simulation iterations more ants locate better solutions.

Ant colony optimization (ACO) is paradigm inspired by the intelligence of real ants, for finding solutions to combinatorial optimization problems. More specifically, ACO is inspired by food foraging behaviors exhibited by ant societies. Ants as individuals are unsophisticated living beings. This algorithm is a member of ant colony algorithms family, in swarm intelligence methods, and it constitutes some metaheuristic optimizations. The first algorithm was aiming to search for an optimal path in a graph, based on the behaviour of ants seeking a path between their colony and a source of food.

The ants communicate using a chemical substance called pheromone. As an ant travels, it deposits a constant amount of pheromone that other ants can follow. Each ant initially moves in a somewhat random fashion, but when an ant encounters a pheromone trail, it must decide whether to follow it. If it follows the trail, the ant's own pheromone reinforces the existing trail, and the increase in pheromone increases the probability of the next ant selecting the path. Therefore, the more the ants travel on a path, the more attractive the path becomes for subsequent ants. Additionally, an ant using a short route to a food source will return to the nest sooner and, therefore, mark its path twice, before other ants return. This directly influences the selection probability for the next ant leaving the nest. Over time, as more ants are able to complete the shorter route, pheromone accumulates faster on shorter paths and the longer paths are less reinforced and ultimately abandoned. When looking for food, ants tend to follow trails of pheromones whose concentration is higher. These trails are created by individuals looking for food, to guide others toward the same sources of food. The concentration of pheromone is stronger in highly visited places. Because of the distance travelled by ants to reach food sources and return to the nest, trails near the nest develop a higher concentration of pheromone. The phenomenon give rise to one of the most important type of optimization algorithms named as ACO [13].

The ACO mechanism has the following desirable properties:

1. ACO is versatile in that it can be applied to similar versions of the same problem. An example of a similar version might be the traveling salesman problem, which can be directly extended to the asymmetric traveling salesman problem.
2. ACO is robust, and can be applied with only minimal changes to other combinational optimization problems, such as the quadratic assignment problem and the job shop scheduling problem.
3. ACO is a population-based approach, and allows positive feedback exploitation to be adopted as a search mechanism.

4.1. ACO based algorithm

An image matrix model is used in computer science as a discrete model. The use of ACO in image processing and graphical applications has received some attention over the past few years [Popovici, A. and Popovici, D., 2002]. As stated previously, ACO techniques appear as a natural tool for image processing due to their local nature and simple parallel computing implementation. In this section, one main algorithm has been presented and investigates its variation and effectiveness for processing of gray level images. The algorithm will correspond to edge detection for gray scale images. The application of the ACO techniques to real images will be presented, which together with the results will show the performance characteristics and comparison. The ACO techniques algorithm for k gray levels of digital images is on the basis of a two-dimensional ACO techniques (I, N, V, f) with V = {0, 1, 2, …, k-1}, where k is a number of states, N is the type of neighborhood (e.g. n is total number of neighbors varies from 0 to 8), while the local transition function defined as f: Vn into V.

Given a configuration c of the pixels in the ACO at a certain time t, the configuration c′ at time t + 1 for each pixel x can be calculated as c′(x) = f(c (x1. . . xn)).

In such a two dimensional ACO, specific neighborhoods can be defined. For example, the so-called Von Neumann neighborhood (4-neighborhood) and Moore neighborhood (8-neighborhood) for a pixel x i, j is defined as:

x i−1, j, x i, j−1, x i, j+1, x i+1,j,

and

x i−1, j , x i, j−1, x i, j+1, x i+1, j, x i-1, j+1, x i+1,j -1, x i+1,j+1, x i-1,j-1, respectively.

4.2. Gray level edge detection

In digital image processing, each image is quantized into pixels. With gray-scale images, each pixel indicates the level of brightness of the image in a particular spot: 0 represents black and 255 represent white. An edge is an abrupt change in the brightness (gray scale level) of the pixels. Edge information for a particular pixel is obtained by exploring the brightness of pixels in the neighborhood of that pixel. If all of the pixels in the neighborhood have almost the same brightness, then there is probably no edge at that point. However, if some of the neighbors are much brighter than the others, then there is probably an edge at that point. Measuring the relative brightness of pixels in a neighborhood is mathematically similar to calculating the derivative of brightness. Brightness values are discrete, not continuous, so the derivative function has been estimated. The objective of the edge detection techniques using ACO techniques is to enhance the magnitude of the local differences in gray level values between regions of the images. Over regions which are different, changes must be made to enhance the edges. Here it has been considered grid lattices of 256 × 256 pixels where each pixel has 256 states values ranging 0-255 using the grayscale method. Each pixel is scanned and checked for being a part of active edge using Moore Neighborhood techniques. A memory matrix representing image is maintained. If a pixel is a part of edge then corresponding matrix pixel value is marked else 0. Finally the edged image is constructed using the original input image and image-edge value matrix.

5. Proposed work

Image processing has been under focus since long, lot of study and work has been done in this field, a variety of techniques and algorithms have emerged in the process. A large number of powerful and effective techniques exist for edge detection in digital images. But sometimes our image in not clear, it's blurred. All edge detection algorithms are not able to process such blurred images. For such images De-blurring Edge Detection algorithms come into picture. These algorithms use Gaussian de-blurring and then implement edge detection. These usually results in loss of temporal data in blurred images, which may lead to loss of necessary information, and hence degraded results.

I propose a new approach towards edge detection in blurred images. This algorithm does not consider image de-blurring hence eliminating any chances of data loss. We know that a blur image will produce multiple edges in an area of concern, few among these edges will be non-prominent and less useful, and few others will be more prominent, strong and useful. We simply do not remove these non-prominent edges, we detect each and every possible edge in our area of concern, calculate its strength, importance and assign different strength values of edge pixels.

I will be converting input images to Grayscale images using generalized formula:-

Grayscale Value= .299 * red + .587 * green + .114 * blue

Strength value of a pixel is calculated as maximum variation in Grayscale values between his own value and neighborhood pixel values. Below Tables show how this is done.

"Table 1. Strength for radius=1".

50	50	100
50	120	100
50	120	100

"Table 2. Strength for radius=2".

0	50	50	100	100
0	50	50	100	100
0	50	120	100	100
0	50	120	100	100
0	50	120	100	100

Table 1.shows strength calculation for radius of 1, out of two possible differences of 120-50=70, 120-100=20, we consider maximum difference of 70.

Table 2.shows strength calculation for radius of 2, out of three possible differences of 120-0=120, 120-50=70, 120-100=20,140-120=20 we consider maximum difference of 120.

Values indicated grayscale values on a scale of 0-255, Center pixel is current pixel for which strength is being calculated. After detected edges assign different color values to these edges according to their strength value. That means edges of different strength are given different color values. In the end these colored edges are drawn/highlighted, this way we can visually correlate and distinguish multiples edges with different strength and importance in our area of concern.

5.1. Algorithm for Proposed Technique

Step 1:- Detect Image Dimensions and Initialize Ant bots depending upon Multiprocessing Capabilities of hardware.

Step 2:- Deploy all SM-Ant-Bots, to cover all pixels in image.

Step 3:- Using Thickness value, An SM-Ant-Bot will check a pixel for variations in color values in neighborhood.

Step 4:- If variation exceeds minimum threshold (10) then SM-Ant-Bot will mark that pixel as Potential Edge Pixel.

Step 5:- Repeat step 3-4 for all pixels i.e. Wait while all pixels are checked.

Step 6:- Deploy all S-Ant-Bots, to cover all edge pixels.

Step 7:- Using Radius Value, An S-Ant-Bot will calculate the maximum variation that lies between pixel and its neighborhood.

Step 8:- Repeat step 7 for all pixels i.e. wait while all edge pixels are processed.

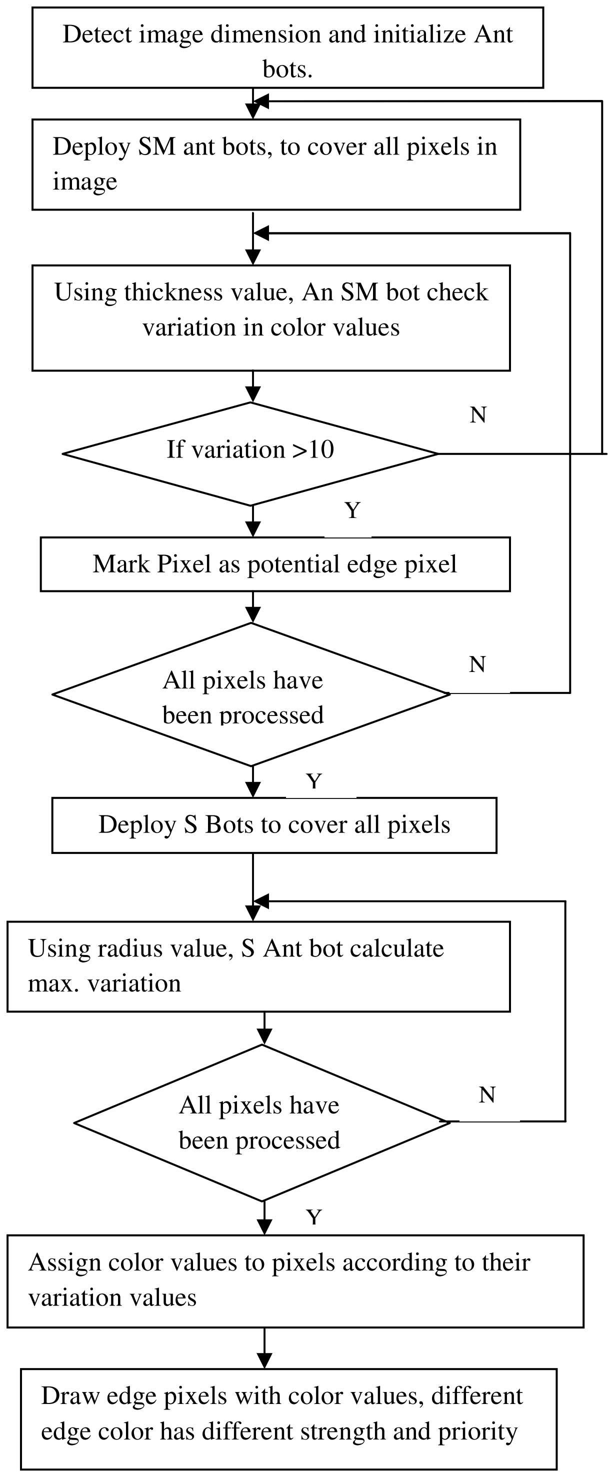
Step 9:- Assign Color values to edge pixels according to their variation value.

Step 10:- Draw edge pixels with their color values, different color values designate different edge strength and hence difference priority.

SM-Ant-Bot: - Search and Mark Ant Bot, is used to find and mark potential edge pixels. Marking a pixel is similar to leaving pheromone signals.

S-Ant-Bot:-Strength Ant Bot, is used to check maximum variation of an edge pixel.

5.2. Flow-Chart for Proposed Technique


Detect image dimension and initialize Ant bots.
Deploy SM ant bots, to cover all pixels in image
Using thickness value, An SM bot check variation in color values
If variation >10
N
Y
Mark Pixel as potential edge pixel
All pixels have been processed
N
Y
Deploy S Bots to cover all pixels
Using radius value, S Ant bot calculate max. variation
All pixels have been processed
N
Y
Assign color values to pixels according to their variation values
Draw edge pixels with color values, different edge color has different strength and priority

“Figure 1. Flow chart of proposed technique”

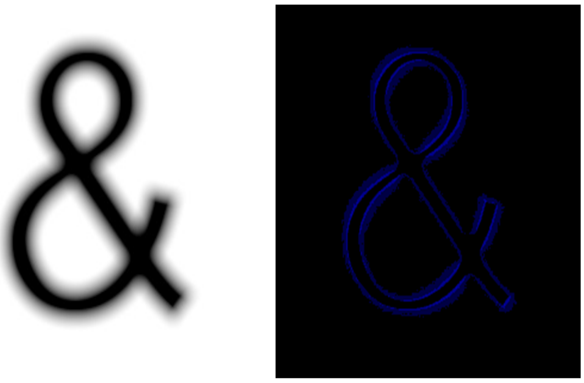
6. Result and Discussion

This section presents and analyses the results produced by above mentioned traditional method and proposed method. It is quite difficult to develop standard performance criteria and methods to evaluate the effectiveness of each edge detector. Locating a real edge pixel becomes extremely crucial. Edge slope angle and its spatial orientation are also important criteria in the evaluation.

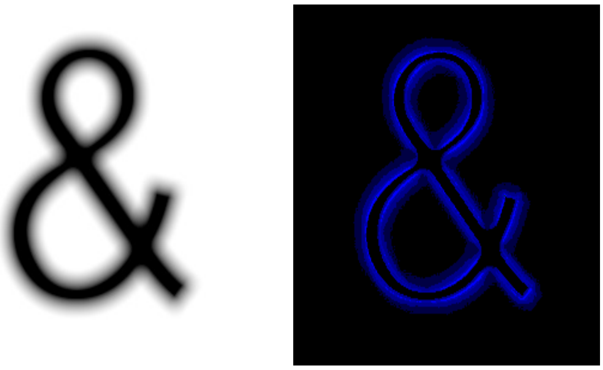
There are five different criteria those are typically used for testing the quality of an edge detector:

- The probability of a false positive (marking something as an edge which isn't an edge)
- The probability of a false negative (failing to mark an edge which actually exists)
- The error in estimating the edge angle.
- The mean square distance of the edge estimate from the true edge.
- The algorithm's tolerance to distorted edges and features such as corners and junctions.

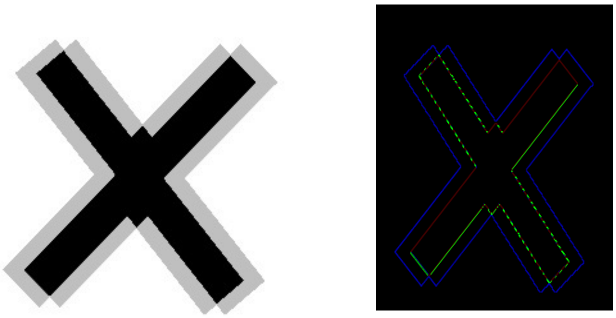
Experiments were carried out over several 256×256 sizes of standard test images. All above said edge detection method like Roberts operator, Sobel operator, Prewitt operator, Canny operator and the proposed method have been implemented on same standard test images using latest software viz, .Net technology.



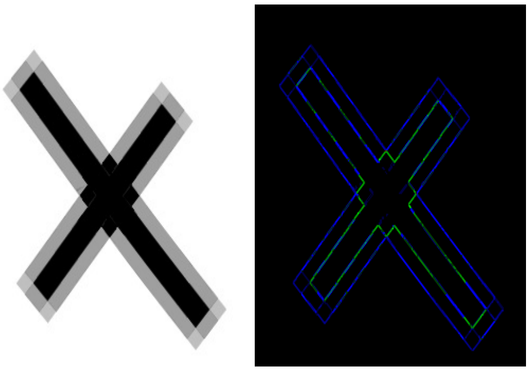
“Figure 2. Edge detection of blur image radius=1”



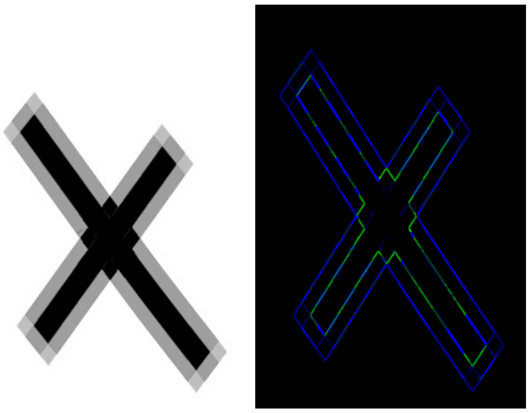
“Figure 3. Edge detection of blur image radius=2”



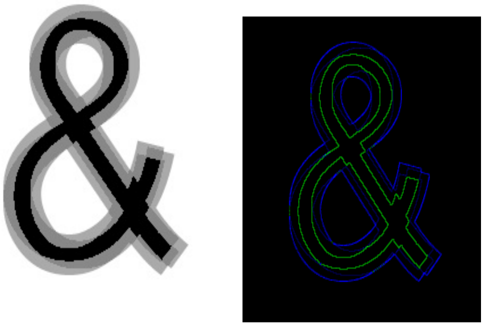
“Figure 4. Edge detection of blur image radius=1”



“Figure 5. Edge detection of blur image radius=1”



“Figure 6. Edge detection of blur image radius=2”



“Figure 7. Edge detection of blur image radius=1”

7. Conclusion

The proposed ACO-based Image Edge detection approach has been successfully implemented. With the evidence of shown images and results, we can state that we have successfully detected edges in normal as well as blurred digital images and prioritized them using different color values, and this prioritization can be visually correlated with the help of differently color marked edges according to their strength and importance.

I have chosen to work with basic shapes having fragment blurs, so that input and results can be visually confirmed by humans. More complex images have also been included in our set of test inputs.

Results appear to be above satisfactory for different sets of inputs, some of them can be visually correlated and some cant. Thus we can say that we have been successful in presenting and implementing a new approach for detecting and prioritizing edges in blurred digital images, without using any de-blurring and loss of data.

This proposed work can be an ideal template and ready reference for a novice researcher in the field of image processing to use a typical ACO algorithm out of the different ACO algorithms for his problem.

In the future this ACO algorithm for edge detection can also be implementing on ACO like hardware and performance issues will also be considered.

8. References

[1] M. Dorigo and S. Thomas, Ant Colony Optimization. Cambridge: MIT Press, 2004.
[2] Jing Tian, Weiyu Yu, and Shengli Xie, “An Ant Colony Optimization Algorithm For Image Edge Detection”, in Proc. of the IEEE International, pp.751-756,2008.
[3] Hossein Nezamabadi-pour · Saeid Saryazdi Esmat Rashedi, “Edge detection using ant algorithms”, in proc. of Springer-Verlag, pp.623-628,2005.
[4] Aleksandar Jevti´c, Joel Quintanilla-Dominguez, M.G. Cortina-Januchs and Diego Andina, “Edge Detection Using Ant Colony Search Algorithm and Multiscale Contrast Enhancement”, in the Proceedings of the 2009 IEEE International Conference on systems, man, and cybernetics, pp.2193-2198,2009.
[5] X. Zhuang, “Edge Feature Extraction in Digital Images with the Ant Colony System”, in proc. of the IEEE international Conference an computational intelligence for Measurement Systems and Applications, pp. 133-136,2004.
[6] Alirezae Rezaee, “Extracting edge of images with ant colony”, in the Journal of ELECTRICAL ENGINEERING,pp. 57–59,2008.
[7] X. Zhuang, “Image Feature Extraction with the perceptual graph based on the ant colony system”,in proc. of the IEEE International conference on systems, man and cybernetics, pp.6354-6359,2004.
[8] X. Zhuang, “Image Feature Extraction with the Perceptual Graph Based on the Ant Colony System”, in the IEEE International Conference on Systems, Man and Cybernetics, pp. 6354-6359,2004.
[9] M. Dorigo and L. M. Gambardella, “Ant colony system: A cooperative learning approach to the traveling salesman problem,” in the proc. of IEEE Trans. On Evolutionary Computation, pp. 53–66, 1997.
[10] M. Dorigo, M. Birattari, and T. Stutzle, “Ant colony optimization,” in the proc. of the IEEE Computational Intelligence Magazine, pp. 28–39,2006.
[11] H.-B. Duan, Ant Colony Algorithms: Theory and Applications. Beijing: Science Press, 2005.

[12] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: Optimization by a colony of cooperating agents,” IEEE Trans. on Systems, Man and Cybernetics, Part B, vol. 26, pp. 29–41, Feb. 1996.

[13] M. Dorigo, M. Birattari, and T. Stutzle, “Ant colony optimization,” IEEE Computational Intelligence Magazine, vol. 1, pp. 28–39, Nov. 2006.

[14] T. Stutzle and H. Holger H, “Max-Min ant system,” Future Generation Computer Systems, vol. 16, pp. 889–914, Jun. 2000.

[15] M. Dorigo, G. D. Caro, and T. Stutzle, “Special Issue on Ant Algorithms, Future Generation Computer Systems”, vol. 16, Jun. 2000.

[16] H. Nezamabadi-Pour, S. Saryazdi, and E. Rashedi, “Edge detection using ant algorithms,” Soft Computing, vol. 10, pp. 623–628, May 2006.

[17] M. Randall and A. Lewis, “A parallel implementation of ant colony optimization,” Journal of Parallel and Distributed Computing, vol. 62, pp. 1421–1432, Sep. 2002.

[18] M. Dorigo, Ant colony optimization web page, http://iridia.ulb.ac.be/mdorigo/ACO/ACO.html.

[19] Anna Veronica Baterina and Carlos Oppus, “Image Edge Detection Using Ant Colony Optimization”, International Journal of circuits, System and Signal Processing, Issue 2 vol. 4, pp. 25-33, 2010.

[20] Bonabeau E., Dorigo M. and Theraulaz G., “Swarm Intelligence, From Natural to Artificial Systems”, Oxford University Press, Oxford, 1999.

[21] Xiaodong Zhuang, Guowei Yang, Hui Zhu , “A Model of Image Feature Extraction Inspired by Ant Swarm System” , in proc. of Fourth International Conference on Natural Computation, pp. 553-557,2008.

[22] Ya-Ping Wong, Victor Chien-Ming Soh, Kar-Weng Ban, Yoon-Teck Bau, “Improved Canny Edges Using Ant Colony Optimization”, in the proc. of IEEE Fifth International Conference on Computer Graphics, Imaging and Visualization, pp. 197-202, 2008.

[23] Jian Zhang and Kun He, Xiuqing Zheng, Jiliu Zhou, “An Ant Colony Optimization Algorithm for Image Edge Detection” in IEEE 2010 International Conference on Artificial Intelligence and Computational Intelligence, pp 215-219,2010.

[24] D. Marr and E. Hildreth, “Theory of edge detection,” Proceedings of the Royal Society of London-Series B, biological Sciences, Vol. 207, pp. 187-217, Feb. 1980.

[25] R.J. Mullen, D. Monekosso, S. Barman, and P. Remagnino, “A review of ant algorithms,” Expert systems with Applications, Vol. 36, pp. 9608-9617, Aug. 2009.

[26] M. Dorigo, V. Maniezzo, and A. Colorni, “Ant system: optimization by a colony of cooperating agents,” IEEE Trans. on Systems, Man and Cybernetics, Part B, vol. 26, pp. 29–41, Feb. 1996.

[27] Hui Zhu, Xiaodong Zhuang, XiangZhong Meng, “Emergent Behavior of Ant Colony System in Digital Images for Acquiring Edge Information” in proc of IEEE Conference pp116-121,2007.

[28] Rafael C. Gonzalez et. al. “ Digital Image processing” third edition, pp. 700-702,2008.

[29] “Global Optimization Algorithms Theory and Application” Thomas Weise second edition Version: 2009-06-26,pg 21-23,40-42.

[30] Lee, K. S. et al. “A new structural optimization method, based on harmony search algorithm” April, 2004,pp 781-798, vol.82,Maryland.

[31] Moscato, P. “On Evolution, Search, Optimization, Genetic Algorithms and Martial Arts: Towards Memetic Algorithms” Cattench concurrent Computation Program (report 826) pp 31-42,1998.

[32] Glover, F. (1990a). "Tabu Search-Part II," ORSA Journal on Computing, 2, 4-32.

[33] Kennedy, J.; Eberhart, R. "Particle Swarm Optimization". Proceedings of IEEE International Conference on Neural Networks. IV. pp. 1942-1948,1995.

[34] Shi, Y.; Eberhart, R.C. "A modified particle swarm optimizer". Proceedings of IEEE International Conference on Evolutionary Computation. pp. 69-73,1998.

[35] A. Colorni, M. Dorigo et V. Maniezzo, “Distributed Optimization by Ant Colonies”, actes de la première conférence , Paris, France, Elsevier Publishing, 134-142,1992.

[36] Om Prakash Verma1 et. al. “A Novel Fuzzy Ant System For Edge Detection”, ”, in Proc. of the 9th IEEE International Conference on Computer and Information Science, pp.228-233,2010.

[37] De-Sian Lu et al. “Edge detection improvement by ant colony optimization”, in Proc. of the 9th IEEE International Conference on Pattern Recognition, pp. 416-425,2007.