数理解析研究所講究録 第2096巻 2018年 90-101

An Experiment with Ant Colony Optimization for Edge Detection in Images

Muhammad Asyran bin Othman[1], Szilárd Zsolt Fazekas[1] and Akihiro Yamamura[1]

Akita University, 1-1 Tegatagakuen-machi, Akita City 010-8502, Japan

Summary. Ant colony optimization (ACO) is a simulation of the natural behavior of ant species; where ants find the shortest path between its nest and food source. Image edge detection is a basic image processing task, where the outlines of the objects in an image are identified, and then extracted. We present the results of an experiment conducted with the ACO algorithm applied to the edge detection problem.

1 Introduction

In this paper we investigate the application of the Ant Colony Optimization algorithm for the image edge detection problem.

Swarm intelligence is a collective behavior of natural organisms, such as birds, fish, and insects. SI algorithms are adaptive and can be used for various purposes. A number of SI algorithms have been proposed and applied to NP-hard problems by many researchers.

Introduced by Dorigo in 1992 ([6]), **ant colony optimization** (ACO) is a well-known swarm intelligence algorithm based on how ants find the shortest path between the nest and the food source. Ants communicate with each other using pheromones. While walking, they secrete a chemical substance called pheromone and leave it on the path for other ants in the colony to follow. The pheromone concentration becomes higher when more ants follow the same path but the amount of pheromone a single ant excretes is the same for any given path. Hence, the pheromone concentration will be higher on shorter paths. All of the ants will gradually follow the path with the higher pheromone concentration (see Fig.1) converging towards the best solution.

Initially developed for tackling NP-hard problems, most prominently the Traveling Salesman Problem (TSP), variants of ACO have been used in a wide variety of contexts, among them numerical problems, even ones for which efficient deterministic algorithms are available, but do not necessarily capture all aspects of the problem.

An interesting example is **edge detection** in images. An edge is a group of pixels that represents the boundaries or the outlines of an object in an image, e.g., the second image in Fig.2. Edge detection is an image analysis process, where a group of pixels that represents the outlines of the image is identified and extracted. Generally, the object and the background can be separated due to the difference between the concentration or intensity of the pixels.

There are several polynomial-time methods already available, such as the Prewitt, Sobel, or Roberts operator based methods or the Canny algorithm, which unifies several approaches and is considered the most accurate method in most edge detection scenarios. These algorithms, however, are computationally expensive for large images as they involve operations for each pixel. To improve processing speed while maintaining accuracy, several variants of

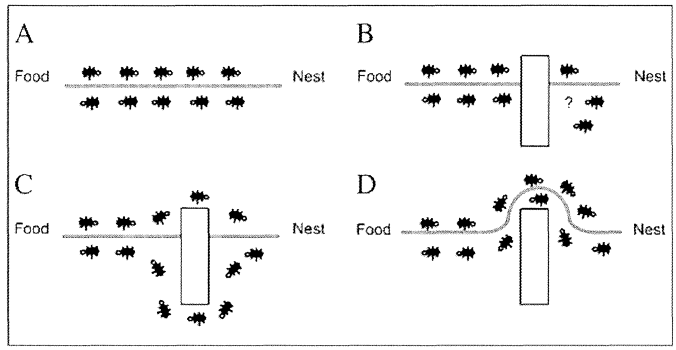

Fig. 1: Ant foraging behavior.
A: Ants in a path between nest and food source. B: An obstacle interrupts the path. C: Ants find two paths around the obstacle. D: Ants walk through the shorter path due to pheromone

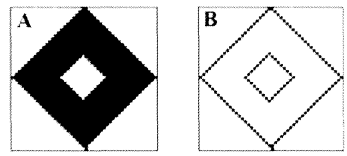

Fig. 2: A: Original image. B: Edges of original image

ACO have been applied to edge detection. The first attempts ([11, 10]) used Dorigo's original Ant Systems. Improved algorithms based on the ACS version of ACO ([5]) have been described in [16] and later in [1]. The behavior of ACO algorithms is highly dependent on the parameters of the system, such as transition probabilities, pheromone evaporation and decay rate, etc. We sought to identify parameter values for which the ACS version of the ACO edge detection algorithm ([16, 1]) produces good results while keeping the running time low. To find those values, we conducted an experiment where we calibrated the parameters through iterated runs on a well-known baseline image and then tested the algorithm on further images to assess the result of the calibration.

First we present a short description of the basic concepts, the ACS version of the ACO algorithm, notions used in edge detection and Otsu's thresholding method, which is used to extract the output from the information provided by the ACO. After the preliminaries we present the results of the experiment.

2 Outline of ACO algorithm used

ACO aims to iteratively find the optimal solution to a given problem. In order to implement ACO, a number K of artificial ants are initialized over the solution space, which consists of $M_1 \times M_2$ nodes, in our case, the pixels of the image. Then, the ants are consecutively moved to the next nodes according to a probabilistic transition matrix $p^{(n)}$. The movement of ants from node i to node j is determined from pheromone information and heuristic information.

Heuristic information is calculated during initialization as the heuristic only depends on the intensity values of the pixels. In our case the heuristic is a normalized intensity variation matrix which has a fixed value. Pheromone information is updated during and after each round in by two separate update methods described later. The final step is extracting the binary level edge information from the final pheromone matrix by Otsu's thresholding technique.

Algorithm 1 Ant Colony Optimization algorithm

Initialization process
    Randomize the position of ants on the nodes of the solution space
    Calculate the heuristic $\eta_{i,j}$ fr each pixel (i, j)
    Assign the initial pheromone information as $\tau^{(0)}$
FOR n from 1 to Number of rounds (Construction Process)
    Move each ant for L steps
    according to probabilistic transition matrix $p^{(n)}$
        Perform the local pheromone updates
    Perform the offline pheromone updates
Decision process: apply Otsu's technique to the final pheromone matrix obtained

3 Configuration of ACO Algorithm for Edge Detection

In our application of the ACO algorithm numerous artificial ants are used. They are moved on a two-dimensional image to find the edge pixels which represent the optimal solution of the problem. Each pixel can be viewed as a node in TSP (Traveling Salesman Problem).

In the edge detection process, ants secrete pheromone on the pixels with high intensity value. The darker the pixel, the higher its intensity value. Hence, we can consider that the darker groups of pixels in an image represent edges of objects. The intensity value of the pixels is represented in Fig.3.

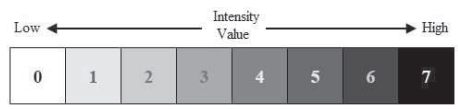

Fig. 3: Intensity values of pixels

The ACO-based approach to edge detection is a three-phased process, where the first phase is Initialization, the second phase is Construction, which also covers the Update, and finally Decision, where it is determined whether the final pheromone matrix obtained throughout the process represents the edge pixels or not. The details of the phases are as follows.

Initialization Process Here, K ants are randomly distributed on $M_1 \times M_2$ pixels of image I. Next, the initial pheromone matrix is assigned as $\tau^{(0)}$. The heuristic information $\eta_{i,j}$ is also calculated during Initialization, i.e., from here on is fixed for every construction step. This

is because the heuristic only depends on the intensity values of the pixels, which does not change during execution. The heuristic information $\eta_{i,j}$ is defined as follows:

$$\eta_{i,j} = \frac{V_c(I_{i,j})}{V_{max}} \tag{1}$$

- $I_{i,j}$: Intensity value of pixel (i, j)
- $V_c(I_{i,j})$: Cumulative intensity variation in the neighborhood of pixel (i, j)
- $V_{max} = max_{pixels(i,j)}\{V_c(i, j)\}$: Maximum local intensity variation in the image (normalization factor)

The value of $\eta_{i,j}$ is high for the pixels that are located in regions with sharp intensity variations, hence, for pixels representing edges in the image. The group of pixels that are adjacent to the pixel (i, j) is known as a *clique*, and is taken to be the 8-neighborhood of the pixel.

The intensity variation between pixel (i, j) and adjacent pixels $V_c(I_{i,j})$, is given by:

$$\begin{aligned} V_c(I_{i,j}) = {} & |I_{i-2,j-2} - I_{i+2,j+2}| + |I_{i+2,j-2} - I_{i-2,j+2}| + |I_{i-1,j-1} - I_{i+1,j+1}| + \\ & |I_{i+1,j-1} - I_{i-1,j+1}| + |I_{i-2,j-1} - I_{i+2,j+1}| + |I_{i+2,j-1} - I_{i-2,j+1}| + \\ & |I_{i-1,j-2} - I_{i+1,j+2}| + |I_{i+1,j-2} - I_{i-1,j+2}| + |I_{i-2,j} - I_{i+2,j}| + \\ & |I_{i,j-2} - I_{i,j+2}| + |I_{i-1,j} - I_{i+1,j}| + |I_{i,j-1} - I_{i,j+1}| \end{aligned} \tag{2}$$

Furthermore, the definition implies that large differences in intensity values between pixels that are located at 180 degrees to each other (relative to pixel (i, j)), gives a high intensity variation.

Construction Process Here, the movement of ants is started and conducted. Firstly, an ant is randomly selected and moved. Afterwards, the next ant is moved until all K ants finished the same round. An ant moves from its current pixel (i, j) to an adjacent pixel (x, y) according to the following probabilistic transition matrix:

$$p^{(n)}_{(i,j)(x,y)} = \frac{(\tau^{(n-1)}_{x,y})^{\alpha}(\eta_{x,y})^{\beta}}{\Sigma_{(x,y)\in\Omega_{(i,j)}}(\tau^{(n-1)}_{x,y})^{\alpha}(\eta_{x,y})^{\beta}} \tag{3}$$

- $\tau^{(n-1)}_{x,y}$: Pheromone information at position (x, y)
- $\eta_{x,y}$: Heuristic information at position (x, y)
- $\Omega_{(i,j)}$: Set of adjacent pixels for the ant which is currently located at pixel (i, j)
- α: Weight of pheromone information
- β: Weight of heuristic information

During the movement of ants, there are two crucial issues that must be considered. The first is determining heuristic information. The second issue is determining the permissible range of the movement of the ants on the image, i.e., the value of $\Omega_{(i,j)}$ in the equation above. It is defined that ants may only move from their current pixel to an adjacent pixel, i.e., they cannot move pixels not directly connected to the one they are currently positioned at. Both through 4-connectivity and 8-connectivity were implemented for adjacency, as shown in Figure 4.

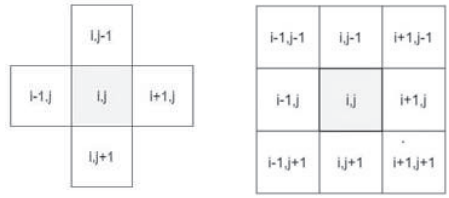

Fig. 4: Various neighborhoods for pixel (i, j)

Update Process Here, pheromone information is updated. There are two separate pheromone information updates. The first one is known as local pheromone update (LPU) and the second one is known as offline pheromone update (OPU).

LPU is performed for each ant k immediately after it has moved in round n using the equation below:

$$\tau_{i,j}^{(n-1)} = \begin{cases} (1-\rho)\cdot\tau_{i,j}^{(n-1)} + \rho\cdot\eta_{i,j} & \text{if } pixel(i,j) \text{ belongs to the best tour} \\ \tau_{i,j}^{(n-1)} & \text{otherwise} \end{cases} \tag{4}$$

- ρ: Pheromone evaporation coefficient $(0 < \rho < 1)$

Meanwhile, OPU is performed immediately after all ants have moved in round n using equation below:

$$\tau_{i,j}^{(n)} = (1-\psi)\cdot\tau_{i,j}^{(n-1)} + \psi\cdot\tau_{i,j}^{(0)} \tag{5}$$

- ψ : Pheromone decay coefficient $(0 < \psi < 1)$
- $\tau_{i,j}^{(0)}$: Initial pheromone information

Pheromone decay aims to diversify the search of edge pixels by ants during the same round. In other words, this is to encourage the ants to select other pixels, explore the image in order to furnish better results of edge pixels. OPU is global, that is, the pheromone level of all pixels is updated, whereas in LPU pheromone level is updated if and only if some ant visited the pixels, so it remains unchanged if the pixel has not yet been visited by any ant.

Decision Process Here, the final pheromone matrix obtained throughout the process above is subjected to one more step which extracts the simplified (bi-level) edge information. A well-known binary decision method is applied to determine the edge pixels. The method is called Otsu's threshold technique [14]. In this process, the grayscale image representing our final pheromone matrix is converted to a binary image representing the detected edges.

In general, Otsu's threshold technique, also known as discriminant analysis method, is used to convert a grayscale image to a binary image. This technique assumes that the image contains two classes of pixels. The method finds the threshold that minimizes the weighted variance within the classes.

The weighted sum of within-class variances for Classes 1 and 2 is given by:

$$\sigma_w^2 = w_1\sigma_1^2 + w_2\sigma_2^2 \tag{6}$$

- w_i: number of pixels in Class i (for $i \in \{1,2\}$)

- m_i: mean of Class i (sum of intensity value / number of pixels)
- σ_i^2: variance of Class i

Calculating within-class variance can be computationally expensive, so we calculate the between-class variance σ_b^2 instead. This is adequate because the between class variance σ_b^2 is maximal exactly when σ_w^2 is minimal, so maximizing the former is equivalent to minimizing the latter. The between-class variance is faster to compute, therefore, our threshold value t is set to

$$\sigma_b^2 = w_1(m_1 - m_2)^2 + w_2(m_2 - m_1)^2 = w_1 w_2(m_1 - m_2)^2 \tag{7}$$

4 Implementation of ACO Algorithm for Edge Detection in Images

4.1 Experiment 1: Parameter calibration for ACO-based edge detection

The experiment was conducted on a computer with Intel Core i5-6600K 3.50GHz CPU having 16 GB RAM and running Windows 10 Pro 64-bit. The algorithm was implemented in MATLAB (version R2011b). The parameters of the implementation were:

IMG: Image data	ALPHA: Value of α in equation 3
PHE: Pheromone information	BETA: Value of β in equation 4
NR: Number of rounds	RHO: Value of ρ in equation 4
NS: Number of ant steps	PSI: Value of ψ in equation 5
NA: Number of ants $\sqrt{IMG_{rows} \times IMG_{cols}}$	

In order to find a good implementation of the ACO algorithm for edge detection, a number of experiments were conducted. The input images used were 256×256 grayscale bitmaps frequently used in image processing studies.

Fig. 5: Input images with resolution 256×256: Cameraman, Lena, Girl, Cat

The results of the ACO edge detection depend on the parameters described above. In the first phase we opted to use the parameter values found in the literature [15, 16], as follows: input image IMG = Cameraman, initial pheromone matrix PHE = 0.0000001, number of ants NA = 256, number of rounds NR = 4, number of steps NS = 900, ALPHA = 1.0, BETA = 1.0, RHO = 0.1, PSI = 0.05 and the result is shown in Fig.6 below.

Based on the images, the program was considered successfully executed, as the white points cover most of the edge pixels of the object. Execution time was 66.94 seconds. The

Fig. 6: (a) Original image and (b) ACO

ALPHA	BETA	RHO	PSI	Exec. time (sec.)
1	1	0.1	0.05	66.97
2	2	0.1	0.05	67.51
0.5	0.5	0.1	0.05	68.96

Table 1: Consideration of ALPHA and BETA

program has been executed five more times to test the variability of the resulting image. Although the resulting images are not shown in this paper, the edges detected were somewhat different, as expected given the random factor inherent to the algorithm, which means the output of ACO-based edge detection methods will not the exact same in each run.

Next, some of the parameter values were changed to examine their effect on the performance of the program. First, we considered different values for parameters ALPHA and BETA. The ALPHA and BETA used in [15, 16] were the starting points and we compared them with three further combinations of the values to determine the optimal ones for use in the next experiment. The other parameters were set as follows: IMG = Cameraman, PHE = 0.0000001, NA = 256, NR = 4, NS = 900 and the results are shown in Figure 7 and Table 1.

Fig. 7: (a) Original image (b) ALPHA=1.0, BETA=1.0 (c) ALPHA=2.0, BETA=2.0 (d) ALPHA=0.5, BETA=0.5

The changes in parameters ALPHA and BETA did not show significant effect in the result, so we adopted ALPHA = 1.0 and BETA = 1.0 as basis parameter value for further experiments, as - not surprisingly - those resulted in the shortest execution time (see table above). Next, a calibration of the parameter RHO was conducted. While using the basis parameter value of ALPHA and BETA obtained in the experiment above, the parameter value of RHO used in the literature was adopted as a baseline and three additional values were tested

ALPHA	BETA	RHO	PSI	Exec. time (sec.)
1	1	0.1	0.05	66.85
1	1	0.5	0.05	67.17
1	1	1	0.05	68.36

Table 2: Consideration of RHO

to determine the value of RHO to be used in the next experiment. The other parameters were set as follows: IMG = Cameraman, PHE = 0.0000001, NA = 256, NR = 4, NS = 900 and the results are shown in Figure 8 and Table 2.

Fig. 8: Experimental results of Cameraman (RHO)

As before, the parameter value RHO to be used as basis in the next experiment, was selected based on the shortest program execution time, hence, from Table 2, RHO = 0.1.

Finally, using the same method as the experiments above, a calibration of parameter value PSI was conducted. The other parameters were set as follows: IMG = Cameraman, PHE = 0.0000001,NA = 256, NR = 4, NS = 900 and the results are shown in Figure 9 and Table 3 below.

Fig. 9: Experimental results of Cameraman (PSI): (a) Original image (b) PSI=0.01 (c) PSI=0.05 (d) PSI=0.1

The parameter value resulting in the shortest program execution time taken was selected, hence, from table above, PSI=0.01.

Based on the experimental results conducted above, the basis parameter values to be used in the next experiments in this paper were obtained: ALPHA = 1.0, BETA = 1.0, RHO = 0.1, PSI = 0.01.

ALPHA	BETA	RHO	PSI	Exec. time (sec.)
1	1	0.1	0.01	66.72
1	1	0.1	0.05	66.98
1	1	0.1	0.1	67.11

Table 3: Consideration of PSI

Next, using the parameter values obtained above, the program was executed using the remaining input images of Lena, Girl, and Cat. The experimental results are shown in Figure 10, Figure 11, and Figure 12 respectively.

Fig. 10: Experimental result of Lena: (a) Original image (b) ACO

Fig. 11: Experimental result of Girl: (a) Original image (b) ACO

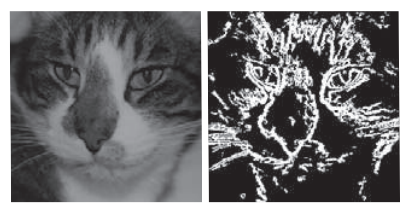

Fig. 12: Experimental result of Cat: (a) Original image (b) ACO

Program execution time for each of the experiments above is shown in the table below.

	ALPHA	BETA	RHO	PSI	Exec. time (sec.)
Lena	1	1	0.1	0.01	67.09
Girl	1	1	0.1	0.01	69.95
Cat	1	1	0.1	0.01	67.11

Table 4: Program execution times

4.2 Experiment 2: Comparison with established edge detection methods

In order to evaluate the performance of the ACO-based edge detection method, we implemented and ran edge detection on the test images using four well established methods (Prewitt, Sobel, Roberts, and Canny). The input images used are same as above and the experimental results are shown in Figure 13, Figure 14, Figure 15 and Figure 16 below.

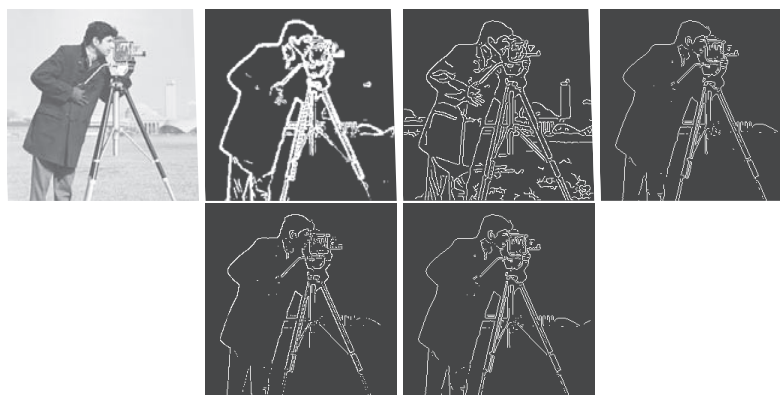

Fig. 13: Experimental result with Cameraman (a) Original image (b) ACO (c) Canny (d) Prewitt (e) Sobel (f) Roberts

5 Conclusion

In Experiment 1, we observed the experimental results of image data Cameraman, as shown in Figure 7. The edge pixels detected when varying the parameters ALPHA and BETA were almost identical. Similarly, the edges detected when varying the parameter RHO were virtually the same. However, decreasing the parameter value PSI resulted in the edges becoming thicker, and vice-versa. Reducing the value of PSI, which represents the pheromone decay coefficient, results in the pheromone decaying at a higher rate, and thus, this encourages the ants to explore the image more and permit the ants to visit other pixels.

Next, we looked the differences between the experimental results of image data Cameraman, as shown in Figure 16, with the other image data Lena, Girl, and Cat, as shown in Figure 17, Figure 18, and Figure 19 respectively. Here, we observed that the image which had the

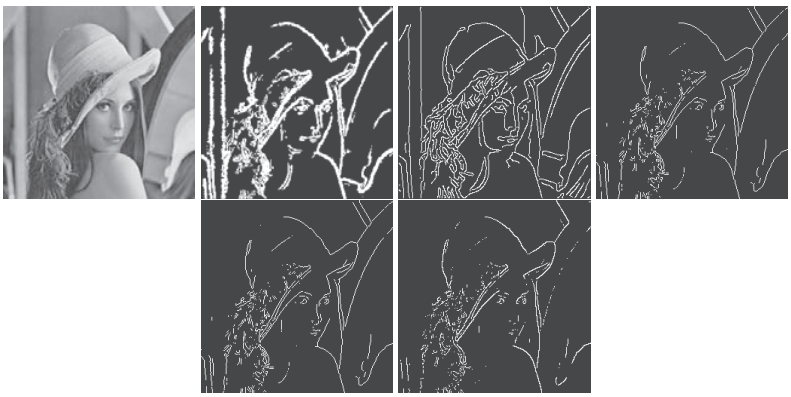

Fig. 14: Experimental result with Lena (a) Original image (b) ACO (c) Canny (d) Prewitt (e) Sobel (f) Roberts

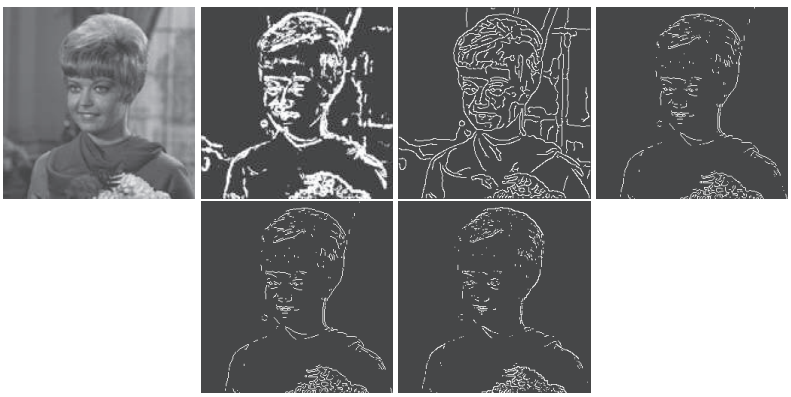

Fig. 15: Experimental result with Girl (a) Original image (b) ACO (c) Canny (d) Prewitt (e) Sobel (f) Roberts

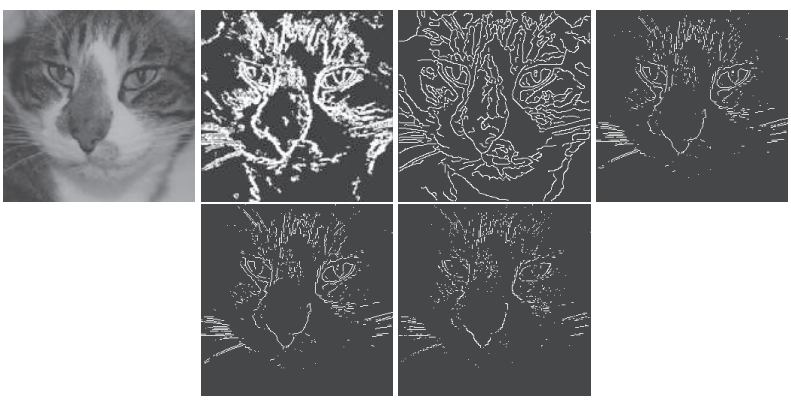

Fig. 16: Experimental result with Cat (a) Original image (b) ACO (c) Canny (d) Prewitt (e) Sobel (f) Roberts

best edge detection outcome using ACO was Cameraman. On the other hand, the image with the worst outcome was Cat. From these results, we conclude that the accuracy of ACO-based edge detection is unsurprisingly better for images with higher contrast.

Finally, in Experiment 2, we observed that the output has the best accuracy when the input images are processed with the Canny algorithm, a widely accepted opinion in the field. On the other hand, when the edge detection is performed with an ACO-based program, the accuracy of resulting edges is comparable to the operator based methods, hence with good parameter calibration ACO-based edge detection is a viable option.

References

1. Baterina, A.V., Oppus, C., "Image edge detection using ant colony optimization", WSEAS Trans. Sig. Process. 6(8), 5867, 2010.
2. Bonabeau, E., Dorigo, M., Theraulaz, G., *Swarm Intelligence: From Natural to Artificial Systems*, Oxford University Press, New York (1999)
3. Canny, J.F., "A computational approach to edge detection", IEEE Trans. Pattern Anal. Mach. Intell. (PAMI) 8(6), 679–697, 1986.
4. M. Dorigo, *Optimization, Learning and Natural Algorithms*, Ph.D. Thesis, Politecnico di Milano, 1992.
5. 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. 2941, 1996.
6. Marco Dorigo, Thomas Stützle, "Ant Colony Optimization", Massachusetts Institute of Technology, 2004
7. Stefka Fidanova and Zlatolilya Ilcheva "Application of Ants Ideas on Image Edge Detection", LSSC 2015, LNCS Vol. 9374, pp. 218–225, 2015.
8. Gonzalez, R.C., Woods, R.E., *Digital Image Processing*, Prentice-Hall Inc., Upper Saddle River (2002)
9. Jain, A.K., *Fundamentals of Digital Image Processing*, Prentice-Hall Inc, Upper Saddle River (1989)
10. D.-S. Lu, C.-C. Chen, "Edge detection improvement by ant colony optimization", Pattern Recognition Letters, vol. 29, pp. 416425, 2008.
11. Nezamabadi-pour, H., Saryazdi, S., Rashedi, E., "Edge detection using ant algorithm", Soft Comput. 10(7), 623628 (2006)
12. Pratt, W.K., *Digital Image Processing*, 2nd edn. Wiley, New York (1991)
13. G. T. Shrivakshan,Dr. C. Chandrasekar, "A Comparison of Various Edge Detection Techniques Used In Image Processing", IJCSI Vol. 9, Issue 5, No. 1, pp. 269-276, 2012.
14. N. Otsu, "A Threshold Selection Method From Gray-Level Histograms", IEEE Transactions On Systems, Man, And Cybernetics, Vol. SMC-9, pg. 62-66,1979
15. Anna Veronica Baterina, Carlos Oppus, "Image Edge Detection Using Ant Colony Optimization", WSEAS Transactions On Signal Processing, Issue 2, Vol. 6, 2010
16. J. Tian,W. Yu,S. Xie, "An Ant Colony Optimization Algorithm for Image Edge Detection", IEEE Congress On Evolutionary Computation, 2008
17. Zhou, P., Ye, W.Q., Wang, Q., "An improved canny algorithm for edge detection", J. Comput. Inf. Syst. 7(5), 15161523 (2011)