MEKATRONIKA
e-ISSN: 2637-0883
VOL. 4, ISSUE 1, 50 – 55
DOI: https://doi.org/10.15282/mekatronika.v4i1.7677

ORIGINAL ARTICLE

A Study on the Parameter Selection of Bat Algorithm in Optimizing Parameters in Camera Auto Calibration Problem

Rahaini Mohd Said[1], Khairul Azha A Aziz[1], Amar Faiz Zainal Abidin[1,*], Jessnor Arif Mat Jizat[2], Ismail Mohd Khairuddin[2], Sigit Widiyanto[3] and Mohamed Faisal Abdul Waduth[4]

[1]Faculty of Electrical & Electronics Engineering Technology, Faculty of Engineering Technology, UniversitiTeknikal Malaysia Melaka, 76100 Melaka, Malaysia.
[2]Faculty of Manufacturing and Mechatronics Engineering Technology, Universiti Malaysia Pahang, 26600 Pahang, Malaysia.
[3]Informatics Department, Gunadarma University, Jawa Barat, Indonesia.
[4]Faculty of Engineering, Universiti Selangor, Bestari Jaya, Malaysia.

ABSTRACT – In camera auto calibration, the major goal is to discover intrinsic parameter values that minimize the cost function. This study proposes to implement Bat algorithm, a stochastic optimization technique, to determine the optimum intrinsic parameter values. Each bat in the Bat Algorithm represents a potential solution to the issue, and each dimension in the Bat Algorithm's search space represents one of the basic parameters: skew, focal length, and magnification factor. The Kruppa's equation is the basis for the cost function in this study. By studying the echolocation behavior of the microbats, the bats will try to improve the fitness with each iteration. The Bat Algorithm's performance is evaluated using a case study from a database from Le2i Universite de Bourgoune. This paper studies the correlation of different parameters selection in Bat Algorithm in solving the camera auto-calibration problem. Finding shows that Bat Algorithm produces output that as expected as theory of Computational Intelligence suggested.

ARTICLE HISTORY
Received: 24th May 2022
Revised: 1s June 2022
Accepted: 16th June 2022

KEYWORDS
Bat algorithm,
Optimization,
Parameter Selection
Camera Auto Calibration

INTRODUCTION

The optimum intrinsic parameter values in the camera auto-calibration problem can be achieved by minimising the cost function given information such as the essential matrix and the fundamental matrix. Any optimization approach, including Genetic Algorithms, Particle Swarm Optimization, and Simulated Annealing, can be used to obtain the optimum value. In 2001, Y. Zhang and Q. Ji proposed using a Genetic Algorithm to solve the camera auto-calibration problem [6]. Genetic Algorithm is used to search for optimal interior and exterior camera parameters. The authors used synthetic and real images for their experiment. The authors concluded that Genetic Algorithm produced a superior performance in term of convergence, accuracy and robustness. In 2008, J. Z. Tao *et al.* proposed the implementation of Particle Swarm Optimization in camera auto-calibration problem process [1]. In the paper, the authors decided to find only the aspect ratio of the intrinsic parameters, which consists of f_u and f_v (focal length in pixels along the axes of the image). The skew, γ is let to 0. While the other two intrinsic parameters: $u_0$and $v_0$is ignored by S. Bougnoux[2]recommendation. The fitness or cost function used in optimizing the parameters is as recommended by R. I. Hartley [3]. In the same year, K. Bilal and J. Qureshi used Genetic Algorithm, Particle Swarm Optimization, and Simulated Annealing to investigate the applicability of nature-inspired optimization techniques for camera auto-calibration [4]. Their main objective is to have a performance benchmark between the algorithms based on several criteria: algorithm efficiency, algorithm accuracy, algorithm reliability, and calibration error. The authors then concluded that different optimization algorithms suitable in different conditions: if the application requires reliability, Genetic Algorithm fit really well, and if the application requires precision, Simulated Annealing or Particle Swarm Optimization will do the job better. In year 2009, X. Song *et al.* proposed another implementation of Particle Swarm Optimization for single camera calibration [5]. The result indicates that Particle Swarm Optimization provides decent calibration accuracy. There have also been attempts to combine different algorithms. To increase the accuracy of camera auto-calibration, J. Li et al. developed a hybrid of Genetic Algorithm and Particle Swarm Optimization [7]. The simplified Kruppa’s equation is used as cost function. The result indicates that the hybrid approach produced better success rate than having Genetic Algorithm or Particle Swarm Optimization as standalone.

From the literature, it is obvious that several attempts have been made to tackle the camera auto-calibration problem utilising Computational Intelligence's optimization methods. This is due to several reasons: easy to understand, easy to implement, and global optimization method [1]. Having mentioned the benefit of Computational Intelligence’s optimization, this paper attempt to experiment with one of the latest Swarm Intelligence called Bat Algorithm. In the second chapter will introduce reader to camera auto-calibration problem. In the third chapter, the reader will be introduced

*CORRESPONDING AUTHOR | Amar Faiz Zainal Abidin | amarfaiz@utem.edu.my

to the Bat Algorithm and shown how it is used to simulate the camera auto-calibration problem. The fourth chapter covered the result obtained from the experiment and in the fifth chapter, we will conclude the finding of the paper.

CAMERA AUTO-CALIBRATION PROBLEM

The main hindrance in camera auto-calibration is to obtain the optimal values of the intrinsic camera parameters for a given number of images. The word optimal here suggests that in theory, there should be zero error during point-to-point matching between the images, but in practice, this is hard to achieve. As the name suggested, auto camera calibration does not require any supervision by human. The intrinsic matrix is as shown in (1).

$$K = \begin{bmatrix} f_u & \gamma & u_0 \\ 0 & f_v & v_0 \\ 0 & 0 & 1 \end{bmatrix} \tag{1}$$

where$[u_0 v_0]^T \gamma$ is the skew, f_u is the product of focal length and magnification factor, ε. The magnification factor, ε is defined by (2).

$$f_v = \varepsilon \times f_u \tag{2}$$

Based on the recommended of previous literatures [1, 3], a point, p is on the absolute conic case, in which vector, $\boldsymbol{x} = (x, y, z)^{\boldsymbol{T}}$ will satisfy (3).

$$\boldsymbol{x}^T \boldsymbol{x} = 0 \tag{3}$$

Based on (3), it possible to extend the work of R. I. Hartley [3], where now each point, p must fit requirements: (4) and (5).

$$p^T K^{-T} K^{-1} p = 0 \tag{4}$$

$$p^T \omega\, p = 0 \tag{5}$$

which the dual absolute conic ofω, ω^* is as stated in (6).

$$\omega^* = KK^T \tag{6}$$

R. I. Hartley [3] simplified the Kruppa's equation to (7) where values of r and s come from diagonal matrix, D which is described in (8). Column vector, U are u_1, u_2, and u_3. Column vector, $V$are v_1, v_2, and v_3.

$$\frac{r^2 {v_1}^T \omega^* v_1}{{u_2}^T \omega^* u_2} = \frac{rs {v_1}^T \omega^* v_2}{-{u_2}^T \omega^* u_1} = \frac{s^2 {v_2}^T \omega^* v_2}{{u_1}^T \omega^* u_1} \tag{7}$$

$$D = \begin{bmatrix} r & & \\ & s & \\ & & 1 \end{bmatrix} \tag{8}$$

Assigning each part of (7) asJ_1, J_2, and J_3, (9) to (11) can be obtained.

$$J_{i1} = \frac{r^2 {v_1}^T \omega^* v_1}{{u_2}^T \omega^* u_2} \tag{9}$$

$$J_{i2} = \frac{rs {v_1}^T \omega^* v_2}{-{u_2}^T \omega^* u_1} \tag{10}$$

$$J_{i3} = \frac{s^2 {v_2}^T \omega^* v_2}{{u_1}^T \omega^* u_1} \tag{11}$$

Thus, the optimized value of the intrinsic matrix parameters can be obtained by finding the combination of parameters that minimize the error in (12).

$$Error = \sum_{z=1}^{im-1} \sqrt{{J_{i1}}^2 + {J_{i2}}^2 + {J_{i3}}^2} \tag{12}$$

where im is number of images. This paper Note that (12) will be use as fitness function for Bat Algorithm for this experiment.

MODELING CAMERA AUTO-CALIBRATION PROBLEM USING BAT ALGORITHM

Implementation of Computational Intelligence's optimization algorithms, in general, had been a great success. The algorithms implemented in a wide range of optimization problem such as Travelling Salesman Problem [15], VLSI problem [16,19], Printed Circuit Board's routing problem [12,14,18], and DNA sequence design problem [13,17]. Thus, the implementation of Computational Intelligence's optimization algorithm in camera auto-calibration problem comes naturally. This paper studies the use of Bat Algorithm in solving the problem. Bat Algorithm was introduced by X.-S. Yang in 2010 [11]. Bat Algorithm is inspired by the echolocation behavior of the microbats [11]. As mentioned earlier, each bat is a candidate solution of the problem, thus camera auto-calibration problem can be modeled as (13).

$$\boldsymbol{s}_m = [f_u\, \beta\, u_0\, f_v v_0]^T \tag{13}$$

where $\boldsymbol{s}_m$is the m-th bat position in the search space. It can be clearly seen that each parameters in camera auto-calibration problem represented by a dimension of the search space; 1st dimension for f_u, 2nd dimension for β, 3rd dimension for u_0, 4th dimension for f_v, and the 5th dimension for v_0. A simple example to give better idea of this model, for $\boldsymbol{s}_2 = [100\ 0\ 250\ 100\ 256]^T$ means that the 2nd bat in Bat Algorithm suggest that the parameters of the intrinsic matrix should be as follows:$f_u = 100, \beta = 0, u_0 = 250, f_v = 100, v_0 = 256$. The algorithm of Bat Algorithm as stated in Algorithm 1 which is adapted from the original Bat Algorithm [11].

The algorithm starts by initializing the bats positions in the search space randomly. In practice, the search space is usually limited to a known range of potential solutions values resides. In our case, we limit the range to [0, 1000] for each dimension of the search space. While $\boldsymbol{v}_m$is set 0 for all bats, initially. Pulse frequency, f_m is set randomly based on (14).

$$f_m = f_{min} + \beta(f_{max} - f_{\min}) \tag{14}$$

where$\beta \in [0,1]$ is a random value that comes from a uniform distribution. For this problem, $f_{min} = 0$and $f_{max} = 100$. The pulse frequency influence the velocity of the bat as stated in (15).

$$\boldsymbol{v}_m^z = \boldsymbol{v}_m^{z-1} + f_m(\boldsymbol{s}_m^{z-1} - \boldsymbol{s}_* \tag{15}$$

where$\boldsymbol{v}_m^z$ is the velocity of m-th bat at z iteration, $\boldsymbol{s}_m^z$ is the position of m-th bat at z iteration, and $\boldsymbol{s}_*$is the position of the global best location found so far. Thus, the new location or position of the bat for the next iteration as (16).

$$\boldsymbol{s}_m^z = \boldsymbol{s}_m^{z-1} + \boldsymbol{v}_m^z \tag{16}$$

Line 10 to 13 in the algorithm indicates that of the pulse rate of the bat is lower than a linear random value, the bat will perform a local search around its area using (18).

$$\boldsymbol{s}_m^z = \boldsymbol{s}_m^z + \varepsilon A_\#^z \tag{17}$$

where$\varepsilon \in [-1,1]$ is a linear random number and $A_\#^z$ is the mean loudness of all the bats at the given iteration. Then the bat will make a random walk according to Line 02 of the algorithm, and if the fitness obtained from the new position is better than the global best solution, and the bat loudness is greater than a linear randomly generated value, the solution is taken as the new bat position.

Table 1. Bat Algorithm for camera auto-calibration

Algorithm 1: Bat Algorithm for camera auto-calibration
01 Set fitness function, $fit(\boldsymbol{s}_m)$ according to (12) where $\boldsymbol{s}_m = [s_{m1}, s_{m2}, \ldots, s_{mn}]^T$

Generate randomly initial population of agent, $\boldsymbol{s}_m$ where $m = 1,2,..,q$
Generate randomly initial velocity of agent, $\boldsymbol{v}_m$ where $m = 1,2,..,q$
Define pulse frequency f_m at $\boldsymbol{s}_m$
Initialize pulse rates, r_m and the loudness A_m
while $z < t$
  for m = 1 to q
    Generate new solution by adjusting frequency using (14) and updating velocity
    using (15) and location using (16).
    if $rand > r_m$
      Select a solution among the best solutions
      Generate a local solution around the selected best solutions using (17)
    end if
    Generate a new solution by flying randomly using (18)
    if $(rand < A_m)$ & $(fit(\boldsymbol{s}_m) < f(\boldsymbol{s}_*))$
      Accept the new solutions
      Increase r_m and reduce A_m
    end if
    Rank the bats and find the current best $\boldsymbol{s}_*$ using (12)
  end for m
end while
Post process results and visualization

In this paper, the values used for pulse rates, r_m and the loudness A_m as suggested in the original paper [11] which stated in (18) and (19).

$$A_m^z = 0.9 \times A_m^z \tag{18}$$

$$r_m^{z+1} = r_m^z \times e^{-(0.9 \times z)} \tag{19}$$

Values of A_m^0 and r_m^0 are randomly assigned in the range of [0, 0.5] and [0.5, 1], respectively. Note that the values of A and r are going to decrease as the iteration incereases. This ensure at the beginning of the simulation, the bat will focus more on exploitation of global solution and at the end of the phase of the simulation, the bat will focus more on exploration of local solution.

The fitness of each bat is then ranked, and if the bats have a higher fitness than the global best solution, the global best solution is updated. The cycle is continued until the stopping condition is met. The best global solution is then presented.

RESULT

Table 2 indicates the parameters of Bat Algorithm used in obtaining the experimental result. The case study was obtained from a database given by Le2i Universite de Bourgoune. [9-10]. The Matlab platform is used to write the algorithm. The simulation was performed using a Windows 10's Operating System with a 2.11 GHz Intel Core i5 (10th generation) CPU and 8 GB DDR4 RAM.

Table 2. Proposed approach's parameters.

Parameter	Value
Number of computation	10
Number of iteration, t	[10, 5000]
Number of bat, q	[2, 50]
Pulse rate at initial iteration, r	[0.5, 1]
Loudness at initial iteration, A	[0, 0.5]
Pulse frequency, f	[0, 100]

Initial finding presented in SETNC 2013 [20] suggested that the Bat Algorithm approach the problem by either converges prematurely (as seen by the convergence curve in Figure 2) or does not (refer Figure 1). In the same proceeding, it is dictated that these two cases are extreme to each other. A reasonable explanation for these extreme cases is because poor selection of Bat Algorithm's parameters. Thus, prompted this study where attempts to examine the cause and effect of the parameters selection of Bat Algorithm in more depth.

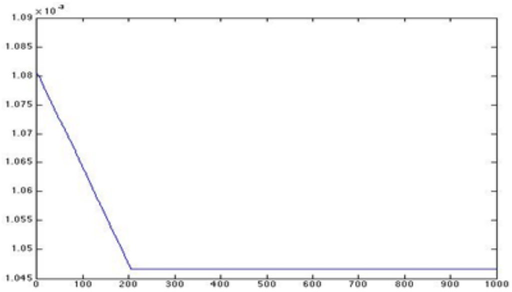

Figure 1. Premature Convergence.

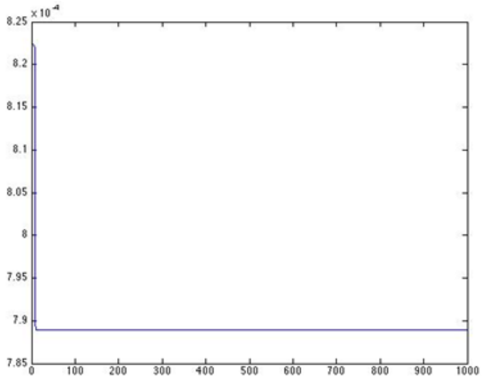

Figure 2. No Converge

The results of the extensive simulation are summarized in Table 3. Note that the error is in 10^{-3}, thus 0.13466 displays in combination of parameters of q = 2 and t = 10 is an actual error of 1.3466×10^{-3}. Theory proposed that an increase number of agent (to a certain value) should be able to reduce the error. Result obtained shows inconsistency which mainly occurred for all combination of iteration for q = 10 and q = 20. This inconsistency can be seen either the result is superior compared to a higher number of agents for the same iteration number. Or the result is inferior compared to a smaller number of agents for the same iteration number. Similar pattern can be seen when increasing the number of iteration. In theory, a greater number of iteration should be able to minimize the error. Instead, result shows inconsistency which mainly occurred for all combination of agents for t = 1000. Having said that, most of the findings are still inline with the theory.

Table 3. Average error (in 10^{-3}) for different parameters of q and t

		Number of bat, q				
		2	5	10	20	50
Number of iteration, t	10	0.142115	0.147437	0.139705	0.135665	0.127082
	20	0.147463	0.165.221	0.142554	0.142672	0.143651
	50	0.142331	0.136062	0.133818	0.124742	0.127799
	100	0.147518	0.134374	0.142514	0.142536	0.130775
	200	0.135414	0.130368	0.142604	0.140718	0.133703
	500	0.140758	0.132676	0.131042	0.142148	0.130908
	1000	0.142774	0.145745	0.134916	0.141575	0.14129
	2000	0.138743	0.130105	0.134152	0.142164	0.130811
	5000	0.134789	0.137064	0.12698	0.124756	0.143409

Another interesting subject to be studied is the average percentage of global convergence's iteration. Global convergece iteration is defined as the iteration number in which the agents obtained found the best soultion for the given simulation. Low percentage (0 to 50%) indicates that the algorithm prematurely converge. Ideally, the percentage needed is around 60% to 80%. While percentage above 90% might suggested that the algorithm still requires more time (or iteration in this case) to find the optimized solution. Based on Table 4, as iteration increased for a given number of agent, the average of percetage of global convergence increases.

Table 4. Average percentage of global convergence for different parameters of q and t

		Number of bat, q				
		2	5	10	20	50
Number of iteration, t	10	20	25	40	20	20
	20	25	40	41	43	50
	50	40	45	50	49	51
	100	51	51	51	53	55
	200	53	55	57	58	58
	500	61	63	65	66	70
	1000	60	67	71	70	70
	2000	68	71	71	73	75
	5000	70	75	76	81	86

CONCLUSION

This paper studies the impact of Bat Algorithm's parameters variation towards solution obtained for camera auto-calibration problem. The aim of the paper to further validate the finding presented earlier in SETNC 2013. The results indicate that he Bat Algorithm's parameter variation demonstrates a positive cause-and-effect correlation. It can be concluded that the result obtained are inline with the Computational Intelligence theory.

REFERENCES

[1] J. Z. Tao, W. Wenhuan, W. Min, "Camera Autocalibration from Kruppa's Equations Using Particle Swarm Optimization," Proceeding of International Conference on Computer Science and Software Engineering, pp. 1032-1034, 2008.

[2] S. Bougnoux, "From Projective to Euclidean Space Under Any Practical Situation, A Criticism of Self-Calibration," Proceeding of 6th International Conference on Computer Vision, pp. 790-796.

[3] R. I. Hartley, "Kruppa's Equations Derived from the Fundamental Matrix," IEEE Transactions on Pattern Analysis and Machine Learning, vol. 19, no. 2, pp. 133-135, 2007.

[4] K. Bilal, J. Qureshi, "Nature Inspired Optimization Techniques for Camera Calibration," Proceeding of International Conference on Emerging Technologies, pp. 27-31, 2008.

[5] X. Song, B. Yang, Z. Feng, T. Xu, D. Zhu, Y. Jiang, "Camera Calibration Based on Particle Swarm Optimization," Proceeding of International Congress on Image and Signal Processing, pp. 1-5, 2009.

[6] Q. Ji, Y. Zhang, "Camera Calibration with Genetic Algorithms," IEEE Transactions on Systems, Man, and Cybernatics, Part A: Systems and Humans, vol. 31, no. 2, pp. 120-130, 2001.

[7] J. Li, Y. Yang, G. Fu, "Camera Self-Calibration Method Based on GA-PSO Algorithm," Proceeding of IEEE International Conference on Cloud Computing and Intelligence Systems, pp. 149-152, 2011.

[8] J. Kennedy, R. Eberhart, "Particle Swarm Optimization," Proceeding of IEEE International Conference on Neural Networks, vol. 4, pp. 1942-1948, 1995.

[9] University of Burgundy, Camera Autocalibration Dataset, URL: http://goo.gl/qxerh [Date Accesed: 19th January 2012].

[10] G. Lemaitre, "3D Digitization: Camera Autocalibration," URL: http://goo.gl/s8CTT [Date Accesed: 19th January 2012].

[11] X. S. Yang, "A New Metaheursitic Bat-Inspired Algorithm," Nature Inspired Coopeartive Strategies for Optimization, Studies in Computational Intelligence, vol. 284, pp. 65-74, 2010.

[12] M. M. Ismail, M. A. Othman, H. A. Sulaiman, M. H. Mirsan, R. H. Ramlee, A. F. Z. Abidin, N. A. Nordin, M. I. Zakaria, M. N. Ayob, F. Yakop, "Firefly Algorithm in PCB Holes Drilling Process," Proceeding of 2012 International Conference in Green and Ubiquitous Technology, pp. 110-113, 2012.

[13] F. Yakop, Z. Ibrahim, A. F. Z. Abidin, Z. M. Yusof, M. S. Mohamad, K. Wan, J. Watada. "An Ant Colony System for Solving DNA Sequence Design Problem in DNA Computing," International Journal of Innovative Computing, Information and Control 8, vol. 10(B), pp. 7329-7339.

[14] M. S. Saelal, A. F. Z. Abidin, A. Adam, J. A. A. Mukred, K. Khalil, Z. M. Yusof, Z. Ibrahim, N. A. Nordin, "An Ant Colony System for Routing in PCB Holes Drilling Process," International Journal of Innovative Management, Information & Production, vol. 3, no. 1, pp. 50-56.

[15] M. M. Ismail, M. I. Zakaria, A. F. Z. Abidin, J. M. Juliani, A. Lit, S. Mirjalili, N. A. Nordin, M. F. M. Saaid, "Magnetic Optimization Algorithm Approach for Travelling Salesman Problem," Proceeding of World Academy of Science, Engineering and Technology, pp. 1393-1397, 2012.

[16] Z. M. Yusof, A. F. Z. Abidin, A. Adam, K. Khalil, J. A. A. Mukred, M. S. Mohamad, M. K. Hani, Z. Ibrahim, "A Two-Step Binary Particle Swarm Optimization Approach for Routing in VLSI," ICIC Express Letters, vol. 6, no. 3, pp. 771-776, 2011.

[17] S. M. Mustaza, A. F. Z. Abidin, Z. Ibrahim, M. A. Shamsudin, A. R. Husain, J. A. A. Mukred, "A Modified Computational Model of Ant Colony System in DNA Sequence Design," Proceeding of 2011 IEEE Student Conference on Research and Development, pp. 187-191, 2011.

[18] M. S. Saelal, A. F. Z. Abidin, A. Adam, N. Nordin, K. Khalil, J. A. A. Mukred, Z. M. Yusof, Z. Ibrahim, "An Ant Colony System for Routing in PCB Holes Drilling Process," Proceeding of the International Symposium on Innovative Management, Information & Production, pp. 292-297, 2011.

[19] Z. M. Yusof, A. F. Z. Abidin, M. N. A. Salam, K. Khalil, J. A. A. Mukred, M. K. Hani, Z. Ibrahim, "A Binary Particle Swarm Optimization Approach for Buffer Insertion in VLSI Routing," Proceeding of the International Symposium on Innovative Management, Information & Production, pp. 7-12, 2011.

[20] K. A. M. Annuar, N. A. Selamat, S. H. A. Razak, A. F. Z. Abidin, S. F. A. Latip, M. F. A. M. Kasai, N. A. Nordin, M. A. Mansor, J. A. M. Jizat, H. C. Shamssudin, S. Widiyanto, H. M. Fadilah, M. F. A. Waduth, K. A. M. Nor, "A Preliminary Study on Camera Auto Calibration Problem using Bat Algorithm," Proceeding of Science & Technology National Conference 2013, 2013.