Turkish Journal of Electrical Engineering & Computer Sciences

http://journals.tubitak.gov.tr/elektrik/

Research Article

Turk J Elec Eng & Comp Sci
(2017) 25: 4545 – 4557

doi:10.3906/elk-1704-167

A novel hybrid approach based on a chaotic cloud gravitational search algorithm to complicated image template matching

Weijia CUI*, Yuzhu HE
School of Instrumentation Science and Opto-Electronics Engineering, Beihang University, Beijing, P.R. China

Received: 14.04.2017 • **Accepted/Published Online:** 15.09.2017 • **Final Version:** 03.12.2017

Abstract: Template matching is the process of accurately extracting the interesting regions in a source image according to reference templates. In this paper, the gravitational search algorithm (GSA) is employed as a novel search strategy for template matching. However, the basic GSA is easily trapped in a local optimum and has a poor exploitation ability. In this paper, to enhance the optimization performance of GSA, a novel cross-search strategy based on chaotic global search (CGS) and cloud local search (CLS) is incorporated into GSA. The new variant is named chaotic cloud GSA (CCGSA). CGS makes full use of the ergodicity of chaos theory to improve global search ability and to avoid premature convergence. Inspired by the randomness and stable tendency of the normal cloud model, CLS was formed to realize a refined exploitation in the neighborhood of the current best solution; therefore, it can enhance optimization efficiency. Comparative experiments on six composite benchmark functions indicate that CCGSA convergence performance is superior to that of two advanced variants of GSA. Moreover, when applied to template matching, CCGSA performs better than the other selected intelligent optimization algorithms.

Key words: Template matching, gravitational search algorithm, chaotic global search, cloud local search, optimization problem

1. Introduction

Template matching is a process for searching a source image for a subgraph that is most similar to that in a template image. Template matching is one of the key technologies for computer vision [1], target detection [2], and object tracking [3]. Therefore, developing a matching method that is effective for high-speed searching and provides accurate matching and outstanding robustness has become the main aim of researchers in this field.

In template matching, the choices of similarity measurement and search strategy determine the matching efficiency. Typical measures of similarity are the sum of absolute differences (SAD) [4], the sum of squared differences (SSD) [5], and normalized cross-correlation (NCC) [6]. All of these measures can be used to effectively evaluate the similarity between two images. However, previous studies show that SAD and SSD are sensitive to the variations of noise and illumination, and their performance is slightly inferior to that of the NCC model [7]. Therefore, considering its robustness for matching, the NCC model is used as the similarity measure in this paper.

To obtain the best matching location, a traditional search method has to measure similarity pixel by pixel in the feasible search region. This full search method would result in a huge cost and take a long time. In recent years, to enhance matching efficiency, many effective search strategies have been developed, and these mainly

*Correspondence: natalienan@buaa.edu.cn

fall into two categories. One is scaling down the search area by using a feature-extraction method [8], a pyramid decomposition method [9], or a wavelet decomposition method [10]. The other strategy is to introduce swarm intelligence algorithms, such as particle swarm optimization (PSO) [11], differential search algorithm (DSA) [12], or biogeography-based optimization [13], as an optimization strategy to guide the matching process. In this paper, GSA [14,15], a novel swarm intelligence algorithm, is introduced to perform the matching task.

GSA has been successfully employed to solve many practical engineering problems, such as feature selection [16] and parameter identification [17]. However, similar to other swarm intelligence algorithms, GSA also has such drawbacks as low convergence accuracy and slow optimization speed, and it is easily trapped in a local optimum. Therefore, in order to overcome these shortcomings, a novel variant of GSA, chaotic cloud GSA (CCGSA), was developed by incorporating chaos theory and the cloud model into the basic GSA.

Chaos theory, which can provide a powerful global search tool, is usually used to improve the global search ability of swarm intelligence algorithms [18–20]. The ergodicity of the chaotic mechanism can make the individuals travel through all positions in the whole space to the greatest extent possible. As an uncertainty conversion model, the cloud model characterizes the relationship between qualitative concepts and quantitative expressions [21]. The model possesses the properties of uncertainty with certainty as well as stability through changes, i.e. randomness and stable tendencies [22]. Using these characteristics, a local search method based on the cloud model is created to effectively exploit a promising solution around the current best position. In addition, a cross-search strategy is proposed to switch the chaotic global search and the cloud local search.

Finally, six composite benchmark functions [23] are used to test and evaluate the search ability of CCGSA. In addition, two novel variants of GSA, chaotic gravitational search constant-based GSA (CGSA) [24] and escape velocity operator-based GSA (EVGSA) [25], are introduced for the purpose of performance comparison. The experimental results demonstrate that the proposed algorithm has a competitive convergence performance compared with CGSA and EVGSA. Moreover, a series of comparative experiments on template matching indicates that CCGSA is superior to five other selected competitors.

The remainder of this paper is organized as follows: Section 2 reviews the principle of GSA; Section 3 elaborates on the cross-search mechanism based on chaotic global search and cloud local search; Section 4 introduces a normalized cross-correlation function and describes the procedure of template matching based on CCGSA; Section 5 verifies the search ability of CCGSA on six composite functions and conducts the template matching with CCGSA for two complicated cases. The study's conclusions are presented in the last section.

2. GSA

GSA was developed from the law of universal gravity [14]. Each individual is viewed as an object with mass. Any two objects attract each other through the force of gravity and move according to the rules of kinematics. The motion of the objects will always tend toward the heavier objects. In the iterative process of GSA, the performance of each individual is evaluated according to its mass, and all the objects eventually converge to the heaviest one, i.e. the global optimal solution. Consider that the position of the ith object at time t in D dimensional space is defined as

$$X_i(t) = (x_i^1, x_i^2, \ldots, x_i^d, \ldots, x_i^D), \tag{1}$$

where x_i^d denotes the position of the dth dimension of object i. After all the objects are evaluated by the fitness function, the inertial mass M_i of object i is calculated as follows:

$$M_i(t) = \frac{fit_i(t) \ - \ worst(t)}{\sum_{j=1}^{N} fit_j(t) \ - \ worst(t)}, \tag{2}$$

where fit_i is the fitness value of object i; N is the population size; $worst(t)$ denotes the worst fitness value among the population.

The resultant force F_i^d of direction d acting on the object i is the sum of the component forces F_{ij}^d of the other $N-1$ objects respectively acting on it. The formula is given as follows:

$$\begin{cases} F_{ij}^d(t) \ = \ G(t) \ \frac{M_i(t) \times M_j(t)}{R_{ij}(t) + \varepsilon} \ (x_j^d(t) \ - \ x_i^d(t)) \\ F_i^d(t) \ = \ \sum\limits_{j=1, \, j \neq i} rand_j F_{ij}^d(t) \end{cases}, \tag{3}$$

where M_i and M_j denote the masses of object i and j, respectively; $rand_j$ is a random number generated from [0, 1]; $R_{ij}(t)$ is the Euclidean distance between objects i and j; ε is a small positive constant; $G(t)$ is the gravitational constant and $G(t) = G_0 \times e^{-\alpha t/T}$, where G_0 and α are the initial constants and T is the maximum iterations.

Next, the acceleration α_i^d of object i in direction d is computed as follows:

$$a_i^d(t) \ = \ \frac{F_i^d(t)}{M_i} \tag{4}$$

Subsequently, the next velocity v_i^d and position x_i^d are updated according to Eqs. (5) and (6):

$$v_i^d(t \ + \ 1) \ = \ rand_i \ \times \ v_i^d \ + \ a_i^d(t) \tag{5}$$

$$x_i^d(t \ + \ 1) \ = \ x_i^d(t) \ + \ v_i^d(t \ + \ 1) \tag{6}$$

The simplified procedure of GSA is

Step 1: Initialize the population X randomly.

Step 2: Evaluate $fit(t)$ with the object function.

Step 3: Update the *worst* and current best solution X_{best}.

Step 4: Calculate $M(t)$ according to Eq. (2).

Step 5: Update $\alpha(t)$, $v(t)$, and $x(t)$ by using Eqs. (3) through (6).

Step 6: Repeat steps 2–5 until the maximum iteration is reached.

3. CCGSA

3.1. Chaotic global search (CGS)

Using chaos theory's properties of randomness, ergodicity, and nonrepetition [18], CGS is constructed to guide the population to search all over the solution space and thus help the algorithm jump out the local optimum. In this paper, a logistic map [20] is introduced as a typical chaotic sequence to fulfill the requirements of CGS. The iterative formula of a one-dimensional logistic map is

$$cx_{i+1} \ = \ \mu \ \times \ cx_i \ \times \ (1 \ - \ cx_i), \tag{7}$$

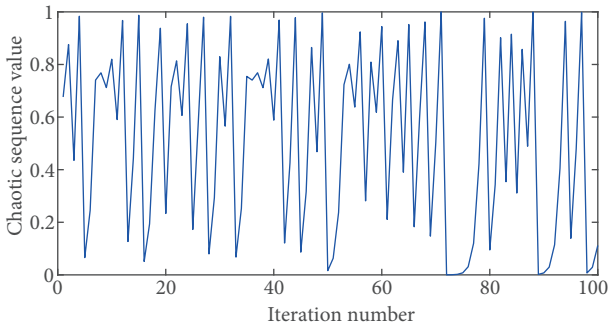

Figure 1. Logistic map function.

where $i = 1, 2, ...$; $\mu = 4$; cx_i denotes the chaotic variable and satisfies $0 \leq cx_i \leq 1$; the initial value cx_0 is generated randomly in [0, 1] and $cx_0 \neq \{0, 0.25, 0.5, 0.75, 1\}$. Figure 1 demonstrates the iterative curve of a logistic map. The procedure of CGS is as follows:

Step 1: Initialize the corresponding parameters of the logistic map; assume the jth element in the ith chaotic vector is cx_i^j, $j = 1, 2, ..., D; i = 1, 2, ..., T_{chaos}$, where T_{chaos} denotes the maximum iteration of the chaotic search.

Step 2: Generate a D dimensional starting vector cx_0 in [0, 1] randomly and set $i = 1$.

Step 3: Generate a D dimensional chaotic vector by using Eq. (7); then set $j = 1$.

Step 4: Project the chaotic variable cx_i^j to decision variables $chaos\_x_i^j$ by using Eq. (8).

$$chaos\_x_i^j = low^j + cx_i^j \times (up^j - low^j), \tag{8}$$

where low^j and up^j are the lower and upper bounds of the jth dimensional decision variables, respectively.

Step 5: Set $j = j + 1$; repeat Step 4 until the ith decision vector is completely projected from D dimensional chaotic space.

Step 6: Set $i = i + 1$; if the maximum iteration is not reached, return to Step 3; otherwise proceed to Step 7.

Step 7: Evaluate the T_{chaos} new individuals $chaos\_x$ and determine the best individual, CC_{best}. If CC_{best} is superior to the current best position X_{best}, substitute X_{best} for CC_{best}.

3.2. Cloud local search (CLS)

The cloud model is an uncertain conversion tool for characterizing the relationship between qualitative concepts and quantitative expressions [21,22]. In the cloud model, three digital feature values (the expectation *Ex*, the entropy *En*, and the hyperentropy *He*) are used to reflect the randomness and stable tendency of a concept in an exact quantitative manner. *Ex* is the center of gravity of the cloud representing the most typical sample for quantifying the concept; *En* denotes the distribution scope of the cloud; *He* reflects the thickness and dispersion of the cloud.

3.2.1. Definition of normal cloud model

The normal cloud model is a commonly used cloud model and is constructed based on normal distribution. The definition [22] is as follows: Let C be the qualitative concept related to the quantitative domain U. If the

quantitative value x in U denotes a random implementation of C, and if $x$follows the two conditions:(i) $x \sim Normal(Ex,\ H)$, where $H \sim Normal(En,\ He)$; (ii) the certainty degree $\mu(x)$ that x belongs to C satisfies:

$$\mu(x) = e^{-\frac{(x-Ex)^2}{2H^2}} \tag{9}$$

then the overall distribution of x in U is seemed as a normal cloud and each x can be regarded as a cloud droplet. The normal cloud generator is used to generate the cloud droplets and can be formulated as follows:

$$x = NCG(Ex,\ En,\ He) \tag{10}$$

Figure 2 demonstrates various normal cloud models based on different digital feature values. Digital feature values directly determine the distribution of cloud droplets. More specifically, the greater En is, the wider the distribution range of cloud droplets is; the larger He is, the more discrete the cloud droplets are.

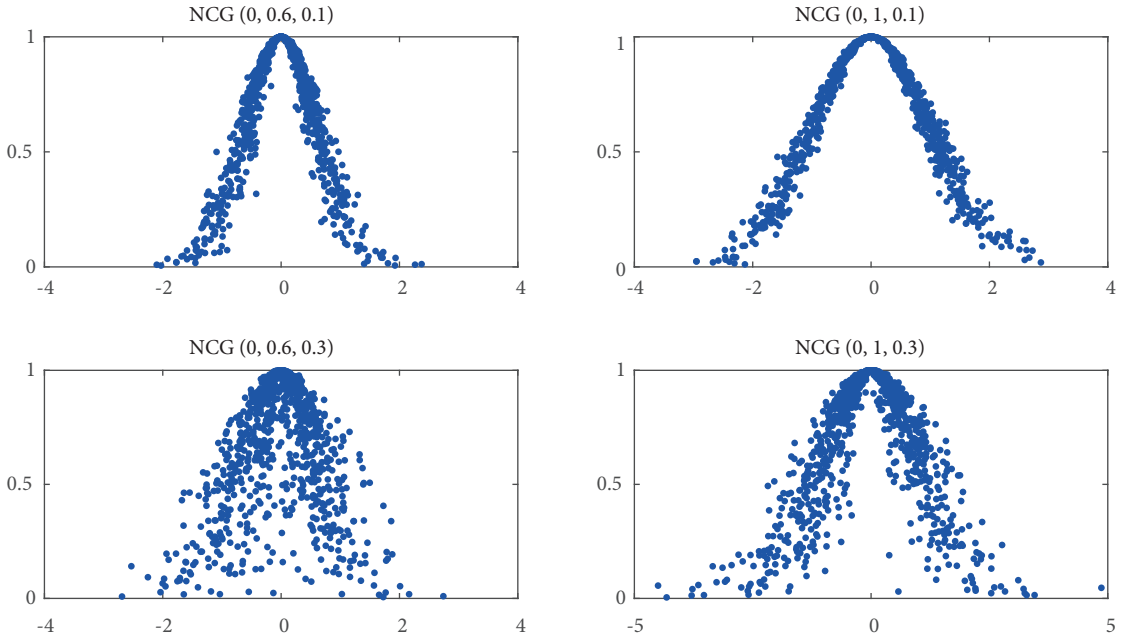

Figure 2. Normal cloud models with various parameters.

3.2.2. Procedure of CLS

There is a higher probability of finding the global optimum or a more promising solution around the current best location [17]. To achieve this, the normal cloud model is utilized to conduct a fine search around the current optimal location. The motivation for this is that the cloud model could embody the essence of population evolution: Ex corresponds to the genetic quality of the parent and the stability of inheritance by the offspring; En corresponds to the mutation scope and search extent; He corresponds to the degree of evolutionary stability and the agglomeration of the offspring. Therefore, the cloud droplets are considered as the new objects swarm, and the randomness and stable tendency of these new individuals are helpful for better movement to the X_{best} to obtain a more accurate solution. The steps of CLS are as follows:

Step 1: Initialize the number of cloud droplets N_{cloud}; the jth element in the ith cloud droplet is denoted as $cloud\_x_i^j$, $i = 1, 2, ..., N_{cloud}$, $j = 1, 2, ..., D$; set $i = 1, j = 1$.

Step 2: Update the three digital feature values, Ex^j, En^j, and He^j, by using Eq. (11).

$$\begin{cases} Ex^j = X_{best}^j \\ En^j = (up^j - low^j)/(0.5 \times (1 + rm) \times t) \\ He^j = En^j \end{cases}, \tag{11}$$

where rm is a random number generated in [0, 1]. From Eq. (11), En and He are dynamically decreased over time t. In this manner, the population diversity can be better kept in the initial stage, and, as the iteration proceeds, the gradual reduction in En and He ensures the convergence rate and precision.

Step 3: Generate a Gaussian distribution-based random value H^j with the mean value En^j and variance He^j.

Step 4: Generate a Gaussian distribution-based random value $cloud\_x_i$ with the mean value Ex^j and variance H^j.

Step 5: Set $j = j + 1$ and repeat Steps 2–5 until the ith cloud droplet is completed in Ddimensional space.

Step 6: If the N_{cloud} cloud droplets are all completed, skip to Step 7; otherwise let $i = i + 1$ and return to Step 2.

Step 7: Calculate the fitness values of the N_{cloud} cloud drops $cloud\_x$ and determine the best individual, CC_{best}, among $cloud\_x$. If CC_{best} is superior to X_{best}, then substitute CC_{best} for X_{best}.

3.3. Cross-search strategy

From Sections 3.1 and 3.2, CGS and CLS are developed to improve the solution quality of X_{best} at each iteration. As shown in Figure 3, CGS can search for the global optimization solution in a larger area to improve the ability to break away from local optima. CLS works in the surrounding area of the current optimal location X_{best} to obtain a better solution. To balance the exploration and exploitation ability of GSA better, a cross search strategy is proposed for switching CGS and CLS with probability p_m. The expression for the strategy is

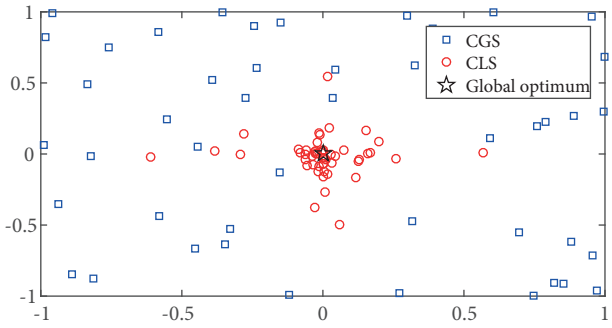

Figure 3. Schematic diagram of chaotic cloud search.

$$Conduct \begin{cases} chaotic\,global\,search & if\,rand < p_m \\ cloud\,local\,search & otherwise \end{cases} \tag{12}$$

Thus, CCGSA is developed by integrating a cross-search strategy into the basic GSA. To clearly illustrate CCGSA, a complete flowchart for CCGSA is shown in Figure 4.

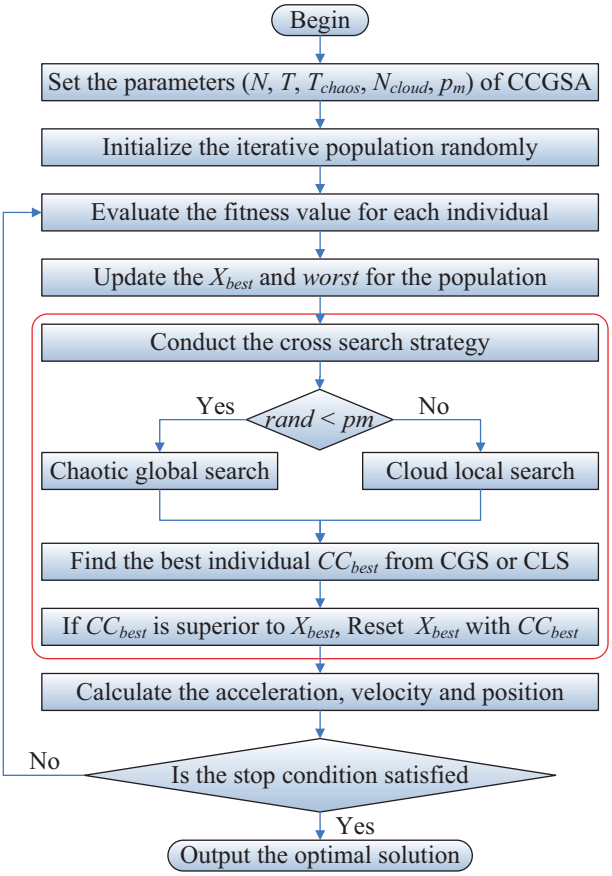

Figure 4. A flowchart for CCGSA.

4. Template matching based on CCGSA

4.1. Fitness function

In this paper, the search task in template matching is performed by using CCGSA. The similarity criterion, normalized cross correlation (NCC) [7], is regarded as the fitness function. The scalar product form of NCC is as follows:

$$R(u,\ v)\ =\ \frac{temp^T\ \cdot\ sub_{(u,\ v)}}{\sqrt{temp^T\ \cdot\ temp}\ \cdot\ \sqrt{sub^T_{(u,\ v)}\ \cdot\ sub_{(u,\ v)}}}, \tag{13}$$

where $temp$ is the template image; sub is a subset of the source image; $temp^T$ and sub^T are the transpositions of $temp$ and sub, respectively; and (u, v) is the top left coordinate of the subset image.

Figure 5 illustrates the basic principle of template matching. S represents the source image; the size of S is $M \times N$; the size of $temp$ is $m \times n$; sub is covered by $temp$ and they have an equal size; and the feasible search region is: $u \in [1, M - m + 1], v \in [1, N - n + 1]$. The matching process finds the optimal coordinate (u, v) in the feasible search region by calculating the similarity between the template image and the subset of the source image. $R(u, v) = 1$ represents the best matching location being obtained.

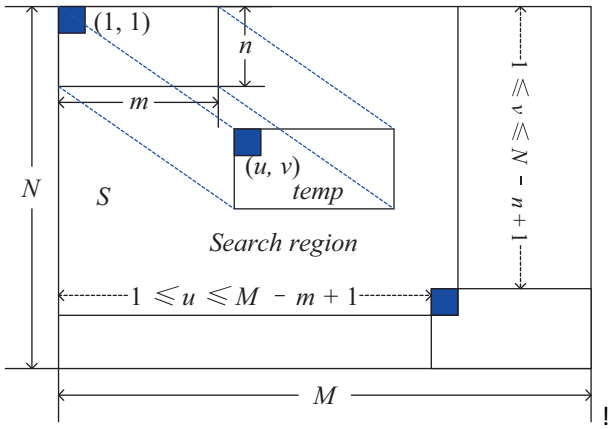

Figure 5. Basic principle of template matching.

4.2. Implementation of CCGSA in template matching

Step 1: Load source image and the template image, and convert them (both are RGB) to gray scale format.

Step 2: Set the population size N, the maximum iterations T for GSA; set the T_{chaos} for CGS and N_{cloud} for CLS; set probability p_m for the cross search strategy.

Step 3: Initialize N individuals randomly. For image matching, the dimension D is 2.

Step 4: Conduct the basic GSA to update the individuals.

Step 5: Conduct the cross-search strategy to improve the X_{best}.

Step 6: If the algorithm reaches the maximum iteration or finds out the best matching location, terminate the CCGSA; otherwise go back to Step 4.

5. Simulation and comparison

In this section, two types of experiments were conducted to evaluate the CCGSA model. One is model verification on six challenging benchmark functions; the other is model application to template matching. The test environment was configured with a 3.20 GHz CPU, 3568 MB RAM under the Windows XP operating system. All experiments are simulated with MATLAB R2011b.

5.1. Model verification on composite benchmark functions

To verify the performance of the proposed algorithm, six composite test functions [23] (CF1–CF6) were used to test its search ability. The dimension of the six functions was 10 and the search range is [–5, 5]. Detailed information about these six functions is listed in [23,25]. Moreover, two novel improved versions of GSA, CGSA [24], and EVGSA [25], were introduced to conduct a comparative experiment. The related parameters of CGSA and EVGSA are in accordance with those in [24,25]. For CCGSA, $T_{chaos} = N_{cloud} = N$, $p_m = 0.5$. To ensure the fairness of the comparison, the population size N is 30, and the fitness evaluation number is 15,000, $G_0 =$ 100, $\alpha = 20$.

Table 1 lists the statistical results including mean and standard deviation over 30 independent runs. CCGSA outperformed CGSA and EVGSA on all six functions in terms of convergence accuracy and robustness. To more intuitively reflect the optimal effect, Figure 6 shows the average evolutionary curves of the three algorithms over 30 independent runs. It is clear that CCGSA demonstrates a stronger global convergence ability

and faster convergence rate. The superiority of CCGSA indicates that the cross-search strategy combining CGS and CLS contributes to GSA to achieve a perfect compromise between global exploration and local exploitation.

Table 1. Comparative results on composite benchmark functions.

CF	CGSA		EVGSA		CCGSA	
	Mean	Std	Mean	Std	Mean	Std
CF1	60.215	53.525	108.501	38.015	16.667	37.929
CF2	37.541	69.560	133.803	106.977	16.921	46.025
CF3	172.538	38.922	207.901	40.998	131.326	27.860
CF4	323.679	56.089	374.032	105.251	286.238	10.080
CF5	108.127	61.459	191.992	120.856	32.075	45.347
CF6	847.553	111.766	869.658	87.517	522.873	82.309

5.2. Model application to template matching

To evaluate the performance of CCGSA in solving the template matching problems, a series of experiments was conducted on two complicated cases. Several intelligent algorithms were introduced to compare with CCGSA: PSO, CPSO, DSA, and CDSA [11,12]. The population size and maximum iterations for all algorithms were set to 100 and 500, respectively. The parameter settings of all algorithms are given as follows:

(1) GSA and CCGSA: $G_0 = 1$, $\alpha = 5$, $T_{chaos} = N_{cloud} = 30$, $p_m = 0.5$.

(2) PSO and CPSO: $C_1 = C_2 = 2$, $\omega_{\max} = 1.2$, $\omega_{\min} = 0.3$.

(3) DSA and CDSA: $p_1 = 0.8\text{rand}$, $p_2 = 0.8\text{rand}$.

Figure 7 respectively demonstrates the source images, template images, and matching results on the two cases. The matching results marked by the red rectangles shows that CCGSA is practical and effective in finding out the best matching subimage from the complex source image. Figures 8a and 8c show the evolutionary curves for CCGSA over 10 independent runs on the two cases. It is obvious that the proposed method possesses a fast convergence speed and stable convergence performance. Figures 8b and 8d demonstrate the average comparative curves for the GSA, CCGSA, PSO, CPSO, DSA, and CDSA obtained with 20 independent runs. The basic GSA has difficulty successfully obtaining an accurate matching location. The proposed method, on the other hand, can converge to the global optimal value, i.e. $R(u, v) = 1$ with the fewest average iterations among these algorithms. In addition, the comparative results also reflect that CCGSA has a higher robustness and matching success rate than the other competitors.

Table 2 lists the average time consumption for GSA, CCGSA, CDSA, CPSO, DSA, and PSO, which are the algorithms that successfully located the best matching position over 30 times. It is clear that CCGSA takes the shortest time out of these competitors. In sum, the experimental results indicate that the cross-search strategy greatly improves GSA performance in matching accuracy and efficiency; additionally, CCGSA shows high competitiveness among these algorithms in template matching.

6. Conclusion

This paper proposes a novel variant of GSA by incorporating a new cross-search strategy based on CGS and CLS into GSA. In CCGSA, CGS, which is based on chaos theory, can enhance population diversity and help

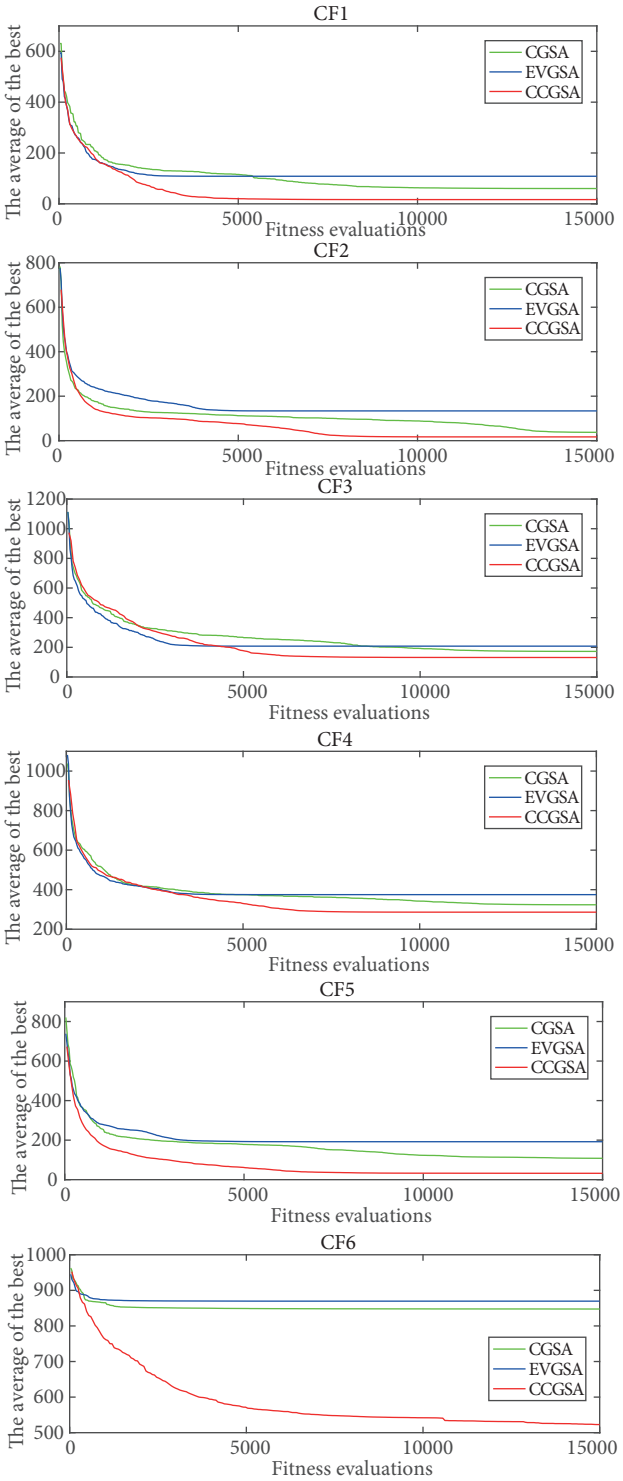

Figure 6. Average evolutionary curves using CGSA, EVGSA, and CCGSA.

overcome the premature convergence problem of GSA. CLS can allow GSA to quickly determine the potential most-promising solution around the current best solution. The experimental results of six composite benchmark functions show that with the help of the cross-search strategy CCGSA has a favorable balance of exploration

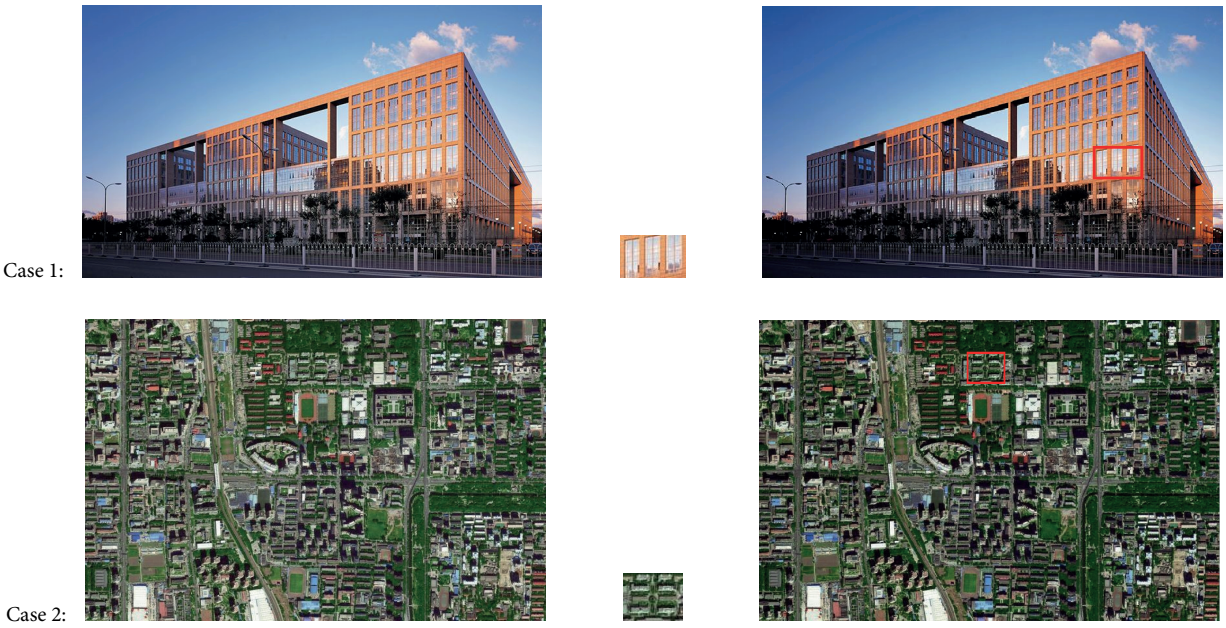

Figure 7. Test cases: a) Source images. Size: case 1: 419 × 709, case 2: 570 × 870; b) Template images. Size: case 1: 49 × 75, case 2: 59 × 72; c) Matching results using CCGSA.

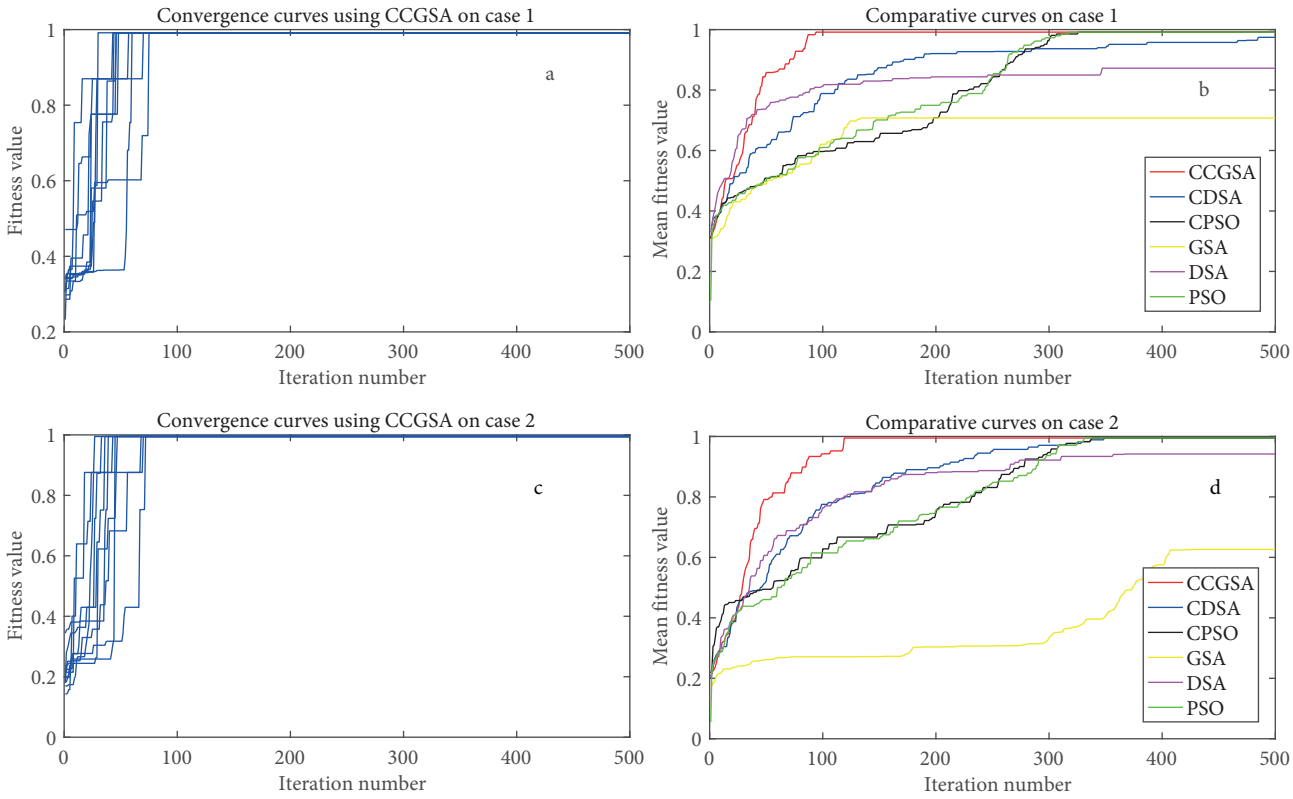

Figure 8. Convergence curves on two cases.

Table 2. Average cost time(s) of the comparative algorithms.

Case	PSO	CPSO	DSA	CDSA	GSA	CCGSA
Case 1	3.6479	3.5602	2.6759	1.7397	2.3659	1.5694
Case 2	4.2366	3.8034	3.1557	2.3979	8.6585	1.9030

and exploitation. In addition, the proposed algorithm demonstrates higher matching efficiency and robustness in template matching than the other selected competitors.

Future work will further investigate the influence of CCGSA on convergence performance with various chaotic maps and cloud models. The proposed algorithm will be applied to various optimization fields, such as multiobjective, multimodel, and binary optimization.

References

[1] Belan PA, Araujo SA, Librantz AFH. Segmentation-free approaches of computer vision for automatic calibration of digital and analog instruments. Measurement 2013; 46: 177-184.

[2] Liu RM, Lu YH, Gong CL, Liu Y. Infrared point target detection with improved template matching. Infrared Phys Techn 2012; 55: 380-387.

[3] Di Caterina G, Soraghan JJ. Robust complete occlusion handling in adaptive template matching target tracking. Electron Lett 2012; 48: 831-U48.

[4] Zhu SP, Li Z, Yu Y. Virtual view synthesis using stereo vision based on the sum of absolute difference. Comput Electr Eng 2014; 40: 236-246.

[5] Tachibana H, Uchida Y, Shiizuka H. Technical note: Determination of the optimized image processing and template matching techniques for a patient intrafraction motion monitoring system. Med Phys 2012; 39: 755-764.

[6] Yoo JC, Ahn CW. Template matching of occluded object under low PSNR. Digit Signal Process 2013; 23: 870-878.

[7] Debella-Gilo M, Kaab A. Sub-pixel precision image matching for measuring surface displacements on mass movements using normalized cross-correlation. Remote Sens Environ 2011; 115: 130-142.

[8] Cheng L, Li MC, Liu YX, Cai WT, Chen YM, Yang K. Remote sensing image matching by integrating affine invariant feature extraction and RANSAC. Comput Electr Eng 2012; 38: 1023-1032.

[9] Senoussi H, Saoudi A. A quadtree algorithm for template matching on a pyramid computer. Theor Comput Sci 1994; 136: 387-417.

[10] Schubert A, Faes A, Kaab A, Meier E. Glacier surface velocity estimation using repeat TerraSAR-X images: Wavelet- vs. correlation-based image matching. ISPRS J Photogramm 2013; 82: 49-62.

[11] Liu F, Duan HB, Deng YM. A chaotic quantum-behaved particle swarm optimization based on lateral inhibition for image matching. Optik 2012; 123: 1955-1960.

[12] Gan L, Duan HB. Biological image processing via Chaotic Differential Search and lateral inhibition. Optik 2014; 125: 2070-2075.

[13] Zhang QF, Duan HB. Chaotic biogeography-based optimization approach to target detection in UAV surveillance. Optik 2014; 125: 7100-7105.

[14] Rashedi E, Nezamabadi-pour H, Saryazdi S. GSA: A gravitational search algorithm. Inform Sciences 2009; 179: 2232-2248.

[15] Altinoz OT, Yilmaz AE, Weber GW. Improvement of the gravitational search algorithm by means of low-discrepancy sobol quasi random-number sequence based initialization. Adv Electr Comput En 2014; 14: 55-62.

[16] Xiang J, Han XH, Duan F, Qiang Y, Xiong XY, Lan Y, Chai HS. A novel hybrid system for feature selection based on an improved gravitational search algorithm and k-NN method. Appl Soft Comput 2015; 31: 293-307.

[17] Ju FY, Hong WC. Application of seasonal SVR with chaotic gravitational search algorithm in electricity forecasting. Appl Math Model 2013; 37: 9643-9651.

[18] Yuan XF, Liu YM, Xiang YZ, Yan XG. Parameter identification of BIPT system using chaotic-enhanced fruit fly optimization algorithm. Appl Math Comput 2015; 268: 1267-1281.

[19] Gandomi AH, Yang XS. Chaotic bat algorithm. J Comput Sci 2014; 5: 224-232.

[20] Zhao DS, He YZ. Chaotic binary bat algorithm for analog test point selection. Analog Integr Circ S 2015; 84: 201-214.

[21] Li DY, Meng HJ, Shi XM. Membership clouds and membership cloud generators. J Comput Res Dev 1995; 32: 15-20 (in Chinese with an abstract in English).

[22] Wang GY, Xu CL, Li DY. Generic normal cloud model. Inform Sciences 2014; 280: 1-15.

[23] Liang JJ, Suganthan PN, Deb K. Novel composition test functions for numerical global optimization. In: IEEE Swarm Intelligence Symposium; 8–10 June 2005; Pasadena, CA, USA: IEEE. pp. 68-75.

[24] Mirjalili S, Gandomi AH. Chaotic gravitational search constants for the gravitational search algorithm. Appl Soft Comput 2017; 53: 407-419.

[25] Guvenc U, Katırcıoglu F. Escape velocity: a new operator for gravitational search algorithm. Neural Comput Appl 2017; 1-16. DOI: 10.1007/s00521-017-2977-9.