HCS-BBD: An Effective Population-Based Approach for Multi-Level Thresholding

Seyed Jalaleddin Mousavirad
Department of Computer Engineering
Hakim Sabzevari University
Sabzevar, Iran

Gerald Schaefer
Department of Computer Science
Loughborough University
Loughborough, U.K.

Diego Oliva, Salvador Hinojosa
Depto. de Ciencias Computacionales
Universidad de Guadalajara
Guadalajara, Mexico

ABSTRACT

Thresholding is one of the most common techniques for image segmentation where an image is partitioned into several parts based on its histogram of pixel intensities. Conventional algorithms work efficiently for bi-level thresholding where an image is divided into fore- and background, but their efficiency drastically declines for the more complex case of multi-level thresholding due to the exhaustive search that is employed. To address this problem, in this paper we consider multi-level thresholding as an optimisation problem and propose a novel population-based algorithm, HCS-BBD, which is based on cuckoo search (CS) and biogeography-based optimisation (BBO). To this end, HCS-BBD integrates a heterogeneous cuckoo search strategy with a biogeography-based discovery operator. Our findings in comparison to state-of-the-art and recent population-based algorithms on different images convincingly demonstrate HCS-BBD's excellent capability in finding optimal threshold values.

CCS CONCEPTS

• **Computing methodologies → Bio-inspired approaches**.

KEYWORDS

Image thresholding, multi-level thresholding, optimisation, cuckoo search, biogeography-based optimisation.

ACM Reference Format:
Seyed Jalaleddin Mousavirad, Gerald Schaefer, and Diego Oliva, Salvador Hinojosa. 2021. HCS-BBD: An Effective Population-Based Approach for Multi-Level Thresholding. In *2021 Genetic and Evolutionary Computation Conference Companion (GECCO '21 Companion), July 10–14, 2021, Lille, France.* ACM, New York, NY, USA, 8 pages. https://doi.org/10.1145/3449726.3463149

1 INTRODUCTION

Thresholding, one of the most common image segmentation approaches due to its simplicity and accuracy [18, 20], uses the histogram of image intensities for segmentation. For simple cases, an object can be separated from the background by selecting an appropriate threshold which can be found in the valley between the two peaks in the histogram. However, most real images have multi-modal histograms, and finding multiple optimal threshold values is not an easy task [15].

Otsu's algorithm [21], one of the most widely employed image thresholding techniques, is based on the between-class variance. While it can work efficiently for bi-level thresholding, the computational time becomes infeasible when increasing the number of thresholds due to the exhaustive search that is employed. Population-based metaheuristic algorithms such as particle swarm optimisation (PSO) [8] or differential evolution (DE) [26] can be used as a reliable alternative to tackle this issue. According to the no free lunch theorem [27], there is no best algorithm to solve all optimisation problem. As a consequence, many researchers have focussed on employing population-based algorithms for multi-level image thresholding. For example, [1] proposes a DE-based image thresholding algorithm based on a mixture of Gaussian distribution, while [17] uses a center-based DE for high-dimensional multi-level image thresholding. In [29], PSO is employed for image thresholding based on a cross-entropy objective function. Other population-based algorithms that have been employed for multi-level image thresholding include dragonfly algorithm (DA) [4], sine cosine algorithm (SCA) [6], multi-verse optimiser (MVO) [7], cuckoo optimisation algorithm (COA) [14], self-organizing migrating algorithm (SOMA) [19], and human mental search (HMS) [16].

Cuckoo search (CS) [28] is an effective population-based algorithm inspired by the breeding behaviour of cuckoos. It employs two main operators, a Levy flight operator for updating each candidate solution, and a discovery operator. CS has also shown satisfactory performance for multi-level image thresholding [22]. Biogeography-based optimisation (BBO) [25] is another population-based optimisation algorithm, based on two main operators, mutation and migration. Generally, CS has advantages in global exploration, while BBO has a stronger exploitation ability [2]. Consequently, combining these two algorithms can benefit from both higher exploration and exploitation. In this paper, we present a novel image thresholding algorithm based on the combination of the two strategies, employing heterogeneous cuckoo search from the CS algorithm and a biogeography-based discovery operator. We employ a variance-based objective function and demonstrate excellent thresholding performance in comparison with other population-based algorithms.

The remainder of the paper is organised as follows. Section 2 briefly describes some background on CS and BBO. Section 3 presents our proposed algorithm, while Section 4 assesses it based on different metrics and in comparison with other algorithms. Section 5 concludes the paper.

GECCO '21 Companion, July 10–14, 2021, Lille, France

ACM ISBN 978-1-4503-8351-6/21/07…$15.00
https://doi.org/10.1145/3449726.3463149

2 BACKGROUND

2.1 Cuckoo search

Cuckoo search (CS) [28] is a population-based metaheuristic algorithm inspired by the breeding behaviour of cuckoos. In CS, each egg corresponds to a candidate solution $x_i = (x_{i,1}, x_{i,2}, ..., x_{i,N})$.

Each candidate solution is updated as

$$x_i^{new} = x_i^{old} + \alpha(x_i^{old} - x_g) \oplus Levy(\beta) = x_i^{old} + \frac{0.01u}{|v|^{\frac{1}{\beta}}}(x_i^{old} - x_g), \quad (1)$$

where $\oplus$ signifies entrywise multiplication, β is the Levy flight exponent, α is the step size, and x_g is the best candidate solution found so far, while u and v are two random random numbers calculated as

$$u \sim N(0, \sigma_u^2), \quad v \sim N(0, \sigma_v^2), \quad (2)$$

with

$$\sigma_u = \left\{ \frac{\Gamma(1+\beta)\sin(\frac{\pi\beta}{2})}{\Gamma[(\frac{1+\beta}{2})]\beta 2^{(\beta-1)/2}} \right\}^{1/\beta}, \quad \sigma_v = 1, \quad (3)$$

where Γ is a standard gamma function.

A discovery operator is employed which is defined as

$$x_{i,j}^{new} = \begin{cases} x_{i,j}^{old} + r.(x_{r1,j}(k) - x_{r2,j}(k)) & \text{if } P \geq pa \\ x_{i,j}^{old} & \text{otherwise} \end{cases}. \quad (4)$$

Algorithm 1 lists the CS algorithm in the form of pseudo-code.

Input : D: dimensionality of problem, $NFE_{\max}$: maximum number of function evaluations, N_P: population size
Output: $\overrightarrow{x^*}$: best solution

Generate the initial population Pop randomly
Evaluate the objective function for each candidate solution
$NFE = N_P$
while $NFE < NFE_{\max}$ **do**
  for $i \leftarrow 1$ **to** N_P **do**
    Generate a new candidate solution, x_i^{new}, using Eq. (1)
    Evaluate objective function for x_i^{new}
    if $f(x_i^{new})$ *is better than* $f(x_i^{old})$ **then**
      Replace old candidate solution x_i^{old} with x_i^{new}
    end
  end
  $NFE = NFE + N_p$
  for $i \leftarrow 1$ **to** N_P **do**
    for $j \leftarrow 1$ **to** D **do**
      Generate x_{ij}^{new} using Eq. (4)
    end
    Evaluate objective function for x_i^{new}
    if $f(x_i^{new})$ *is better than* $f(x_i^{old})$ **then**
      Replace old candidate solution x_i^{old} with x_i^{new}
    end
  end
  $NFE = NFE + N_p$
end
$x^* \leftarrow$ Best candidate solution in Pop

Algorithm 1: Pseudo-code of CS algorithm.

2.2 Biogeography-based optimisation

Biogeography-based optimisation (BBO) [25] is a population-based algorithm which simulates island biogeography for optimisation. Each candidate solution in BBO is called a habitat, and its quality corresponds to the habitat suitability index (HSI). A candidate solution can be seen as a set of suitability index variables (SIVs).

In each iteration, the population is sorted based on the objective function from best to worst and each candidate solution is assigned to an immigration rate $\lambda_K = (1 - \frac{K}{N_P})I$ and an emigration rate $\mu_K = \frac{K}{N_P}E$, where I and E are the maximum immigration and emigration rates (here $I = E = 1$), and K is the number of species of the habitats ($K = N_P - i$) with K for the best candidate solution being $N_P - 1$, K for the second best candidate solution $N_P - 2$, and K for the worst candidate solution 0.

BBO has two main operators, migration and mutation. The migration operator shares information from different candidate solutions, while the mutation operator alters a single candidate solution.

The BBO algorithm is given in pseudo-code form in Algorithm 2.

Input : D: dimensionality of problem, $NFE_{\max}$: maximum number of function evaluations, N_P: population size
Output: $\overrightarrow{x^*}$: the best solution

Generate the initial population Pop uniform randomly
Evaluate the objective function for each candidate solution
$NFE = N_P$
while $NFE < NFE_{\max}$ **do**
  Sort population from best to worst
  Assign each candidate solution an immigration rate λ_i and emigration rate μ_i
  for $i \leftarrow 1$ **to** N_P **do**
    for $j \leftarrow 1$ **to** D **do**
      if $rand < \lambda_i$ **then**
        Select a habitat x_k
        $x_{i,j}^{new} = x_{k,j}^{old}$
      else
        $x_{i,j}^{new} = x_{i,j}^{old}$
      end
    end
  end
  for $i \leftarrow 1$ **to** N_P **do**
    if $rand < p_i$ **then**
      Change x_i^{new} by mutation
    end
  end
  Evaluate objective function f(x_i^{new})
  $NFE = NFE + N_p$
  Conduct elitism stage
end
$x^* \leftarrow$ best candidate solution in Pop

Algorithm 2: Pseudo-code of BBO algorithm.

3 PROPOSED HCS-BBD ALGORITHM

Inspired by [2], in this paper, we propose a novel multi-level image thresholding algorithm based on a combination of BBO and CS. In the following, we first explain the components of our proposed HCS-BBD algorithm, and then detail its workings.

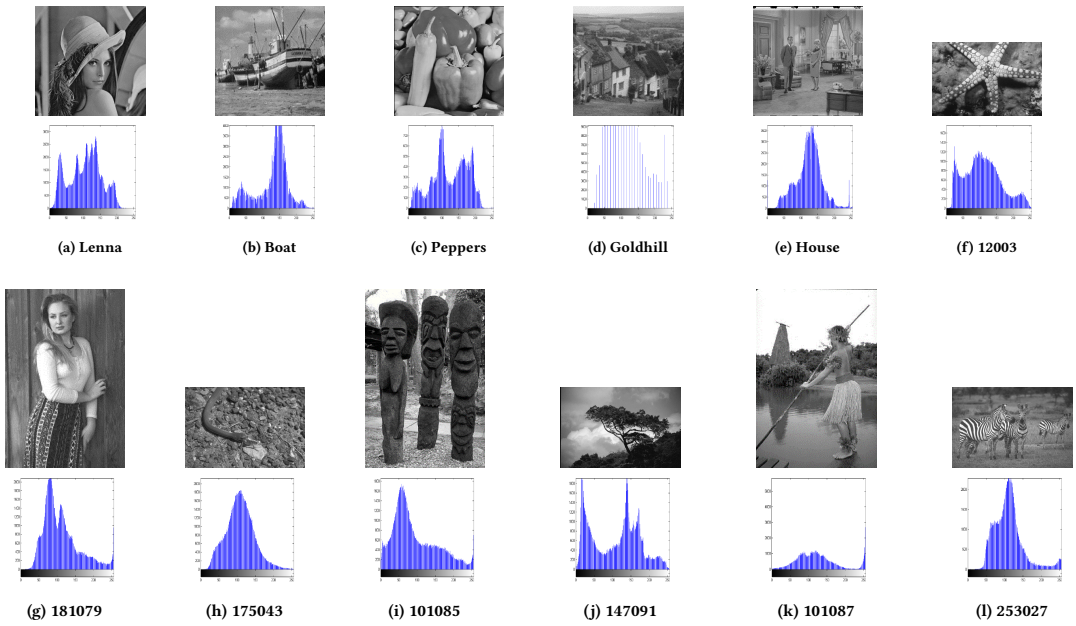

Figure 1: Test images and their histograms.

3.1 Heterogeneous CS strategy

We employ a heterogeneous CS strategy (HCS) [3, 5] which is based on a Levy flight distribution and a quantum mechanism. In HCS, a candidate solution is updated as

$$x_i^{new} = \begin{cases} x_i^{old} + \alpha.(x_i - x_j) \oplus Levy(\beta) & \text{if } \frac{2}{3} < sr \leq 1 \\ \bar{x_i} + L.(\bar{x_i} - x_i^{old}) & \text{if } \frac{1}{3} < sr \leq \frac{2}{3} \\ \bar{x_i} + \epsilon.(x_g - x_i^{old}) & \text{otherwise} \end{cases}, \quad (5)$$

where $L = \sigma ln(1/\lambda)$, $\epsilon = \sigma^{\lambda}$, x_g is the best candidate solution found so far, $\bar{x}$ is the average of all candidate solutions, and sr and λ are two random numbers between 0 and 1. The updating strategy thus has three components, one based on a Levy distribution, while the others are based on a quantum mechanism. This heterogeneous strategy can generate different candidate solutions with more diversification ability.

3.2 Biogeography-based discovery strategy

HCS-BBD uses a biogeography-based discovery (BBD) strategy to create a new candidate solution. To this end, the population is sorted from best to sort. Then, an emigration rate is assigned to each candidate solution as

$$\mu_k = \frac{k}{N_P} E. \quad (6)$$

The BBD strategy for creating a new candidate solution is described in Algorithm 3. Candidate solutions with better objective function values can share more features with others, leading to enhanced exploitation of the algorithm.

3.3 Representation

The representation determines how a candidate solution is encoded. In this paper, we employ a one-dimensional array $x = [t_1, t_2, ..., t_m]$ whose length is the number of threshold values, where t_i is the i-th threshold value. The upper and lower bounds of the search space are set as 0 and $2^n - 1$, respectively, along each dimension, where n is the number of bits representing a pixel.

Input : D: dimensionality of problem, x^{old}: candidate solution, P_a: discovery probability
Output: x^{new}: new candidate solution

for $j \leftarrow 1$ **to** D **do**
  if $rand > P_a$ **then**
    Select a candidate solution with probability $\propto \mu_k$
    Generate a random number α between 0 and 1
    $x_{i,j}^{new} = \alpha x_{i,j}^{old} + (1 - \alpha) x_{k,j}^{old}$
  else
    $x_{i,j}^{new} = x_{i,j}^{old}$
  end
end

Algorithm 3: BBD strategy.

3.4 Objective function

In this paper, we employ a variance-based objective function to assess each candidate solution. If the image pixels fall into L levels and the number of pixels in the i-th level is equal to n_i, then the normalised histogram is obtained as

$$p_i = n_i/N, p_i \geq 0, \sum_{i=1}^{L} p_i = 1. \quad (7)$$

In bi-level thresholding with threshold t, levels $[1, \ldots, t]$ levels will be allocated to class C_0 and levels $[t+1, \ldots, L]$ to class C_1. The optimal value for t is the one that maximises

$$\sigma_B^2 = \omega_0(\mu_0 - \mu_t)^2 + \omega_1(\mu_1 - \mu_t)^2, \quad (8)$$

where μ_t is the mean intensity of the entire image

$$\mu_t = \omega_0\mu_0 + \omega_1, \quad \omega_0 + \omega_1 = 1, \tag{9}$$

with

$$\mu_0 = \sum_{i=1}^{k} ip_i/\omega_0 \text{ and } \mu_1 = \sum_{i=k+1}^{L} ip_i/\omega_1. \tag{10}$$

This method can be extended for multi-level image thresholding to find m thresholds as $[t_1, t_2, \ldots, t_m]$ to partition the image into m classes: C_0 for $[0, \ldots, t_1 - 1]$, C_1 for $[t_1, \ldots, t_2 - 1]$, ..., C_m for $[t_m, \ldots, t_2 - 1]$. The objective function then becomes

$$J = \omega_0(\mu_0 - \mu_t)^2 + \omega_1(\mu_1 - \mu_t)^2 + \cdots + \omega_m(\mu_m - \mu_t)^2, \tag{11}$$

which is the one we use in this paper.

Input : D: dimensionality of problem, NFE_{max}: maximum number of function evaluations, N_P: population size, P_a: discovery probability
Output: $\overrightarrow{x^*}$: the best solution

Generate the initial population Pop randomly using encoding strategy from Section 3.3
Evaluate the objective function for each candidate solution using Eq. (11)
$NFE = N_P$
while $NFE < NFE_{max}$ **do**
  // HCS strategy
  for $i \leftarrow 1$ **to** N_P **do**
    Generate a new candidate solution, x_i^{new} by HCS strategy using Eq. (5)
    Evaluate the objective function $f(x_i^{new})$ using Eq. (11)
    if $f(x_i^{new})$ *is better than* $f(x_i^{old})$ **then**
      replace old candidate solution x_i^{old} with x_i^{new}
    end
  end
  $NFE = NFE + N_p$
  // BBD strategy
  Sort the population from best to worst
  Assign emigration rate μ to each candidate solution
  for $i \leftarrow 1$ **to** N_P **do**
    Generate a new candidate solution x_i^{new} using Algorithm 3
    Evaluate the objective function $f(x_i^{new})$ using Eq. (11)
    if $f(x_i^{new})$ *is better than* $f(x_i^{old})$ **then**
      replace old candidate solution x_i^{old} with x_i^{new}
    end
  end
  $NFE = NFE + N_p$
end
$x^* \leftarrow$ best candidate solution in Pop

Algorithm 4: HCS-BBD algorithm for multi-level image thresholding.

3.5 Algorithm

Our proposed HCS-BBD algorithm is designed to find optimal threshold values for a multi-level image thresholding problem. Algorithm 4 details the workings of HCS-BBD in form of pseudo-code.

4 EXPERIMENTAL RESULTS

We evaluate the performance of our proposed HCS-BBD algorithm and assess it against a set of other population-based competitors. To this end, we select five popular images, namely *Boats*, *Peppers*, *Goldhill*, *Lenna*, and *House*, as well as seven commonly employed images from the Berkley segmentation repository [9], namely *12003*, *181079*, *175043*, *101085*, *147091*, *101087*, and *253027*. Figure 1 shows the images together with their histograms. As we can observe, different images have different histogram characteristics. Some such as *Lenna* have several peaks and valleys, while others such as *101087* have a smoother distribution, whereas e.g. *Goldhill* has a histogram with abrupt changes.

We compare our algorithm with a number of other algorithms including DE [26], PSO [24], GSA [23], DA [10], SCA [11], MVO [13], SSA [12], BBO [25], and CS [28]. We select BBO and CS for comparison since our algorithm is based on these two. DE and PSO algorithms are state-of-the-art algorithms, while some others such as MVO and SSA are among the most recent approaches.

We set the number of thresholds to 5 and 10, and each algorithm is run 25 times. Statistical results including mean and standard

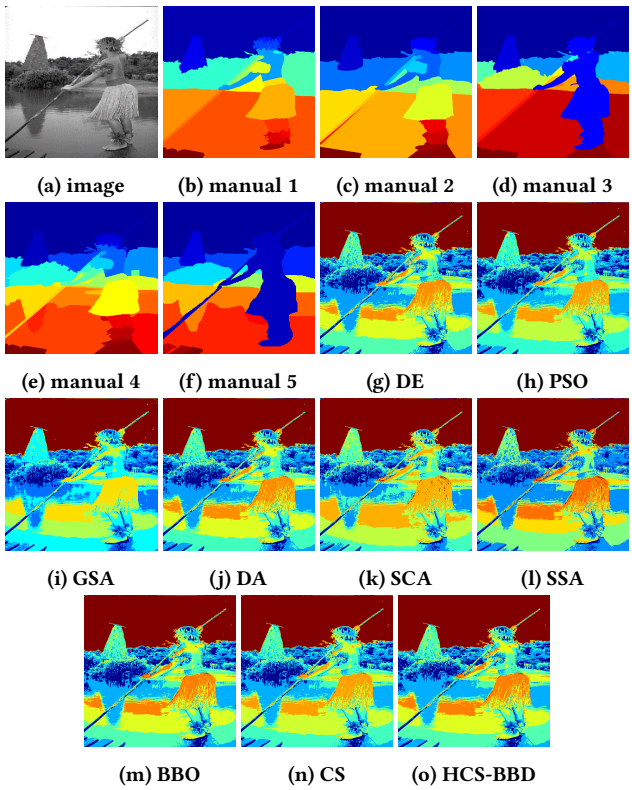

Figure 2: Thresholded *101087* images for $D = 5$ and all algorithms.

Table 1: Objective function results for all algorithms and images with $D = 5$.

image		DE	PSO	GSA	DA	SCA	MVO	SSA	BBO	CS	HCS-BBD
Boats	mean	2090.51	2091.33	2051.32	2091.59	2072.89	2090.11	2086.75	2092.14	2092.63	2092.58
	std.dev.	1.36	0.79	13.68	0.62	10.64	1.39	3.76	0.49	0.09	0.14
	rank	6	5	10	4	9	7	8	3	1	2
Peppers	mean	2734.53	2736.24	2688.08	2735.31	2711.20	2734.60	2727.84	2737.62	2737.89	2737.98
	std.dev.	2.64	1.06	22.34	7.65	11.62	1.71	5.60	0.45	0.21	0.12
	rank	7	4	10	5	9	6	8	3	2	1
Goldhill	mean	2336.97	2339.61	2291.43	2337.99	2320.71	2338.10	2336.06	2339.52	2340.16	2340.23
	std.dev.	2.25	0.58	23.37	7.16	13.52	1.22	3.21	1.03	0.25	0.00
	rank	7	3	10	6	9	5	8	4	2	1
Lenna	mean	2211.60	2214.10	2159.79	2213.88	2193.86	2213.18	2209.45	2215.04	2215.52	2215.79
	std.dev.	5.08	0.87	23.64	1.15	10.69	1.28	3.72	0.83	0.34	0.15
	rank	7	4	10	5	9	6	8	3	2	1
House	mean	1867.93	1870.14	1814.89	1870.51	1849.04	1868.56	1863.86	1870.97	1871.64	1871.76
	std.dev.	2.35	0.79	20.63	1.18	10.96	1.34	4.02	1.36	0.29	0.16
	rank	7	5	10	4	9	6	8	3	2	1
12003	mean	2908.91	2910.22	2859.46	2911.68	2885.84	2909.40	2902.54	2912.16	2912.52	2912.48
	std.dev.	1.97	1.22	21.29	1.01	16.17	1.75	4.91	0.48	0.19	0.20
	rank	7	5	10	4	9	6	8	3	1	2
181079	mean	2342.87	2344.33	2295.45	2345.36	2316.30	2342.76	2337.91	2346.55	2346.68	2346.71
	std.dev.	2.40	1.00	17.20	1.06	16.14	2.16	3.51	0.25	0.17	0.13
	rank	6	5	10	4	9	7	8	3	2	1
175043	mean	1313.81	1315.58	1271.83	1315.33	1296.08	1314.08	1311.03	1316.20	1316.64	1316.77
	std.dev.	2.16	0.62	12.28	1.26	11.34	1.05	2.12	0.73	0.22	0.18
	rank	7	4	10	5	9	6	8	3	2	1
101085	mean	3710.66	3710.82	3668.31	3713.60	3692.75	3710.43	3703.37	3714.04	3714.72	3714.58
	std.dev.	2.63	1.95	16.71	1.05	10.30	2.25	6.53	0.64	0.16	0.24
	rank	6	5	10	4	9	7	8	3	1	2
147091	mean	4127.85	4128.63	4082.37	4129.99	4105.02	4127.18	4119.42	4130.75	4131.20	4131.11
	std.dev.	1.95	1.29	19.46	0.93	13.78	1.88	5.34	0.45	0.12	0.24
	rank	6	5	10	4	9	7	8	3	1	2
101085	mean	5353.43	5354.01	5321.87	5354.84	5336.67	5353.11	5349.06	5355.26	5355.59	5355.70
	std.dev.	1.70	0.88	14.42	0.67	8.82	1.31	3.79	0.50	0.18	0.11
	rank	6	5	10	4	9	7	8	3	2	1
253027	mean	1607.39	1609.44	1568.16	1609.25	1591.69	1608.25	1606.73	1610.04	1610.49	1610.68
	std.dev.	1.89	0.83	18.57	1.45	11.39	1.36	2.85	0.64	0.28	0.09
	rank	7	4	10	5	9	6	8	3	2	1
average rank		6.58	4.50	10.00	4.50	9.00	6.33	8.00	3.08	1.67	1.33
overall rank		7	4.5	10	4.5	9	6	8	3	2	1

deviation are reported. The population size for all algorithms is set to 50, while the number of function evaluations (as stopping criterion) is set to 25,000. In our proposed algorithm, the habitat modification probability, maximum immigration rate, maximum emigration rate, and p_a are set to 1, 1, 1, and 0.25, respectively, while for other algorithms, we employed default settings from the cited publications.

First, we assess the performance of HCS-BBD visually on *101087* as a representative image and show the segmented images for all algorithms in Figure 2. Since the image is from the Berkeley segmentation repository [9], it comes with several (often quite different) manual segmentations, which are also shown in Figure 2. As we can see, HCS-BBD can yield a better segmentation which is for example noticeable in the lake area of the image which shows less noise for HCS-BBD compared to the other algorithms.

Table 1 shows the objective function results for $D = 5$. As we can see from there, HCS-BBD is top ranked for 8 of the 12 images, and ranked second for the others resulting in an average rank of 1.33, thus clearly giving the best performance of all methods. The average ranks for CS and BBO are 1.67 and 3.08, respectively. As is evident, our proposed method can lead to a significant improvement over both.

For $D = 10$, the results are given in Table 2. HCS-BBD ranks first for 11 of the 12 images, leading to a clear first overall rank. CS is ranked second, yet the margin between the average ranks of CS and HCS-BBD is wide.

Feature similarity index measure (FSIM) [30] is a popular image quality measure in the literature. Tables 3 and 4 show FSIM results for $D = 5$ and $D = 10$, respectively. As can be seen from there, for both cases HCS-BBD clearly outperforms all other algorithms.

Last not least, we perform a statistical comparison of HCS-BBD with the other algorithms. In particular, we conduct a Wilcoxon statistical signed rank test at 5% significance level whose results are given in Table 5. As we can see, in all cases and based on

Table 2: Objective function results for all algorithms and images with $D = 10$.

image		DE	PSO	GSA	DA	SCA	MVO	SSA	BBO	CS	HCS-BBD
Boats	mean	2143.96	2146.60	2124.61	2145.46	2125.86	2145.95	2143.65	2149.27	2150.61	2151.45
	std.dev.	3.51	1.48	9.11	3.17	4.87	2.07	2.30	1.33	1.05	0.57
	rank	7	4	10	6	9	5	8	3	2	1
Peppers	mean	2785.77	2788.29	2766.04	2786.31	2769.47	2787.93	2785.27	2791.46	2792.82	2793.71
	std.dev.	3.24	1.88	7.20	3.76	8.03	3.55	2.92	1.44	1.00	0.47
	rank	7	4	10	6	9	5	8	3	2	1
Goldhill	mean	2398.78	2400.78	2380.23	2397.80	2378.12	2400.02	2398.35	2403.54	2403.39	2405.74
	std.dev.	2.42	1.30	6.72	3.82	6.36	1.84	2.35	1.29	1.59	0.89
	rank	6	4	9	8	10	5	7	2	3	1
Lenna	mean	2257.22	2260.03	2233.29	2256.26	2243.14	2258.37	2256.77	2261.95	2262.43	2264.56
	std.dev.	2.36	1.45	11.87	4.53	4.92	1.76	2.14	1.20	1.30	0.63
	rank	6	4	10	8	9	5	7	3	2	1
House	mean	1931.57	1933.83	1908.33	1932.89	1913.33	1932.61	1930.08	1936.35	1937.55	1938.36
	std.dev.	2.81	1.69	8.72	2.92	8.58	2.28	2.31	1.68	0.68	0.39
	rank	7	4	10	5	9	6	8	3	2	1
12003	mean	2978.48	2981.24	2955.05	2978.26	2958.11	2978.95	2977.79	2983.72	2985.82	2986.26
	std.dev.	2.34	2.03	6.98	4.15	5.99	1.92	3.13	1.30	0.70	0.55
	rank	6	4	10	7	9	5	8	3	2	1
181079	mean	2402.08	2404.25	2382.04	2402.45	2384.93	2403.02	2401.28	2407.26	2408.33	2409.24
	std.dev.	2.59	1.80	9.73	4.30	6.42	2.25	3.37	1.22	0.94	0.55
	rank	7	4	10	6	9	5	8	3	2	1
175043	mean	1361.75	1363.69	1340.69	1360.51	1347.27	1362.79	1361.92	1365.37	1366.14	1367.56
	std.dev.	2.38	1.17	9.48	4.23	5.46	1.07	1.96	0.93	1.05	0.40
	rank	7	4	10	8	9	5	6	3	2	1
101085	mean	3803.32	3803.59	3777.99	3801.87	3786.02	3803.00	3798.19	3807.75	3810.51	3810.50
	std.dev.	3.48	1.97	9.84	4.25	6.50	3.00	3.37	1.26	0.76	0.65
	rank	5	4	10	7	9	6	8	3	1	2
147091	mean	4190.31	4191.83	4169.77	4189.45	4174.92	4190.83	4187.98	4195.50	4197.26	4197.97
	std.dev.	3.37	2.14	7.74	3.30	4.95	2.59	3.15	1.55	0.66	0.64
	rank	6	4	10	7	9	5	8	3	2	1
101085	mean	5404.37	5405.80	5390.91	5404.57	5391.97	5405.76	5403.52	5408.25	5409.29	5409.87
	std.dev.	2.15	1.19	5.05	3.11	4.07	1.46	2.07	0.86	0.61	0.37
	rank	7	4	10	6	9	5	8	3	2	1
253027	mean	1654.96	1657.94	1637.32	1655.25	1640.48	1656.84	1655.08	1660.07	1660.36	1661.79
	std.dev.	2.84	1.50	7.95	2.92	5.44	1.82	1.95	1.07	1.02	0.32
	rank	8	4	10	6	9	5	7	3	2	1
average rank		6.58	4.00	9.92	6.67	9.08	5.17	7.58	2.92	2.00	1.08
overall rank		6	4	10	7	9	5	8	3	2	1

both objective function and FSIM measures, the obtained p values are below 0.05, meaning that our proposed algorithm statistically significantly outperforms the other methods.

5 CONCLUSIONS

In this paper, we have proposed a novel multi-level image thresholding algorithm, HCS-BBD, based on cuckoo search (CS) and biogeography-based optimisation (BBO) leveraging the exploration ability of CS and the exploitation ability of BBO. Our HCS-BBD algorithm encodes the threshold values as candidate solution, and uses a variance-based objective function. Compared with nine other population-based optimisation algorithms, an extensive set of experiments show HCS-BBD to outperform them and to deliver excellent multi-level thresholding performance. In future work, we intend to extend the algorithm to other domains such as neural network training, while other objective functions are also under investigation.

REFERENCES

[1] M. Ali, C. W. Ahn, and M. Pant. 2014. Multi-level image thresholding by synergetic differential evolution. *Applied Soft Computing* 17 (2014), 1–11.

[2] X. Chen and K. Yu. 2019. Hybridizing cuckoo search algorithm with biogeography-based optimization for estimating photovoltaic model parameters. *Solar Energy* 180 (2019), 192–206.

[3] N. J. Cheung, X.-M. Ding, and H.-B. Shen. 2016. A nonhomogeneous cuckoo search algorithm based on quantum mechanism for real parameter optimization. *IEEE Transactions on Cybernetics* 47, 2 (2016), 391–402.

[4] M.-A. Díaz-Cortés, N. Ortega-Sánchez, S. Hinojosa, D. Oliva, E. Cuevas, R. Rojas, and A. Demin. 2018. A multi-level thresholding method for breast thermograms analysis using dragonfly algorithm. *Infrared Physics & Technology* 93 (2018), 346–361.

[5] X. Ding, Z. Xu, N. J. Cheung, and X. Liu. 2015. Parameter estimation of Takagi–Sugeno fuzzy system using heterogeneous cuckoo search algorithm. *Neurocomputing* 151 (2015), 1332–1342.

[6] S. Gupta and K. Deep. 2019. Improved sine cosine algorithm with crossover scheme for global optimization. *Knowledge-Based Systems* 165 (2019), 374–406.

[7] H. Jia, X. Peng, W. Song, C. Lang, Z. Xing, and K. Sun. 2019. Multiverse optimization algorithm based on Lévy flight improvement for multithreshold color image segmentation. *IEEE Access* 7 (2019), 32805–32844.

[8] J. Kennedy and R. Eberhart. 1995. Particle swarm optimization (PSO). In *IEEE International Conference on Neural Networks*. 1942–1948.

Table 3: FSIM results for all algorithms and images with $D = 5$.

image		DE	PSO	GSA	DA	SCA	MVO	SSA	BBO	CS	HCS-BBD
Boats	mean	0.9235	0.9241	0.8939	0.9242	0.9098	0.9234	0.9200	0.9241	0.9248	0.9249
	std.dev.	1.3596	0.7910	13.6796	0.6225	10.6442	1.3949	3.7584	0.4941	0.0850	0.1395
	rank	6	4.5	10	3	9	7	8	4.5	2	1
Peppers	mean	0.8341	0.8353	0.8076	0.8346	0.8171	0.8352	0.8292	0.8366	0.8367	0.8366
	std.dev.	2.6364	1.0646	22.3393	7.6469	11.6244	1.7127	5.5966	0.4505	0.2141	0.1178
	rank	7	4	10	6	9	5	8	2.5	1	2.5
Goldhill	mean	0.9221	0.9236	0.8915	0.9227	0.9105	0.9234	0.9210	0.9235	0.9247	0.9250
	std.dev.	2.2520	0.5825	23.3665	7.1649	13.5249	1.2199	3.2117	1.0295	0.2543	0.0000
	rank	7	3	10	6	9	5	8	4	2	1
Lenna	mean	0.8877	0.8906	0.8530	0.8914	0.8740	0.8909	0.8866	0.8916	0.8931	0.8935
	std.dev.	5.0806	0.8703	23.6370	1.1461	10.6932	1.2810	3.7242	0.8269	0.3389	0.1470
	rank	7	6	10	4	9	5	8	3	2	1
House	mean	0.9145	0.9159	0.8801	0.9159	0.9032	0.9153	0.9118	0.9164	0.9168	0.9164
	std.dev.	2.3529	0.7881	20.6265	1.1831	10.9551	1.3395	4.0172	1.3562	0.2851	0.1641
	rank	7	4.5	10	4.5	9	6	8	2.5	1	2.5
12003	mean	0.8210	0.8218	0.7786	0.8235	0.7987	0.8214	0.8155	0.8235	0.8241	0.8246
	std.dev.	1.9693	1.2159	21.2939	1.0079	16.1665	1.7538	4.9056	0.4752	0.1946	0.1961
	rank	7	5	10	3.5	9	6	8	3.5	2	1
181079	mean	0.8073	0.8077	0.7748	0.8074	0.7887	0.8051	0.8013	0.8079	0.8077	0.8081
	std.dev.	2.4047	0.9972	17.2015	1.0630	16.1414	2.1639	3.5066	0.2455	0.1732	0.1318
	rank	6	3.5	10	5	9	7	8	2	3.5	1
175043	mean	0.8989	0.8999	0.8635	0.9005	0.8842	0.8987	0.8955	0.9010	0.9010	0.9008
	std.dev.	2.1621	0.6168	12.2775	1.2641	11.3428	1.0498	2.1249	0.7312	0.2164	0.1751
	rank	6	5	10	4	9	7	8	1.5	1.5	3
101085	mean	0.8676	0.8669	0.8406	0.8692	0.8543	0.8671	0.8619	0.8688	0.8697	0.8695
	std.dev.	2.6279	1.9491	16.7055	1.0485	10.3028	2.2486	6.5287	0.6393	0.1642	0.2383
	rank	5	7	10	3	9	6	8	4	1	2
147091	mean	0.8431	0.8459	0.8210	0.8470	0.8276	0.8462	0.8385	0.8468	0.8477	0.8475
	std.dev.	1.9458	1.2934	19.4573	0.9271	13.7819	1.8750	5.3352	0.4532	0.1225	0.2421
	rank	7	6	10	3	9	5	8	4	1	2
101085	mean	0.8508	0.8515	0.8309	0.8522	0.8398	0.8509	0.8482	0.8523	0.8531	0.8534
	std.dev.	1.7010	0.8845	14.4158	0.6733	8.8160	1.3145	3.7877	0.4957	0.1820	0.1119
	rank	7	5	10	4	9	6	8	3	2	1
253027	mean	0.8768	0.8811	0.8460	0.8776	0.8641	0.8783	0.8761	0.8794	0.8816	0.8823
	std.dev.	1.8863	0.8254	18.5669	1.4494	11.3921	1.3634	2.8541	0.6417	0.2781	0.0947
	rank	7	3	10	6	9	5	8	4	2	1
average rank		6.58	4.71	10.00	4.33	9.00	5.83	8.00	3.21	1.75	1.58
overall rank		7	5	10	4	9	6	8	3	2	1

[9] D. Martin, C. Fowlkes, D. Tal, and J. Malik. 2001. A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics. In *8th International Conference on Computer Vision*, Vol. 2. 416–423.

[10] S. Mirjalili. 2016. Dragonfly algorithm: a new meta-heuristic optimization technique for solving single-objective, discrete, and multi-objective problems. *Neural Computing and Applications* 27, 4 (2016), 1053–1073.

[11] S. Mirjalili. 2016. SCA: a sine cosine algorithm for solving optimization problems. *Knowledge-Based Systems* 96 (2016), 120–133.

[12] S. Mirjalili, A. H. Gandomi, S. Z. Mirjalili, S. Saremi, H. Faris, and S. M. Mirjalili. 2017. Salp swarm algorithm: A bio-inspired optimizer for engineering design problems. *Advances in Engineering Software* 114 (2017), 163–191.

[13] S. Mirjalili, S. M. Mirjalili, and A. Hatamlou. 2016. Multi-verse optimizer: a nature-inspired algorithm for global optimization. *Neural Computing and Applications* 27, 2 (2016), 495–513.

[14] S. J. Mousavirad and H. Ebrahimpour-Komleh. 2015. Entropy based optimal multilevel thresholding using cuckoo optimization algorithm. In *11th International Conference on Innovations in Information Technology*. 302–307.

[15] S. J. Mousavirad and H. Ebrahimpour-Komleh. 2017. Multilevel image thresholding using entropy of histogram and recently developed population-based metaheuristic algorithms. *Evolutionary Intelligence* 10, 1-2 (2017), 45–75.

[16] S. J. Mousavirad and H. Ebrahimpour-Komleh. 2019. Human mental search-based multilevel thresholding for image segmentation. *Applied Soft Computing* (2019).

[17] S. J. Mousavirad, S. Rahnamayan, and G. Schaefer. 2020. Many-level image thresholding using a center-based differential evolution algorithm. In *Congress on Evolutionary Computation*.

[18] S. J. Mousavirad, G. Schaefer, and H. Ebrahimpour-Komleh. 2019. A benchmark of population-based metaheuristic algorithms for high-dimensional multi-level image thresholding. In *IEEE Congress on Evolutionary Computation*. 2394–2401.

[19] S. J. Mousavirad, G. Schaefer, and I. Korovin. 2020. High-dimensional multi-level image thresholding using self-organizing migrating algorithm. In *Genetic and Evolutionary Computation Conference Companion*. 1454–1459.

[20] S. J. Mousavirad, G. Schaefer, Z. Movahedi, and I. Korovin. 2020. High-dimensional multi-level maximum variance threshold selection for image segmentation: a benchmark of recent population-based metaheuristic algorithms. In *Genetic and Evolutionary Computation Conference Companion*. 1608–1613.

[21] N. Otsu. 1979. A threshold selection method from gray-level histograms. *IEEE Transactions on Systems, Man, and Cybernetics* 9, 1 (1979), 62–66.

[22] S. Pare, A. Kumar, V. Bajaj, and G. K. Singh. 2017. An efficient method for multilevel color image thresholding using cuckoo search algorithm based on minimum cross entropy. *Applied Soft Computing* 61 (2017), 570–592.

[23] E. Rashedi, H. Nezamabadi-Pour, and S. Saryazdi. 2009. GSA: a gravitational search algorithm. *Information Sciences* 179, 13 (2009), 2232–2248.

[24] Y. Shi and R. Eberhart. 1998. A modified particle swarm optimizer. In *IEEE International Conference on Evolutionary Computation*. 69–73.

Table 4: FSIM results for all algorithms and images with $D = 10$.

image		DE	PSO	GSA	DA	SCA	MVO	SSA	BBO	CS	HCS-BBD
Boats	mean	0.9676	0.9715	0.9510	0.9707	0.9517	0.9706	0.9686	0.9744	0.9760	0.9769
	std.dev.	3.5109	1.4763	9.1107	3.1711	4.8727	2.0691	2.3007	1.3263	1.0526	0.5749
	rank	8	4	10	5	9	6	7	3	2	1
Peppers	mean	0.9048	0.9090	0.8787	0.9066	0.8810	0.9078	0.9064	0.9153	0.9186	0.9203
	std.dev.	3.2401	1.8804	7.2028	3.7551	8.0281	3.5481	2.9223	1.4408	1.0022	0.4667
	rank	8	4	10	6	9	5	7	3	2	1
Goldhill	mean	0.9701	0.9703	0.9527	0.9694	0.9522	0.9698	0.9679	0.9732	0.9739	0.9745
	std.dev.	2.4151	1.3026	6.7227	3.8174	6.3623	1.8436	2.3540	1.2886	1.5876	0.8949
	rank	5	4	9	7	10	6	8	3	2	1
Lenna	mean	0.9447	0.9491	0.9169	0.9445	0.9293	0.9478	0.9438	0.9529	0.9546	0.9561
	std.dev.	2.3623	1.4477	11.8650	4.5304	4.9243	1.7553	2.1422	1.2009	1.2960	0.6270
	rank	6	4	10	7	9	5	8	3	2	1
House	mean	0.9656	0.9686	0.9473	0.9693	0.9498	0.9675	0.9637	0.9714	0.9733	0.9728
	std.dev.	2.8134	1.6877	8.7225	2.9236	8.5777	2.2827	2.3143	1.6848	0.6844	0.3917
	rank	7	5	10	4	9	6	8	3	1	2
12003	mean	0.9143	0.9189	0.8786	0.9132	0.8824	0.9139	0.9138	0.9235	0.9274	0.9280
	std.dev.	2.3359	2.0293	6.9807	4.1517	5.9884	1.9237	3.1299	1.2991	0.6955	0.5489
	rank	5	4	10	8	9	6	7	3	2	1
181079	mean	0.9026	0.9072	0.8698	0.9026	0.8719	0.9034	0.9004	0.9120	0.9139	0.9168
	std.dev.	2.5941	1.7952	9.7276	4.2976	6.4241	2.2517	3.3670	1.2172	0.9393	0.5534
	rank	6.5	4	10	6.5	9	5	8	3	2	1
175043	mean	0.9571	0.9597	0.9351	0.9557	0.9386	0.9592	0.9553	0.9623	0.9647	0.9644
	std.dev.	2.3802	1.1668	9.4810	4.2312	5.4629	1.0748	1.9561	0.9323	1.0524	0.3994
	rank	6	4	10	7	9	5	8	3	1	2
101085	mean	0.9434	0.9439	0.9249	0.9436	0.9279	0.9441	0.9367	0.9473	0.9504	0.9492
	std.dev.	3.4770	1.9715	9.8360	4.2537	6.5007	2.9961	3.3685	1.2586	0.7595	0.6535
	rank	7	5	10	6	9	4	8	3	1	2
147091	mean	0.8993	0.8987	0.8845	0.8983	0.8862	0.9017	0.8964	0.9042	0.9067	0.9074
	std.dev.	3.3711	2.1365	7.7430	3.3031	4.9543	2.5855	3.1521	1.5523	0.6589	0.6445
	rank	5	6	10	7	9	4	8	3	2	1
101085	mean	0.9235	0.9261	0.9049	0.9245	0.9052	0.9262	0.9227	0.9307	0.9324	0.9330
	std.dev.	2.1522	1.1852	5.0459	3.1085	4.0689	1.4614	2.0673	0.8629	0.6135	0.3726
	rank	7	5	10	6	9	4	8	3	2	1
253027	mean	0.9400	0.9461	0.9151	0.9407	0.9194	0.9442	0.9400	0.9500	0.9506	0.9525
	std.dev.	2.8448	1.5014	7.9451	2.9239	5.4422	1.8226	1.9532	1.0703	1.0195	0.3205
	rank	7.5	4	10	6	9	5	7.5	3	2	1
average rank		6.50	4.42	9.92	6.29	9.08	5.08	7.71	3.00	1.75	1.25
overall rank		7	4	10	6	9	5	8	3	2	1

[25] D. Simon. 2008. Biogeography-based optimization. *IEEE Transactions on Evolutionary Computation* 12, 6 (2008), 702–713.
[26] R. Storn and K. Price. 1997. Differential evolution–a simple and efficient heuristic for global optimization over continuous spaces. *Journal of Global Optimization* 11, 4 (1997), 341–359.
[27] D. Wolpert and G. Macready. 1997. No free lunch theorems for optimization. *IEEE Transactions on Evolutionary Computation* 1, 1 (1997), 67–82.
[28] X.-S. Yang and S. Deb. 2009. Cuckoo search via Lévy flights. In *World Congress on Nature & Biologically Inspired Computing*. 210–214.
[29] P.-Y. Yin. 2007. Multilevel minimum cross entropy threshold selection based on particle swarm optimization. *Appl. Math. Comput.* 184, 2 (2007), 503–513.
[30] L. Zhang, L. Zhang, X. Mou, and D. Zhang. 2011. FSIM: A feature similarity index for image quality assessment. *IEEE Transactions on Image Processing* 20, 8 (2011), 2378–2386.

Table 5: Results of Wilcoxon signed rank test.

	p value	
	objective function	FSIM
HCS-BBD vs. DE	1.8215e-05	1.8108e-05
HCS-BBD vs. PSO	1.8215e-05	1.8197e-05
HCS-BBD vs. GSA	1.8215e-05	1.8197e-05
HCS-BBD vs. DA	1.8215e-05	1.8162e-05
HCS-BBD vs. SCA	1.8215e-05	1.8197e-05
HCS-BBD vs. MVO	1.8215e-05	1.8197e-05
HCS-BBD vs. SSA	1.8215e-05	1.8126e-05
HCS-BBD vs. BBO	1.8215e-05	4.9112e-05
HCS-BBD vs. CS	3.5504e-04	0.0128