Multi-Strategy MO-K-Means Method for Cluster-Center Optimization in Color Image Segmentation

Haoqin Chen[1], Weibin Kong[2]*, Zhongqing Fang[3], Xiaotian Chen[4], Xin Shen[5], Tiantian Song[6]
College of Information Engineering, Yancheng Institute of Technology, Jiangsu Yancheng 224051, China[1, 2, 3, 4, 5, 6]
Jiangsu Provincial Engineering Technology Center, Multimodal Perception and Intelligent Control of Offshore Wind Power Systems, Yancheng 224051, China[2]

***Abstract*—Color image segmentation is a fundamental problem in image processing and computer vision. K-means is widely used for clustering-based color image segmentation, but its results are sensitive to the initial cluster centers. In addition, clustering based only on RGB features does not directly use the spatial relationships between pixels. To improve cluster-center search, this study proposes a method that combines a multi-strategy Musk Ox Optimizer (MO) with weighted K-means, referred to as Multi-Strategy MO-K-means. The image is divided into super-pixels using Simple Linear Iterative Clustering (SLIC). The mean RGB vector and pixel count of each superpixel are used as the clustering feature and weight, respectively. Multi-Strategy MO then searches for RGB cluster centers, which are used to initialize weighted K-means. After the centers are updated, the final cluster labels are assigned at the superpixel level. Experiments on the BSDS500 test set show that Multi-Strategy MO reduced SearchSSE by 16.29%–20.11% compared with the four comparison optimizers under the same evaluation budget. Compared with K-means++, the proposed method achieved a 20.41% relative improvement in Boundary-F and produced more coherent segmentation regions.**

***Keywords*—Color image segmentation; K-means clustering; Musk Ox Optimizer; cluster-center optimization; SLIC superpixels**

I. INTRODUCTION

Image segmentation is used to divide an image into several regions with similar characteristics according to features such as color and texture. It is a fundamental process in image processing and has been widely applied in medical image analysis, remote sensing, and other fields [1]. Compared with grayscale images, color images provide richer color information, which can be used to distinguish different regions more effectively. However, segmentation is made more difficult by complex backgrounds, illumination variations, and color differences within the same region. Without requiring large amounts of manually annotated data, clustering methods can be used to group pixels according to the similarity of their features and have therefore been widely applied to unsupervised color image segmentation. Among these methods, K-means is widely used because of its simple structure, ease of implementation, and computational efficiency [2]. However, its performance is sensitive to the selection of the initial cluster centers, and poor initialization may cause the algorithm to converge to a suboptimal local solution [3].

To address limitations in K-means initialization and the use of spatial information, previous studies have explored different improvements to clustering-based color image segmentation. Kumar et al. identified high-density cluster cores to determine candidate cluster centers and improve K-means initialization [4]. For spatial information, superpixels group spatially connected pixels with similar characteristics into regional units, providing a region-level representation for clustering [5], [6]. Building on this idea, Xie et al. combined superpixel merging with a multi-feature adaptive fusion metric and extended clustering to the superpixel level [7]. Ji et al. used an improved version of Simple Linear Iterative Clustering (SLIC) to generate adaptive superpixels and introduced color and local spatial information into the clustering process [8]. Chen et al. represented each superpixel using its regional mean feature and pixel count for superpixel-level clustering [9]. Wu et al. incorporated superpixel generation and fuzzy sparse subspace clustering into a unified learning framework [10]. Beyond superpixel representations, Zhu et al. improved C-means clustering through distance measurement and feature weighting [11]. Wu et al. introduced local and non-local spatial information together with sparse regularization to reduce the effect of noise on clustering results [12]. These studies provide different approaches to improving clustering-based color image segmentation through cluster-center initialization, region-level representation, and the use of spatial information.

In addition to improvements in clustering initialization and spatial representation, metaheuristic algorithms have also been used for search and optimization in image segmentation. For clustering-based image segmentation, population-based metaheuristics can optimize clustering configurations through search, while local search can be introduced to improve the search process [13]. To improve search capability, different strategies have been incorporated into population-based optimization algorithms for image segmentation. Hu et al. introduced population reorganization and a mutation-based replacement mechanism into particle swarm optimization and applied the method to multi-threshold segmentation of pathological images [14]. For clustering-based segmentation, Dey et al. used a quantum-inspired swarm intelligence algorithm for automatic clustering of color images, with RGB cluster centers determined through the optimization process [15]. Zhang et al. adopted a multi-population structure and assigned differential mutation, multiple crossover, and best-individual guidance to different subpopulations. The improved algorithm was then applied to image segmentation based on fast and robust fuzzy C-means (FRFCM) clustering [16]. Ma et al. introduced opposition-based learning and vertical crossover into the Manta Ray Foraging Optimizer to improve the balance between exploration and exploitation and applied it to multi-threshold

*Corresponding author.

color image segmentation [17]. Mousavirad et al. compared 23 population-based metaheuristic algorithms and found that newly proposed algorithms do not necessarily outperform existing classical algorithms [18]. These studies indicate that the search mechanisms of metaheuristic algorithms should be designed according to the specific optimization problem.

To improve the search behavior of metaheuristic algorithms, previous studies have combined different search strategies. Seyyedabbasi proposed the hybrid multi-strategy optimizer COSGO and applied it to multilevel thresholding for color image segmentation [19]. Yuan et al. proposed the Musk Ox Optimizer (MO) [20]. It simulates the migration, foraging, and defense behaviors of musk ox groups. MO selects different update modes according to warning and safety signals. The population is then updated through iteration. For K-means-based color image segmentation, a set of RGB cluster centers can be represented as one candidate solution in a continuous search space. Therefore, a metaheuristic algorithm can be used to search for suitable cluster centers.

To address these problems, a Multi-Strategy MO-K-means method is proposed for color image segmentation. K-means is sensitive to the initial cluster centers, while RGB-based clustering makes limited use of the spatial relationships between pixels. First, the image is divided into superpixels using SLIC. The mean RGB vector and pixel count of each superpixel are then used to construct the clustering data. RGB cluster centers are searched by Multi-Strategy MO and are subsequently used to initialize weighted K-means. After the cluster centers are updated, the final labels are assigned at the superpixel level. The proposed method is evaluated in terms of cluster-center search performance and segmentation results.

II. K-Means-Based Color Image Segmentation

K-means partitions data by alternating between sample assignment and cluster-center update. In color image segmentation, each pixel is represented by its red, green, and blue components. Each cluster center represents a color prototype. Assume that the processed image contains N pixels. For $n = 1, 2, ..., N$, the n-th pixel is represented as follows:

$$\mathbf{p}_n = \left(p_{n,\mathrm{R}}, p_{n,\mathrm{G}}, p_{n,\mathrm{B}} \right)^{\mathrm{T}} \in [0,255]^3 \tag{1}$$

where, $\mathbf{p}_n$ denotes the RGB vector of the n-th pixel, and $p_{n,R}$, $p_{n,G}$, and $p_{n,B}$ denote its red, green, and blue components, respectively.

For K clusters, the set of RGB cluster centers is defined as follows:

$$\begin{aligned} &\mathrm{C} = \{\mathbf{c}_1, \mathbf{c}_2, \ldots, \mathbf{c}_K\}, \\ &\mathbf{c}_k = \left(c_{k,\mathrm{R}}, c_{k,\mathrm{G}}, c_{k,\mathrm{B}} \right)^{\mathrm{T}} \in [0,255]^3 \end{aligned} \tag{2}$$

where, C denotes the set of cluster centers. c_k denotes the k-th RGB cluster center. $c_{k,R}$, $c_{k,G}$, and $c_{k,B}$ denote its red, green, and blue components, respectively.

Given the cluster centers, each pixel is assigned to the nearest center according to the squared Euclidean distance:

$$q_n = \arg\min_{1 \le k \le K} \left\| \mathbf{p}_n - \mathbf{c}_k \right\|_2^2 \tag{3}$$

where, q_n denotes the cluster label of the n-th pixel. If several cluster centers have the same minimum distance, the center with the smallest index is selected. The set of pixel indices in the k-th cluster, I_k, is defined as:

$$\mathrm{I}_k = \{ n | q_n = k, n = 1, 2, \ldots, N \} \tag{4}$$

For each nonempty cluster, its cluster center is updated to the mean RGB vector of the pixels in that cluster:

$$\mathbf{c}_k \leftarrow \frac{1}{|\mathrm{I}_k|} \sum_{n \in \mathrm{I}_k} \mathbf{p}_n, \qquad |\mathrm{I}_k| > 0 \tag{5}$$

where, $|\mathrm{I}_k|$ denotes the number of pixels in the k-th cluster.

In standard RGB clustering, the clustering error is defined as the sum of squared distances from all pixels to their nearest cluster centers:

$$J_{\mathrm{RGB}}(\mathrm{C}) = \sum_{n=1}^{N} \min_{1 \le k \le K} \left\| \mathbf{p}_n - \mathbf{c}_k \right\|_2^2 \tag{6}$$

where, $J_{RGB}(\mathrm{C})$ denotes the RGB clustering error. For fixed cluster centers, each pixel is assigned to its nearest center according to Eq. (3). With the pixel assignments fixed, the center of each nonempty cluster is updated using the mean in Eq. (5), which minimizes the within-cluster sum of squared errors (SSE). K-means alternates between these two steps.

K-means is sensitive to the initial cluster centers, and different initializations may lead to different local solutions. In the proposed method, the image is first divided into superpixels using SLIC. The mean RGB vector and pixel count of each superpixel are then used to construct the weighted clustering objective. Multi-Strategy MO is used to search for RGB cluster centers, and the resulting centers are used to initialize weighted K-means.

III. Proposed Multi-Strategy MO-K-Means Method

The proposed method combines SLIC, Multi-Strategy MO, and weighted K-means for color image segmentation. First, SLIC divides the input image into superpixels. The mean RGB vector and pixel count of each superpixel are used to construct the clustering data. Multi-Strategy MO then searches for RGB cluster centers. The searched centers are used to initialize weighted K-means. After the cluster centers are updated, each superpixel is assigned a final cluster label. The label is then assigned to all pixels in that superpixel. The overall procedure is shown in Fig. 1.

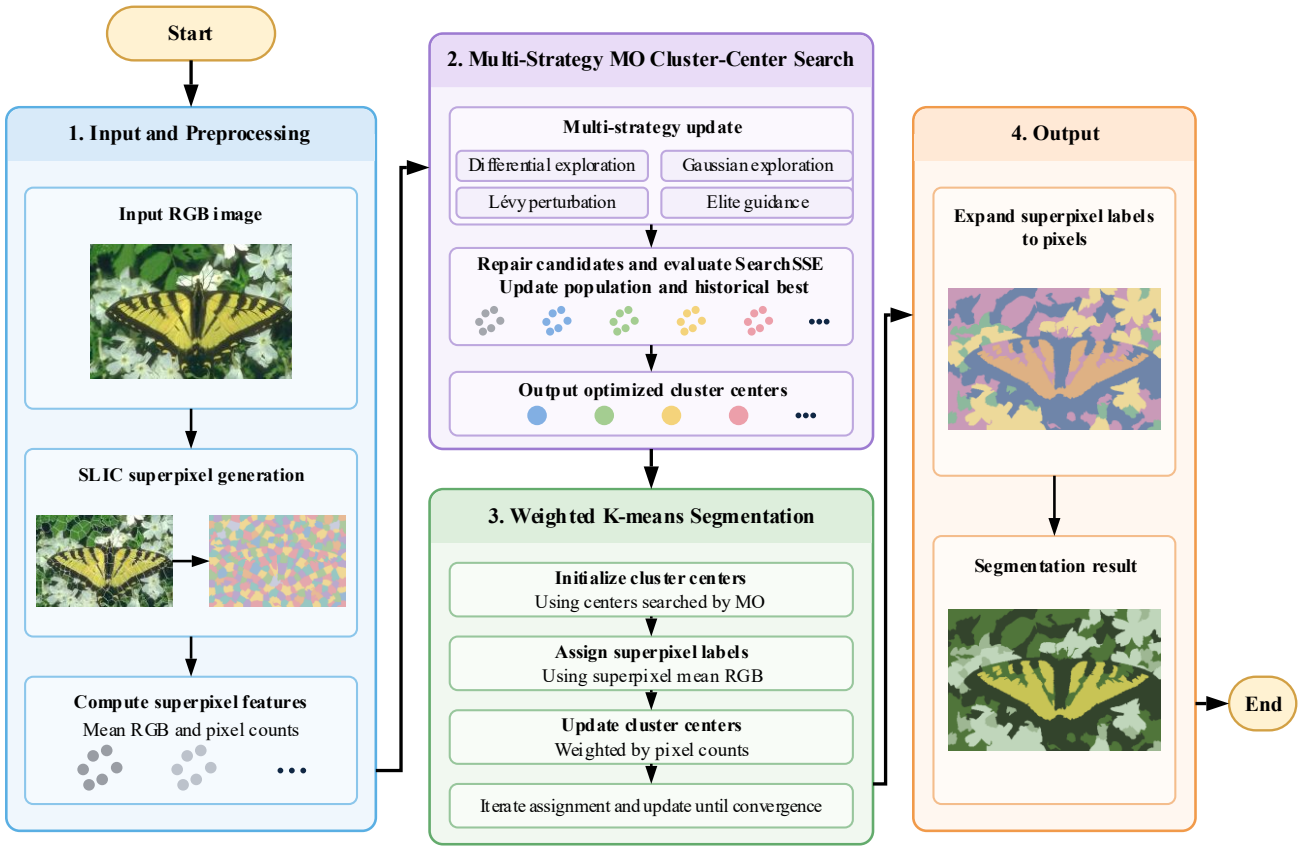

Fig. 1. Flowchart of the proposed multi-strategy MO-K-means method.

A. SLIC Superpixels

To use spatial information, SLIC first divides the input RGB image into superpixels. Let M denote the number of generated superpixels. Let Ω_s denote the set of pixels in the s-th superpixel. Its pixel count n_s and mean RGB vector $\boldsymbol{\mu}_s$ are defined as follows:

$$n_s = |\Omega_s|$$
$$\boldsymbol{\mu}_s = \frac{1}{n_s} \sum_{n \in \Omega_s} \mathbf{p}_n, s = 1,\ldots,M \tag{7}$$

where, $\mathbf{p}_n$ denotes the RGB vector of n-th pixel. n_s denotes the number of pixels in the s-th superpixel, and $\boldsymbol{\mu}_s$ denotes its mean RGB vector.

In the following clustering process, $\boldsymbol{\mu}_s$ represents the color information of the s-th superpixel, and n_s is used as its weight. For the cluster-center set C, the weighted clustering objective is defined as follows:

$$J_{SLIC}(\mathrm{C}) = \sum_{s=1}^{M} n_s \min_{1 \le k \le K} \|\boldsymbol{\mu}_s - \mathbf{c}_k\|_2^2 \tag{8}$$

where, $J_{SLIC}(\mathrm{C})$ denotes the weighted clustering objective based on the superpixels. c_k denotes the k-th RGB cluster center. Multi-Strategy MO uses this objective to evaluate candidate cluster centers. The searched centers are then used to initialize weighted K-means.

B. Multi-Strategy MO

Multi-Strategy MO uses the weighted clustering objective in Eq. (8) to search for RGB cluster centers. The method is built on the migration, foraging, and defense update framework of MO. It also introduces differential, Gaussian, and Lévy exploration, together with historical-best and elite guidance.

1) Candidate representation: Let P denote the population size and T the number of updates. At the beginning of the t-th update, the i-th candidate represents a complete set of K RGB cluster centers:

$$\mathbf{X}_i^{(t)} = \left[\left(\mathbf{c}_{i,1}^{(t)}\right)^{\mathrm{T}}, \ldots, \left(\mathbf{c}_{i,K}^{(t)}\right)^{\mathrm{T}} \right]$$
$$D = 3K, \qquad \mathbf{X}_i^{(t)} \in [0,255]^D \tag{9}$$

where, $X_i^{(t)}$ denotes the position of the i-th individual at the beginning of the t-th update. $c_{i,k}^{(t)}$ denotes the k-th RGB cluster center encoded by this individual. D denotes the dimension of the search space.

The initial population is generated from the distinct RGB colors present in the input image. Each candidate randomly selects K different colors as the initial cluster centers. Each candidate is evaluated using the objective function in Eq. (8).

2) Basic MO update: The basic MO uses warning and safety signals to select migration, foraging, or defense at each update. Let $t = 1,\ldots,T$ denote the update index. $X_i^{(t)}$ denotes the position of the i-th individual at the beginning of the t-th update. $G^{(t)}$ denotes the historical best position found before the current update. $S_2^{(t)}$ and $W^{(t)}$ denote the second-ranked and worst positions in the current population, respectively. The basic candidate position of the i-th individual is denoted by $B_i^{(t)}$.

a) Warning and safety signals: The warning signal determines whether the current population performs the migration update. When the warning level is low, the safety signal is used to choose between foraging and defense. The warning and safety signals at the t-th update are defined as follows:

$$\delta_t = 2\left(1-\frac{t-1}{T}\right)\left(1-r_{1,t}\right)$$
$$\eta_t = r_{2,t} \tag{10}$$

where, δ_t and η_t denote the warning and safety signals, respectively. $r_{1,t}$ and $r_{2,t}$ are two independent random numbers drawn from $U(0,1)$. Both random numbers are generated once at each update and are shared by all individuals in the current population.

When $|\delta_t| \geq 1$, the current population performs the migration update. When $|\delta_t| < 1$, the foraging update is performed if $\eta_t \geq 0.5$. Otherwise, the defense update is performed.

b) Migration update: When the warning signal meets the migration condition, each individual uses the difference between two population positions to obtain a new search direction. The logistic coefficient β_t is first calculated. The basic candidate position of the i-th individual is then obtained as follows:

$$\beta_t = 1-\frac{1}{1+\exp\left(10\frac{0.5T-t}{T}\right)}$$
$$\mathbf{B}_i^{(t)} = \mathbf{X}_i^{(t)} + \beta_t^2 r_{3,t}^4 \times \left(\mathbf{X}_{\pi_t(i)}^{(t)} - \mathbf{X}_{\sigma_t(i)}^{(t)}\right) \tag{11}$$

where, $r_{3,t} \sim U(0,1)$ and is generated once for the current migration update. $\pi_t(i)$ and $\sigma_t(i)$ denote the individual indices at position i after two independent random permutations of the population indices. Eq. (11) uses the difference between two population positions to move the current individual. All positions on the right-hand side are taken from the population at the beginning of the current update.

c) Safe-foraging update: When the warning signal is low and the safety signal meets the foraging condition, each individual uses the historical best position as the main reference. The foraging weight $w_{f,t}$ and the basic candidate position of the i-th individual are defined as follows:

$$w_{f,t} = 0.001\exp(-0.01\frac{t}{T})$$
$$\mathbf{B}_i^{(t)} = \mathbf{G}^{(t)} + q_{i,t}^3 w_{f,t}\left(\mathbf{X}_i^{(t)} - \mathbf{G}^{(t)}\right) \tag{12}$$

where, $q_{i,t} \sim U(0,1)$ and is generated independently for each individual. This update uses the historical best position $G^{(t)}$ as the reference. The basic candidate position is generated from the difference between the current position and the historical best position.

d) Defense update: When the warning signal is low and the safety signal does not meet the foraging condition, the population enters the defense update. The population is divided into two non-overlapping groups. Different update rules are used for the two groups. The number of individuals in the first group is defined as follows:

$$m = \max\left\{1, \text{round}\left(0.4P\right)\right\} \tag{13}$$

where, m denotes the number of individuals in the first group, and P denotes the population size. The first group contains individuals 1 to m. The second group contains individuals $m+1$ to P.

For the first group, the current second-ranked position, the current worst position, and the historical best position are used to generate the basic candidate position:

$$w_{d,t} = 100\exp\left(-0.01\frac{t}{T}\right)$$
$$\mathbf{B}_i^{(t)} = \mathbf{X}_i^{(t)} + a_{i,t}^2\left(\mathbf{S}_2^{(t)} - \mathbf{G}^{(t)}\right) + b_{i,t} w_{d,t}\left(\mathbf{G}^{(t)} - \mathbf{W}^{(t)}\right),$$
$$i = 1,\ldots,m \tag{14}$$

where, $w_{d,t}$ denotes the defense weight at the t-th update. $a_{i,t}$ and $b_{i,t}$ are generated independently for the i-th individual and follow $U(0,1)$.

For the second group, the basic candidate position is generated using the difference between the current position and the historical best position together with a Lévy random vector:

$$\mathbf{B}_i^{(t)} = \mathbf{X}_i^{(t)} + c_{i,t} \times \left[\left(\mathbf{G}^{(t)} - \mathbf{X}_i^{(t)} \right) \otimes \mathbf{L}_i^{(t)} \right], \quad i = m+1, \ldots, P \tag{15}$$

where, $c_{i,t}$ is independently drawn from $U(0,1)$ for each individual, and $\otimes$ denotes element-wise multiplication. $\mathbf{L}_i^{(t)} \in \mathbb{R}^D$ denotes a newly generated D-dimensional Lévy random vector.

e) Lévy vector generation: Lévy random vectors are used in the second group of the defense update and in the later Lévy exploration. The vectors are generated using Mantegna's method. The stability index is fixed at $\nu = 3/2$, and the corresponding Gaussian scale σ_ν is defined as follows:

$$\sigma_\nu = \left[\frac{\Gamma(1+\nu)\sin(\pi\nu/2)}{\Gamma\left((1+\nu)/2\right)\nu 2^{(\nu-1)/2}} \right]^{1/\nu} \tag{16}$$

where, $\Gamma(\cdot)$ denotes the Gamma function.

For a D-dimensional Lévy random vector, the j-th component is generated from two independent Gaussian random variables:

$$L_{i,j}^{(t)} = \frac{\xi_{i,j}^{(t)}}{\left[\max\left(\left| \zeta_{i,j}^{(t)} \right|, \varepsilon_0 \right) \right]^{1/\nu}}, \qquad j = 1, \ldots, D \tag{17}$$

where, $\xi_{i,j}^{(t)} \sim N(0, \sigma_\nu^2)$ and $\zeta_{i,j}^{(t)} \sim N(0,1)$. The constant ε_0 is set to MATLAB's double-precision realmin to avoid division by zero. They are mutually independent and are sampled independently across different search dimensions. $L_{i,j}^{(t)}$ denotes the j-th component of the Lévy random vector.

After the basic update, each obtains a basic candidate position $B_i^{(t)}$. The method then enters the multi-strategy search. Exploration starts from $B_i^{(t)}$, while exploitation generates a candidate directly from the current position $X_i^{(t)}$.

3) Multi-strategy search: After the basic MO update, Multi-Strategy MO performs the multi-strategy search. Search progress and population diversity are used to adjust the exploration perturbation. The method then selects between exploration and exploitation with a fixed probability. Exploration includes differential, Gaussian, and Lévy strategies. Exploitation uses historical-best guidance, elite guidance, and Gaussian perturbation to generate a candidate position. The overall search process is shown in Fig. 2.

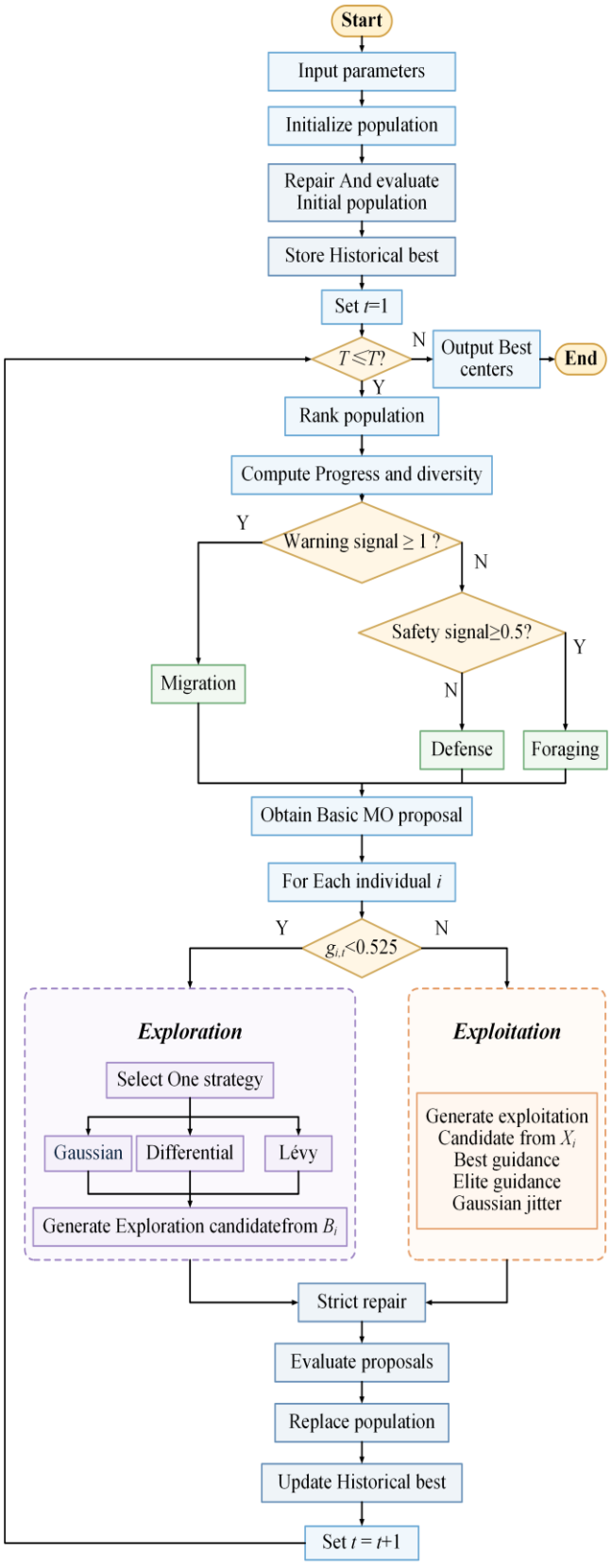

Fig. 2. Flowchart of the multi-strategy MO search process.

The search progress is first defined as follows:

$$\tau_t = \frac{t-1}{T-1}, \qquad t = 1,\ldots,T \tag{18}$$

The population diversity is defined as follows:

$$Z_{i,j}^{(t)} = \frac{X_{i,j}^{(t)} - l_j}{u_j - l_j}$$
$$V_t = \frac{1}{D}\sum_{j=1}^{D}\sqrt{\frac{1}{P-1}\sum_{i=1}^{P}\left(Z_{i,j}^{(t)} - \bar{Z}_j^{(t)}\right)^2} \tag{19}$$

where, $X_{i,j}^{(t)}$ denotes the coordinate of the i-th individual in the j-th search dimension at the beginning of the t-th update. l_j and u_j denote the lower and upper bounds of the j-th dimension, respectively. $\bar{Z}_j^{(t)}$ denotes the population mean of the normalized coordinate in the j-th dimension. V_t denotes the population diversity at the t-th update.

To adjust the exploration perturbation when the population becomes concentrated, the low-diversity coefficient d_t is defined as follows:

$$d_t = \max\left(0, 1 - \frac{V_t}{0.10}\right) \tag{20}$$

When $V_t \geq 0.10$, $d_t = 0$. When $V_t < 0.10$, d_t increases as the population diversity decreases.

For the i-th individual, an independent random number $g_{i,t} \sim U(0,1)$ is generated. Exploration is selected when $g_{i,t} < 0.525$. Otherwise, exploitation is selected. The exploration probability is fixed at 0.525 throughout the search. When exploration is selected, one of the differential, Gaussian, and Lévy strategies is chosen with equal probability. Only one exploration strategy is used each time.

The perturbation amplitudes of the three exploration strategies depend on the search progress τ_t and the low-diversity coefficient d_t:

$$a_{D,t} = 0.15 + 0.35\left(1 - \tau_t\right) + 0.20 d_t$$
$$a_{G,t} = 0.015 + 0.06\left(1 - \tau_t\right) + 0.08 d_t$$
$$a_{L,t} = 0.01 + 0.04\left(1 - \tau_t\right) + 0.05 d_t \tag{21}$$

where, $a_{D,t}$, $a_{G,t}$, and $a_{L,t}$ denote the perturbation coefficients for differential, Gaussian, and Lévy exploration, respectively. $\mathbf{Y}_i^{(t)}$ denotes the candidate position generated in the multi-strategy search.

In differential exploration, the basic candidate position is perturbed using the difference between two population positions:

$$\mathbf{Y}_i^{(t)} = \mathbf{B}_i^{(t)} + a_{D,t}\left(\mathbf{X}_{r_1}^{(t)} - \mathbf{X}_{r_2}^{(t)}\right) \tag{22}$$

where, r_1 and r_2 are two different population indices sampled without replacement. This update starts from the basic candidate position $\mathbf{B}_i^{(t)}$. The difference between the two population positions provides the search direction.

In Gaussian exploration, a Gaussian perturbation scaled by the search range is added to the basic candidate position:

$$\mathbf{Y}_i^{(t)} = \mathbf{B}_i^{(t)} + a_{G,t}\mathbf{R} \odot \mathbf{z}_i^{(t)}$$
$$\mathbf{z}_i^{(t)} \sim \mathrm{N}\left(\mathbf{0}, \mathbf{I}_D\right) \tag{23}$$

where, $\mathbf{R} = \mathbf{u} - \mathbf{l}$ is the vector of search ranges. $\mathbf{z}_i^{(t)}$ is a D-dimensional standard Gaussian random vector, and $\mathbf{I}_D$ is the D-dimensional identity matrix.

In Lévy exploration, a newly generated Lévy random vector is applied to the basic candidate position:

$$\mathbf{Y}_i^{(t)} = \mathbf{B}_i^{(t)} + a_{L,t}\mathbf{R} \odot \mathbf{L}_{i,\exp}^{(t)} \tag{24}$$

where, $\mathbf{L}_{i,\exp}^{(t)}$ denotes a Lévy random vector generated using the method described above.

Unlike exploration, exploitation starts directly from the position $\mathbf{X}_i^{(t)}$ at the beginning of the current update. The historical best position and the current elite individuals are then used to generate the candidate position.

First, the top individuals are selected according to the fitness ranking of the current population. The number of elite individuals and their mean position are defined as follows:

$$P_e = \max\left(1, \lceil 0.20P \rceil\right)$$
$$\mathbf{E}^{(t)} = \frac{1}{P_e}\sum_{j=1}^{P_e}\mathbf{X}_{\rho_t(j)}^{(t)} \tag{25}$$

where, P_e denotes the number of elite individuals. $\rho_t(j)$ denotes the index of the individual ranked j-th by the current fitness values. $\boldsymbol{E}^{(t)}$ denotes the mean position of the elite individuals.

The weights for historical-best guidance and elite guidance are defined as follows:

$$b_t = 0.20 + 0.55\tau_t$$
$$e_t = 0.10 + 0.25\tau_t \tag{26}$$

where, b_t and e_t denote the weights for historical-best guidance and elite guidance, respectively.

The exploitation candidate position is then defined as follows:

$$\begin{aligned}\mathbf{Y}_i^{(t)} = \mathbf{X}_i^{(t)} &+ b_t q_{b,i,t}\left(\mathbf{G}^{(t)} - \mathbf{X}_i^{(t)}\right) \\ &+ e_t q_{e,i,t}\left(\mathbf{E}^{(t)} - \mathbf{X}_i^{(t)}\right) \\ &+ 0.01(1-\tau_t)\mathbf{R}\square\ \boldsymbol{\varepsilon}_i^{(t)}\end{aligned} \tag{27}$$

where, $q_{b,i,t}$ and $q_{e,i,t}$ are independent random numbers drawn from $U(0,1)$. $\varepsilon_i^{(t)}$ is a D-dimensional standard Gaussian random vector. The three terms represent historical-best guidance, elite guidance, and Gaussian perturbation that decreases with the search progress, respectively.

After $\mathbf{Y}_i^{(t)}$ is generated, non-finite coordinates are replaced by the midpoint of the search bounds, and out-of-bounds coordinates are clipped to [0, 255]. Duplicate cluster centers are identified using a Euclidean distance threshold of $\tau = 10^{-10}\max\left(1, \|(255,255,255)\|_2\right)$ and processed sequentially. Each duplicate center is replaced by the sample with the largest computed minimum squared distance to the previously accepted centers. The samples are obtained from the SLIC superpixel mean RGB values expanded in pixel order, with ties resolved by selecting the first sample. If fewer than K distinct clustering samples are available, or no replacement distance exceeds the threshold, the repair process terminates.

The Multi-Strategy MO search procedure is summarized in Algorithm 1. In each iteration, the basic MO rules in Eq. (11)–(15) first generate $\mathbf{B}_i^{(t)}$. Exploration then generates $\mathbf{Y}_i^{(t)}$ from $\mathbf{B}_i^{(t)}$ using Eq. (22)–(24), whereas exploitation generates $\mathbf{Y}_i^{(t)}$ directly from $\mathbf{X}_i^{(t)}$ using Eq. (25)–(27). Invalid RGB coordinates and duplicate cluster centers in each $\mathbf{Y}_i^{(t)}$ are repaired before fitness evaluation using (8). The evaluated $\mathbf{Y}_i^{(t)}$ candidates then replace the current population. The historical-best position is updated only if the new population achieves a lower objective value than the stored best.

Algorithm 1: Pseudocode of the Multi-Strategy MO Cluster-Center Search

Input: RGB image; SLIC RGB means, pixel counts, and label map; K, P, T; random seed.
Output: Historical-best RGB cluster centers.

Set the twister seed; initialize P positions $\mathbf{X}_i^{(1)}$ with K distinct observed RGB colors using Eq. (9).
Reset the same seed. Repair the initial positions; recover their center sets using Eq. (9) and evaluate using Eq. (8). Store the historical best.
For $t = 1,...,T$ do
  Use generation-start states.
  Compute V_t using Eq. (19); rank population indices without reordering positions.
  Obtain $\mathbf{S}_2^{(t)}$, $\mathbf{W}^{(t)}$; compute $\mathbf{E}^{(t)}$ using Eq. (25) and τ_t using Eq. (18).
  Draw shared δ_t and η_t using Eq. (10).
  if $|\delta_t| \geq 1$ then
    Generate all $\mathbf{B}_i^{(t)}$ using Eq. (11).
  else if $\eta_t \geq 0.5$ then
    Generate all $\mathbf{B}_i^{(t)}$ using Eq. (12).
  else
    Generate all $\mathbf{B}_i^{(t)}$ using Eq. (13)–(15).
  end if
  Compute d_t from V_t using Eq. (20).
  for $i = 1,...,P$ do
    Draw $g_{i,t} \sim U(0,1)$.
    if $g_{i,t} < 0.525$ then
      Select one of Eq. (22)–(24) with equal probability; generate $\mathbf{Y}_i^{(t)}$ from $\mathbf{B}_i^{(t)}$ using Eq. (21).
    else
      Draw the Gaussian vector, then the historical-best and elite random factors; generate $\mathbf{Y}_i^{(t)}$ from $\mathbf{X}_i^{(t)}$, $\mathbf{G}^{(t)}$, and $\mathbf{E}^{(t)}$ using Eq. (26)–(27).
    end if
  end for
  for $i = 1,...,P$ do
    Repair $Y_i^{(t)}$; recover its centers using Eq. (9) and evaluate using Eq. (8).
  end for
  Replace the entire population with the repaired candidates and their objective values, without parent-candidate comparison. Update the historical best only if the new minimum objective is strictly lower; otherwise retain it.
end for
Recover the historical-best RGB centers using the inverse of Eq. (9). Re-evaluate outside SearchFE; return the centers if the stored objective agrees within $10^{-8}\max(1, \text{stored objective})$, otherwise report an error.

C. Weighted K-Means

After the Multi-Strategy MO search, the resulting RGB cluster centers are used to initialize weighted K-means. Weighted K-means uses the superpixel mean RGB vectors and pixel counts defined in Section III-A. It alternates between label assignment and cluster-center update to refine the searched cluster centers.

First, the cluster label of the s-th superpixel is determined from the squared Euclidean distance between its mean RGB vector and the current cluster centers:

$$q_s = \arg\min_{1\leq k\leq K} \|\boldsymbol{\mu}_s - \mathbf{c}_k\|_2^2 \tag{28}$$

After label assignment, the cluster centers are updated using the pixel counts of the superpixels. For the k-th nonempty cluster, the cluster center is calculated as follows:

$$\mathbf{c}_k = \frac{\sum_{s:q_s=k} n_s \boldsymbol{\mu}_s}{\sum_{s:q_s=k} n_s} \tag{29}$$

where, n_s denotes the number of pixels in the s-th superpixel. Thus, the contribution of each superpixel to the center update depends on its pixel count. This center update is consistent with the weighted clustering objective defined in Section III-A.

Label assignment and cluster-center update are repeated until the maximum cluster-center displacement or the relative change in weighted SSE is no greater than 10^{-8}, or until 100 iterations are reached. If an empty cluster occurs, its center is reinitialized using the sample with the largest current squared assignment error. Multiple empty clusters are processed sequentially in cluster-index order, followed by label reassignment.

After weighted K-means is completed, the final label of each superpixel is determined using the final cluster centers and the corresponding label-assignment equation. The label of each superpixel is then assigned to all pixels within that superpixel. For any $n \in \Omega_s$,

$$q_n = q_s, \qquad n \in \Omega_s \tag{30}$$

where, q_n denotes the cluster label of the n-th pixel. Therefore, all pixels in the same superpixel have the same cluster label. The final segmentation result is obtained from these pixel labels.

D. Multi-Strategy MO-K-Means Segmentation Procedure

Based on the method described above, the color image segmentation procedure is summarized as follows.

1) The RGB image is first input, and the number of clusters and other parameters are set. The image is then divided into superpixels using SLIC. For each superpixel, the mean RGB vector and pixel count are calculated to construct the weighted clustering data.

2) A population of RGB cluster centers is initialized, and its fitness is evaluated using the weighted clustering objective. The Multi-Strategy MO search shown in Fig. 2 is then performed to obtain the searched RGB cluster centers.

3) The searched cluster centers are used to initialize weighted K-means. Label assignment and cluster-center update are performed alternately until the final cluster centers are obtained.

4) The cluster label of each superpixel is determined using the final cluster centers. The label assigned to each superpixel is then applied to all pixels within that superpixel to obtain the final segmentation result.

These steps form the proposed Multi-Strategy MO-K-means procedure for color image segmentation.

IV. Experimental Results and Analysis

In this section, the proposed method is evaluated in terms of cluster-center search and final segmentation performance.

A. Experimental Setup and Evaluation

Experiments were conducted on the 200 color images in the official test set of the Berkeley Segmentation Dataset and Benchmark 500 (BSDS500), using MATLAB R2024b on a Windows 11 computer equipped with an Intel Core i7-14650HX CPU and 32 GiB RAM. All images were processed in RGB space. The number of clusters was set to $K = 5$, and each image was independently processed 10 times. For SLIC, the target superpixel area was set to 32 pixels, and the compactness parameter was set to 10. The main parameters of Multi-Strategy MO are listed in Table I.

TABLE I. Main Parameters of Multi-Strategy MO

Parameter	K	Population size	Iterations	Function evaluations	P_{exp}
Value	5	12	15	192	0.525

Two levels of evaluation were used. At the search level, SearchSSE denotes the weighted clustering objective obtained at the end of the optimizer search and was used to evaluate cluster-center search performance. Multi-Strategy MO was compared with Original MO, Particle Swarm Optimization (PSO), Differential Evolution (DE), and Partial Reinforcement Optimizer (PRO). The five optimizers used the same initial populations, clustering objective, and number of function evaluations.

At the segmentation level, the final pixel labels were evaluated using Boundary-F, Probabilistic Rand Index (PRI), Variation of Information (VI), Global Consistency Error (GCE), and Boundary Displacement Error (BDE). Higher values indicate better performance for Boundary-F and PRI, whereas lower values are preferred for VI, GCE, and BDE. A pixel-level RGB K-means++ method was used as the complete segmentation baseline. It applied K-means++ initialization and standard unweighted K-means directly to RGB pixels, without SLIC superpixels, MO search, pixel-count weighting, or weighted K-means refinement. The SLIC + weighted K-means++ baseline used superpixel mean RGB vectors and pixel-count weights for K-means++ initialization and subsequent weighted K-means refinement. The SLIC superpixels were also evaluated using Boundary Recall (BR), Undersegmentation Error (UE), and Achievable Segmentation Accuracy (ASA) to assess boundary preservation, undersegmentation, and the maximum achievable agreement with ground-truth regions, respectively.

For each method, the results were averaged over ten runs for each image, and the mean and SD were then calculated across all 200 images. SearchSSE was compared using two-sided Wilcoxon signed-rank tests with Holm correction. Paired effect sizes and 95% confidence intervals were also calculated, with the intervals estimated using paired bootstrap resampling.

B. Cluster-Center Search Performance

SearchSSE was used to compare the cluster-center search performance of Multi-Strategy MO, Original MO, PSO, DE, and PRO. The SearchSSE distributions and convergence curves of the five optimizers are shown in Fig. 3 and Fig. 4, respectively, and the mean results are listed in Table II. Multi-Strategy MO obtained the lowest SearchSSE. Compared with Original MO, PSO, DE, and PRO, SearchSSE was reduced by 20.11%, 16.29%, 16.48%, and 18.85%, respectively. The convergence curves show the same trend.

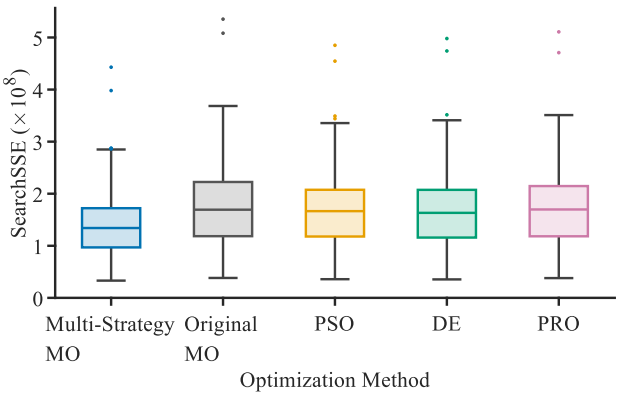

Fig. 3. SearchSSE distributions of the compared optimizers.

TABLE II. CLUSTER-CENTER SEARCH PERFORMANCE UNDER THE UNIFIED SETTING

Method	SearchSSE (Mean ± SD, ×10⁸)	Reduction (%) [95% CI]
Multi-Strategy MO	1.431 ± 0.632	—
Original MO	1.791 ± 0.813	20.11[19.32, 20.92]
PSO	1.709 ± 0.724	16.29[15.53, 17.07]
DE	1.713 ± 0.751	16.48[15.87, 17.11]
PRO	1.763 ± 0.759	18.85[17.97, 19.81]

Multi-Strategy MO obtained a lower mean SearchSSE than the four comparison optimizers across the 200 test images. The differences remained significant after Holm correction ($p < 10^{-33}$), with paired effect sizes ranging from 1.68 to 2.05. The 95% confidence intervals for the relative reductions were also above zero.

To examine the contributions of the main search strategies, differential exploration, historical-best guidance, and elite guidance were removed individually and in combination. The resulting configurations were compared in terms of SearchSSE, convergence behavior, and relative optimization time.

As shown in Table III, removing differential exploration, historical-best guidance, and elite guidance separately increased SearchSSE by 3.84%, 7.92%, and 3.28%, respectively. The largest increase of 12.77% was observed when all three strategies were removed. The proposed method also achieved the lowest normalized convergence AUC of 0.8516.

TABLE III. ABLATION OF THE MAIN SEARCH MECHANISMS

Removed component	SearchSSE increase (%)	Normalized AUC	Relative optimization time (%)
Differential exploration	3.84	0.8633	99.69
Historical-best guidance	7.92	0.8903	99.56
Elite guidance	3.28	0.8698	99.74
Differential exploration and historical-best guidance	10.41	0.8956	100.00
Differential exploration and elite guidance	6.87	0.8785	99.26
Historical-best guidance and elite guidance	10.67	0.9052	99.61
All three strategies	12.77	0.9099	99.13
Proposed method	0.00	0.8516	100.00

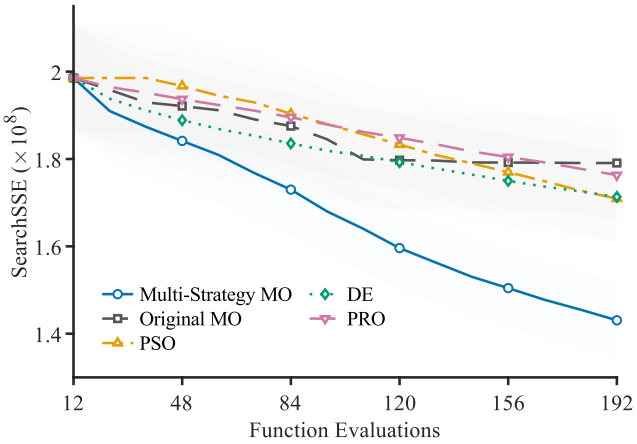

Fig. 4. Convergence curves of the compared optimizers.

Across all BSDS500 test images, 25.81% of repair calls modified candidate positions, and 2.34% involved duplicate-center replacement. SSE decreased after repair for 94.6% of the modified candidates, with a median relative reduction of 5.07%.

C. Effects of Segmentation Components

The effects of SLIC-based superpixel representation and weighted K-means refinement on the segmentation process were further examined. For SLIC, Boundary-F was compared with and without superpixel-based spatial representation at three image scales. As shown in Table IV, Boundary-F was increased by 0.1140, 0.0912, and 0.0765 at R256, R384, and R512-cap, respectively, with an average gain of 0.0939. Positive improvements were obtained at all three image scales, indicating that the superpixel representation helps preserve spatial structure and region boundaries.

TABLE IV. EFFECT OF SLIC REPRESENTATION ON BOUNDARY QUALITY

Resolution	Mean ΔBoundary-F
R256	0.1140
R384	0.0912
R512-cap	0.0765
Average	0.0939

After the cluster-center search by Multi-Strategy MO, weighted K-means was used to further refine the searched cluster centers. After refinement, the mean weighted SSE was reduced from approximately 1.4307×10^8 to 1.0200×10^8, while the mean PRI increased from 0.6985 to 0.7027. These results show that weighted K-means further refines the clustering result before the final pixel labels are generated.

D. Segmentation Results and Analysis

First, the final segmentation performance of Multi-Strategy MO-K-means was compared with K-means++ and SLIC + weighted K-means++ under the same BSDS500 test setting. The quantitative results are shown in Table V.

TABLE V. SEGMENTATION PERFORMANCE COMPARISON

Metric	K-means++	SLIC + Weighted K-means++	Multi-Strategy MO-K-means	Improvement (%)
Boundary-F ↑	0.404	0.474	0.487	20.41
PRI ↑	0.699	0.699	0.703	0.58
VI ↓	3.245	3.156	3.177	2.09
GCE ↓	0.461	0.452	0.450	2.33
BDE ↓	14.521	14.260	13.880	4.41

Multi-Strategy MO-K-means achieved a Boundary-F of 0.486920, representing a relative improvement of 20.41% over K-means++. Improvements were also observed in PRI, VI, GCE, and BDE, demonstrating better overall segmentation performance. SLIC + weighted K-means++ also obtained favorable segmentation results, with a Boundary-F of 0.474 and a BDE of 14.260, reflecting the contribution of superpixel representation to boundary preservation.

Fig. 5 shows the segmentation results for three test images. Compared with K-means++, Multi-Strategy MO-K-means produced more coherent regions and reduced fine-scale fragmentation. These differences can be clearly observed around object boundaries and in large homogeneous regions.

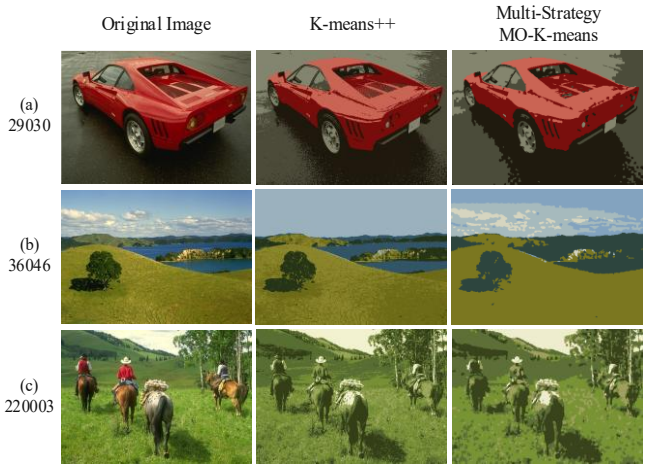

Fig. 5. Qualitative comparison between K-means++ and the multi-strategy MO-K-means.

Table VI lists the PRI and BDE results reported by Xu et al. [21] alongside those obtained in this work. The published results are included for reference under their original evaluation settings.

TABLE VI. COMPARISON WITH PUBLISHED RESULTS

Method	PRI	BDE
FCM-SLICM	0.605	25.351
FRFCM	0.680	15.318
KWFLICM	0.633	19.905
RSSFCA	0.690	16.219
SFFCM	0.632	20.964
SLIC-AFCF	0.714	15.440
GMM-AFCF	0.707	15.567
MMGR-AFCF	0.718	16.367
DSP-FCM$_N$	0.720	15.270
DSP-FCM	0.727	14.217
Multi-Strategy MO-K-means	0.703	13.880

In addition, MetaImClust reported PRI values of 0.6800, 0.6798, and 0.6796 for PO, AHA, and PRO, respectively [22], while Multi-Strategy MO-K-means achieved a PRI of 0.7027. These results further show that the proposed method achieves competitive performance in terms of PRI.

TABLE VII. COMPUTATIONAL COST ANALYSIS OF THE PROPOSED METHOD

Computational stage	Time (s/image)	Proportion (%)
SLIC preprocessing	0.086 ± 0.009	0.94
MO optimization (excluding repair)	3.420 ± 0.335	37.26
Candidate repair	4.505 ± 0.453	49.09
Weighted K-means refinement	0.370 ± 0.156	4.03
Final label assignment	0.796 ± 0.101	8.67
Total	9.177 ± 0.930	100.00

As shown in Table VII, the proposed method required 9.177 ± 0.930 s per image on average across the BSDS500 test set. MO search and candidate repair accounted for most of the execution time. Weighted K-means refinement required 27.23 ± 11.19 iterations on average. The sampled peak process working set was 1142.21 ± 322.00 MB, with an average increase of 29.18 MB over the pre-run baseline.

TABLE VIII. BOUNDARY QUALITY OF SLIC SUPERPIXELS

Resolution	BR ↑	UE ↓	ASA ↑
R256	0.8845	0.0607	0.9695
R384	0.9542	0.0454	0.9773
R512-cap	0.9805	0.0379	0.9810

Table VIII presents the boundary quality of SLIC superpixels at three image scales. BR and ASA increased with image resolution, while UE decreased. At R512-cap, BR and ASA reached 0.9805 and 0.9810, respectively, with a UE of 0.0379.

Table IX presents the segmentation results for K = 3, 5, 7, and 9. SLIC + Weighted K-means++ achieved higher Boundary-F and lower VI than conventional K-means++ at all four K values. Multi-Strategy MO-K-means obtained the highest Boundary-F and PRI among the three methods, with Boundary-F improvements of 14.6%–22.0% over K-means++. As K increased, Boundary-F decreased, whereas PRI and VI increased for all three methods.

TABLE IX. SENSITIVITY OF SEGMENTATION PERFORMANCE TO THE NUMBER OF CLUSTERS

K	Method	Boundary-F ↑	PRI ↑	VI ↓
3	K-means++	0.4573	0.6511	2.8773
	SLIC + Weighted K-means++	0.5123	0.6527	2.8278
	Multi-Strategy MO-K-means	0.5243	0.6561	2.8272
5	K-means++	0.4044	0.6987	3.2453
	SLIC + Weighted K-means++	0.4744	0.6986	3.1558
	Multi-Strategy MO-K-means	0.4869	0.7027	3.1774
7	K-means++	0.3763	0.7113	3.5172
	SLIC + Weighted K-means++	0.4447	0.7089	3.4295
	Multi-Strategy MO-K-means	0.4573	0.7152	3.4288
9	K-means++	0.3582	0.7169	3.7354
	SLIC + Weighted K-means++	0.4247	0.7140	3.6220
	Multi-Strategy MO-K-means	0.4369	0.7214	3.6246

V. CONCLUSIONS

This study proposes a Multi-Strategy MO-K-means algorithm for unsupervised color image segmentation to address the sensitivity of traditional K-means to initial cluster centers and its limited use of spatial information. The proposed algorithm integrates SLIC superpixel preprocessing, an improved Musk Ox Optimizer, and weighted K-means. SLIC superpixels incorporate spatial information, while Multi-Strategy MO balances global exploration and local exploitation to obtain improved initial cluster centers for subsequent refinement by weighted K-means. Experimental results show that Multi-Strategy MO reduces SearchSSE by 16.29%–20.11% compared with four other optimizers. The proposed method also improves all five segmentation metrics over K-means++, with a 20.41% relative improvement in Boundary-F. These results demonstrate its ability to produce more coherent segmentation regions and preserve image boundaries.

The number of clusters is currently predefined, and the segmentation results depend on the SLIC superpixel partition. Future work will focus on automatic selection of the number of clusters and boundary-aware superpixel refinement to improve the adaptability of the proposed method to different image structures.

ACKNOWLEDGMENT

This work was supported by the National Key Research and Development Program of China under Grant 2024YFB2908601.

REFERENCES

[1] K. K. Brar et al., 'Image segmentation review: Theoretical background and recent advances', Information Fusion, vol. 114, p. 102608, Feb. 2025.

[2] A. M. Ikotun, A. E. Ezugwu, L. Abualigah, B. Abuhaija, and J. Heming, 'K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data', Information Sciences, vol. 622, pp. 178–210, Apr. 2023.

[3] N. Bajpai, J. H. Paik, and S. Sarkar, 'Balanced seed selection for K-means clustering with determinantal point process', Pattern Recognition, vol. 164, p. 111548, Aug. 2025.

[4] A. Kumar, A. Kumar, R. Mallipeddi, and D.-G. Lee, 'High-density cluster core-based k -means clustering with an unknown number of clusters', Applied Soft Computing, vol. 155, p. 111419, Apr. 2024.

[5] I. B. Barcelos, F. D. C. Belém, L. D. M. João, Z. K. G. D. Patrocínio, A. X. Falcão, and S. J. F. Guimarães, 'A Comprehensive Review and New Taxonomy on Superpixel Segmentation', ACM Comput. Surv., vol. 56, no. 8, pp. 1–39, Aug. 2024.

[6] B. Sasmal and K. G. Dhal, 'A survey on the utilization of Superpixel image for clustering-based image segmentation', Multimed Tools Appl, vol. 82, no. 23, pp. 35493–35555, Sep. 2023.

[7] Zeyu Xie, Xiao Luo, Defang Zhao, and Xinyu Chen, 'Fuzzy C‐means clustering algorithm based on superpixel merging and multi‐feature adaptive fusion measurement', IET Image Processing, vol. 18, no. 1, pp. 140‐153, Jan. 2024.

[8] B. Ji, X. Hu, F. Ding, Y. Ji, and H. Gao, 'An effective color image segmentation approach using superpixel-neutrosophic C-means clustering and gradient-structural similarity', Optik, vol. 260, p. 169039, Jun. 2022.

[9] L. Chen, Y.-P. Zhao, and C. Zhang, 'Efficient kernel fuzzy clustering via random Fourier superpixel and graph prior for color image segmentation', Engineering Applications of Artificial Intelligence, vol. 116, p. 105335, Nov. 2022.

[10] C. Wu and J. Zhao, 'Joint learning framework of superpixel generation and fuzzy sparse subspace clustering for color image segmentation', Signal Processing, vol. 222, p. 109515, Sep. 2024.

[11] S. Zhu, Z. Liu, S. Letchmunan, and H. Qiu, 'Auto feature weighted c -means type clustering methods for color image segmentation', Engineering Applications of Artificial Intelligence, vol. 153, p. 110768, Aug. 2025.

[12] J. Wu, X. Wang, Y. Liu, and C. Fang, 'Adaptive sparse regularized fuzzy clustering noise image segmentation algorithm based on complementary spatial information', Expert Systems with Applications, vol. 256, p. 124943, Dec. 2024.

[13] S. J. Mousavirad, H. Ebrahimpour-Komleh, and G. Schaefer, 'Automatic clustering using a local search-based human mental search algorithm for image segmentation', Applied Soft Computing, vol. 96, p. 106604, Nov. 2020.

[14] G. Hu, Y. Zheng, E. H. Houssein, and G. Wei, 'DRPSO: A multi-strategy fusion particle swarm optimization algorithm with a replacement mechanism for colon cancer pathology image segmentation', Computers in Biology and Medicine, vol. 178, p. 108780, Aug. 2024.

[15] A. Dey, S. Dey, S. Bhattacharyya, J. Platos, and V. Snasel, 'Quantum-inspired meta‐heuristic approaches for automatic clustering of colour images', Int J of Intelligent Sys, vol. 36, no. 9, pp. 4852‐4901, Sep. 2021.

[16] X. Zhang, S. Wen, and D. Wang, 'Multi-population biogeography-based optimization algorithm and its application to image segmentation', Applied Soft Computing, vol. 124, p. 109005, Jul. 2022.

[17] B. Ma, J. L. Pereira, D. Oliva, S. Liu, and Y.-H. Kuo, 'Manta ray foraging optimizer-based image segmentation with a two-strategy enhancement', Knowledge-Based Systems, vol. 262, p. 110247, Feb. 2023.

[18] S. J. Mousavirad, G. Schaefer, H. Zhou, and M. H. Moghadam, 'How effective are current population-based metaheuristic algorithms for variance-based multi-level image thresholding?', Knowledge-Based Systems, vol. 272, p. 110587, Jul. 2023.

[19] A. Seyyedabbasi, 'A Hybrid Multi-Strategy Optimization Metaheuristic Algorithm for Multi-Level Thresholding Color Image Segmentation', Applied Sciences, vol. 15, no. 13, p. 7255, Jun. 2025.

[20] Y. Yuan et al., 'Musk ox optimizer (MO): a novel optimization algorithm and its application', Cluster Comput, vol. 28, no. 16, p. 1041, Dec. 2025.

[21] G. Xu, T. Suo, Z. Kong, L. Deng, and H. Zhu, 'Deep Superpixel Prior Fuzzy C‐Means Clustering With Weighted Local Entropy for Image Segmentation', Electronics Letters, vol. 61, no. 1, p. e70503, 2025.

[22] T. Bharasa, A. Das, K. G. Dhal, K. M. Das, and E. H. Houssein, 'MetaImClust: Automatic Crisp Clustering using Eminent Metaheuristics for Image Segmentation', Knowledge-Based Systems, vol. 346, p. 116129, Jul. 2026.