2023 5th International Conference on Data-driven Optimization of Complex Systems (DOCS) | 979-8-3503-9352-1/23/$31.00 ©2023 IEEE | DOI: 10.1109/DOCS60977.2023.10294635

Surrogate-Assisted Hybrid Metaheuristic for Mixed-Variable 3-D Deployment Optimization of Directional Sensor Networks

Yuntian Zhang[1, 2], Chen Chen*[1, 2], Tongyu Wu[1, 2], Changhao Miao[1, 2], and Shuxin Ding[3, 4]
[1]School of Automation, Beijing Institute of Technology, Beijing 100081, China
[2]National Key Lab of Autonomous Intelligent Unmanned Systems, Beijing 100081, China
[3]Signal and Communication Research Institute, China Academy of Railway Sciences Corporation Limited, Beijing 100081, China
[4]Traffic Management Laboratory for High-Speed Railway, National Engineering Research Center of System Technology for High-Speed Railway and Urban Rail Transit, Beijing 100081, China

***Abstract*—A major concern in designing sensor networks is the deployment problem. However, establishing an efficient algorithm for the real-world deployment problem is challenging due to three issues, which are 1) the realistic mixed-integer nonlinear programming problem (MINLP) with mixed-variable; 2) the combinatorial subset selection problem; and 3) the expensive computational cost for fitness evaluation in the 3-D coverage problem. Therefore, this paper addresses these challenges and proposes a surrogate-assisted hybrid metaheuristic for mixed-variable 3-D deployment optimization of directional sensor networks (DSNs). First, an MINLP with flexible coordinate transformation technique and an efficient mixed-variable encoding scheme are introduced to model and represent the problem. We propose hybrid metaheuristic which applies two reproduction methods respectively for discrete and continuous variables. Second, we design sparse population-based incremental learning (s-PBIL) to handle inherent subset selection problem. s-PBIL could accurately learn the required information, and automatically learn a sparse distribution. Third, a mixed-variable surrogate with unifying space under Bayesian model management is incorporated to reduce the expensive computational cost. Experiment results on real-world deployment scenarios scaling from small-size to large-size show the effectiveness of the proposed algorithm.**

***Index Terms*—Hybrid metaheuristic, mixed-variable, 3-D deployment, directional sensor networks (DSNs), sparse population-based incremental learning (s-PBIL), surrogate.**

I. Introduction

Sensor networks consist of many interconnected sensor nodes that can sense, measure, and gather information about their surrounding environment [1], [2]. A major concern in designing sensor networks is the deployment problem. Frequently, masses of real-life applications call for a practical issue: how to deploy a set of directional sensors in finite candidate positions under 3-D realistic scenarios, to optimize the performance indicator of interest? Take an example, in the defensive scenario, commanders tend to deploy radars in several candidate positions, to monitor the incoming targets [3]. Another example in civilian illustrates the requirement of determining optimal camera placement to achieve angular coverage continuously over a given region [4].

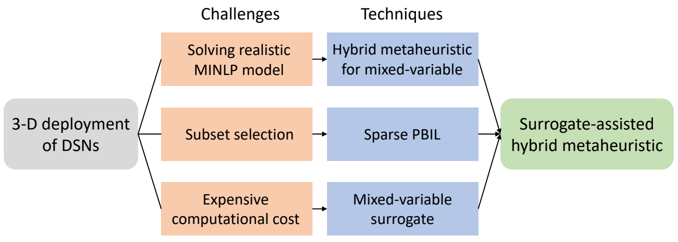

Challenges
Techniques
3-D deployment of DSNs
Solving realistic MINLP model
Subset selection
Expensive computational cost
Hybrid metaheuristic for mixed-variable
Sparse PBIL
Mixed-variable surrogate
Surrogate-assisted hybrid metaheuristic

Fig. 1. Motivations and contributions of this paper, which incorporates three techniques to address three challenges in 3-D deployment optimization of DSNs.

The above issue sheds light on the requirement of realistic mathematical models and efficient optimization approaches[1]. Table I summarizes the reviewed literature of high quality on the sensor networks deployment problem. We recap these previous works in Section II in detail. However, there are at least the following challenges in three aspects. First, 3-D deployment optimization of directional sensor networks (DSNs) consists of different variable types. Most of the literature considers the continuous deployment space, in which the sensors can conduct arbitrary movement. While in real-life, a set of sensors may select their positions in finite candidate positions [3], incurring relatively challenging discrete optimization. What is more, the discrete part is coupled with the optimization of continuous parameters (i.e., configurations of sensors), resulting in a hard optimization problem. This

This work was supported in part by the Outstanding Youth Science Foundation under Grant 62022015; in part by the National Natural Science Foundations of China (NSFC) under Grant 62273044, 62088101, and U20B2073; in part by the Shanghai Municipal Science and Technology Major Project under Grant 2021SHZDZX0100; in part by the Shanghai Municipal Commission of Science and Technology Project under Grant 19511132101; and in part by the BIT Research and Innovation Promoting Project under Grant 2022YCXY036. *(Corresponding author: Chen Chen.) (e-mail: xiaofan@bit.edu.cn)*

[1]In this paper, we use the approach and the algorithm interchangeably.

TABLE I
COMPARISON OF EXISTING SOLUTIONS TO THE SENSOR NETWORKS DEPLOYMENT

Reference	Dimension	Sensing Model	Deployment Space	Solution Method	MINLP Formulation?	Hybrid Algorithms?
[5]	2-D	Probabilistic, Omnidirectional	Continuous	Virtual Force Algorithm (VFA)	–	–
[3]	3-D	Probabilistic, Omnidirectional	Discrete	LINGO Solver	–	–
[6]	3-D	Binary, Omnidirectional	Continuous	Improved PSO	–	–
[7]	3-D	Probabilistic, Directional	Continuous	SA, L-BFGS, CMA-ES, etc.	–	–
[8]	2-D	Probabilistic, Omnidirectional	Continuous	d-PSO	–	–
[9]	2-D	Binary, Omnidirectional	Discrete	MOEA/D	✓	–
[10]	3-D	Binary, Directional	Continuous	Distributed Parallel MOEA	–	–
[11]	2-D	Binary, Omnidirectional	Continuous	Memetic Algorithm (MA)	–	–
[12]	3-D	Probabilistic, Directional	Discrete	Multi-objective GA	✓	–
[13]	2-D	Probabilistic, Omnidirectional	Continuous	Multitasking Co-evolutionary PSO	–	–
This paper	3-D	Probabilistic, Directional	Discrete	**Hybrid**: **s-PBIL** + SLPSO	✓	✓

SA: Simulated Annealing; **L-BFGS**: Limited-memory Broyden-Fletcher-Goldfarb-Shanno; **CMA-ES**: Covariance Matrix Adaptation Evolution Strategy; **MOEA/D**: Multi-objective Evolutionary Algorithm based on Decomposition; **SLPSO**: Social Learning Particle Swarm Optimizer.

mixed-variable characteristic calls for an efficient hybrid metaheuristic algorithm [14]. Second, the inherent subset selection problem (i.e., deploying a set of sensors in finite candidate positions) is hard for conventional evolutionary algorithms. Classical reproduction schemes, such as uniform crossover in genetic algorithm (GA) and velocity update scheme in particle swarm optimization (PSO) [15], can easily destroy the fixed-size and sparse subset pattern and are not desired. Third, the fitness evaluation for the coverage problem under the realistic sensing model and 3-D environment is computationally expensive (i.e., the calculation of line-of-sight in target coverage), making the optimization efficiency low.

This paper addresses these challenges with proposed techniques. The motivations and major contributions of this paper are summarized and justified as Fig. 1.

First, we formulate a mixed-integer nonlinear programming problem (MINLP) and introduce a flexible coordinate transformation technique to efficiently model realistic sensors. We then customize an efficient mixed-variable encoding scheme to represent the optimization problem. Binary coding and real number coding are employed for discrete part and continuous part, respectively. Furthermore, we propose hybrid metaheuristic which applies two reproduction methods respectively for discrete and continuous variables. Second, sparse population-based incremental learning (s-PBIL) is customized to handle subset selection problem. s-PBIL tries to sample while learning from good solutions and construct a probability model for discrete variables. Different from raw PBIL [16], s-PBIL could accurately learn the required information and automatically learn a sparse distribution. Third, a mixed-variable surrogate with unifying space under Bayesian model management is incorporated to reduce the expensive computational cost in optimizing 3-D deployment of DSNs.

With the above analyses and techniques, we obtain a surrogate-assisted hybrid metaheuristic for mixed-variable 3-D deployment optimization of DSNs. Experiment results on real-world deployment scenarios scaling from small-size to large-size show the effectiveness of the proposed algorithm.

The rest of this paper is organized as follows. The next Section summarizes related work. Section III and IV describe the mathematical modelling and optimization algorithms, respectively. Section V presents and analyses the experimental results. Section VI concludes this paper and discusses some open issues.

II. Related Work

Various optimization approaches are employed to tackle the deployment problem of sensor networks. Table I summarizes the reviewed literature of high quality on the sensor networks deployment problem.

The components of sensor networks deployment mainly include dimensions of deployment space, sensing model, continuity of deployment space, and solution methods, as shown in Table I. Different components result in various formulations, such as continuous optimization, combinatorial optimization, and MINLP in this paper. Various formulations call for suitable solution methods. Since the deployment problem in sensor networks is NP-complete [17], heuristic and evolutionary approaches are designed to crack this hard nut. Virtual Force Algorithm (VFA) and its variants are used as a sensor deployment strategy [5], [18]. Akbarzadeh *et al.* [7] first develop a probabilistic sensing model with line-of-sight based coverage to tackle the sensor placement problem. Several optimization schemes are introduced, including Simulated Annealing (SA), Limited-memory Broyden-Fletcher-Goldfarb-Shanno (L-BFGS), and Covariance Matrix Adaptation Evolution Strategy (CMA-ES). Acting as an efficient optimizer for continuous optimization problems, PSO [19] is widely used in the continuous deployment space. Ding *et al.* [8] propose a disturbance PSO (d-PSO) with a Gaussian perturbation to update the velocity, which shows fast convergence. Lian *et al.* [6] present a method for 3-D deployment optimization of sensor networks based on an improved PSO algorithm. Wang *et al.* [20] utilize improved PSO based on the resampling technique named resampled PSO (RPSO) to solve the coverage problem of sensor networks in the Internet of Things (IoT). Memetic Algorithm (MA) [11] is also applied to solve the deployment problem. A recent work [13] investigates an efficient co-evolutionary PSO with evolutionary multitasking (EMT) for stochastic area coverage of heterogeneous sensors.

We remark that Zhang *et al.* [9] formulate an MINLP, which considers limited-power sensors with adjustable ranges deployed along a linear domain to form a barrier to detect intruding incidents. MOEA/D framework as well as local search with problem-specific knowledge is employed to solve the problem. However, this paper oversimplifies the sensing model (i.e., binary and omnidirectional) and does not utilize the mixed-variable algorithm. Saad *et al.* [12] recently revisit the 3-D sensor networks deployment problem while considering realistic assumptions regarding the modelling of both sensors and the environment. A multi-objective genetic algorithm endowed with new adaptive and guided genetic operators is applied. However, this paper also differs from our work in two aspects. On the one hand, Saad *et al.* do not consider fixed-size subset selection problem, while considering the number of sensors deployed as an objective. It is friendly to GA operators. On the other hand, they do not introduce the mixed-variable scheme, while using binary coding for continuous variables. Due to the limitation of the length of the binary string, the performance of binary-code GA deteriorates when solving more precise problems [21].

III. Preliminaries and Problem Formulation

In this section, we introduce the deployment space and customized probabilistic directional sensing model. We also formulate the target coverage problem and make a preliminary analysis of it.

A. Deployment Space

This paper focuses on discrete deployment space. We first discretize the deployment space via grid points. Suppose we have 50km $\times$ 50km deployment space and discretize it with 1km resolution, there will be total of 2,500 grid points (x, y). Each grid point corresponds with a height value z. Thus, many (x, y, z) pairs formulate a 3-D matrix which is usually called the Digital Elevation Model (DEM).

However, in real-world scenarios, not all these grid points can be deployed with sensors. Lakes and steep hills are not friendly to the deployment issue. What is more, sometimes on the battlefield, sensors (i.e., radars) are deployed in several pre-constructed positions, which are carefully selected and could receive stable support [3]. Thus, the available deployment points are reduced to Z, the size of which is less than 2,500. If we possess $|S|$ sensors, the problem is converted to select $|S|$ points from Z for deployment. The dimension of Z is $|Z|$ and each dimension is a binary variable b_j. If the j-th point is selected to deploy a sensor, $b_j = 1$; otherwise, $b_j = 0$.

B. Sensing Model

To better describe real-world scenarios, probabilistic DSNs is introduced in recent years [7], [10], [12], [22]. Unlike the ideal binary and omnidirectional sensing model, the probabilistic directional sensing model can better depict various sensors and provide much flexibility. As shown below, the binary and omnidirectional sensing model is a special case of the probabilistic directional sensing model and could be included.

Suppose a sensor node s_i and target point q, the coverage probability $P(s_i, q)$ of probabilistic directional sensing model can be modelled as [12]

$$P(s_i, q) = \mu_d(\|s_i - q\|) \times \mu_p(\alpha_{p_{qi}}) \times \mu_t(\alpha_{t_{qi}}) \times v(s_i, q). \quad (1)$$

We further define each term in Eq. (1) in detail.

1) The distance member function: $\mu_d(\|s_i - q\|)$

$$\mu_d(\|s_i - q\|) = 1 - \frac{1}{1 + exp(-\beta_d(\|s_i - q\| - t_d))}, \quad (2)$$

where $\|s_i - q\| = \sqrt{\Delta x^2 + \Delta y^2 + \Delta z^2}$, $\Delta x = x_q - x_i$, $\Delta y = y_q - y_i$, $\Delta z = z_q - z_i$. The coordinate (x_i, y_i, z_i) of sensor s_i is determined by binary decision variable b_j (i.e., the j-th point) defined in Section III-A. The distance member function is configured through two parameters β_d and t_d.

2) The pan member function: $\mu_p(\alpha_{p_{qi}})$

$$\begin{aligned} \mu_p(\alpha_{p_{qi}}) = & \frac{1}{1 + exp(-\beta_p(\alpha_{p_{qi}} + t_p))} \\ & - \frac{1}{1 + exp(-\beta_p(\alpha_{p_{qi}} - t_p))}, \end{aligned} \quad (3)$$

where $\alpha_{p_{qi}} = arccos(\frac{y'_i}{\sqrt{(x'_i)^2 + (y'_i)^2}}) \in [0°, 180°]$. x'_i, y'_i are calculated based on coordinate transformation

$$\begin{bmatrix} x'_i \\ y'_i \end{bmatrix} = \begin{bmatrix} cos(\theta_i) & -sin(\theta_i) \\ sin(\theta_i) & cos(\theta_i) \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix},$$

where θ_i is a continuous decision variable with range of $[-180°, 180°]$. The pan member function is configured through two parameters β_p and t_p.

3) The tilt member function: $\mu_t(\alpha_{t_{qi}})$

$$\begin{aligned} \mu_t(\alpha_{p_{ti}}) = & \frac{1}{1 + exp(-\beta_t(\alpha_{t_{qi}} + t_t))} \\ & - \frac{1}{1 + exp(-\beta_t(\alpha_{t_{qi}} - t_t))}, \end{aligned} \quad (4)$$

where $\alpha_{t_{qi}} = arctan(\frac{z''_i}{\sqrt{(x''_i)^2 + (y''_i)^2 + (z''_i)^2}}) \in [-90°, 90°]$. x''_i, y''_i, z''_i are calculated based on coordinate transformation

$$\begin{bmatrix} x''_i \\ y''_i \\ z''_i \end{bmatrix} = \begin{bmatrix} cos(\varphi_i) & sin(\varphi_i) & 0 \\ -sin(\varphi_i) & cos(\varphi_i) & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \Delta x \\ \Delta y \\ \Delta z \end{bmatrix},$$

where φ_i is a continuous decision variable with range of $[-90°, 90°]$. The tilt member function is configured through two parameters β_t and t_t.

4) Line-of-sight (LOS): $v(s_i, q)$

Considering a realistic 3-D environment, some targets inside the sensor's range can not be detected because of the shelter from hills or buildings. Thus, we introduce a binary term $v(s_i, q) \in \{0, 1\}$ to represent the shelter. $v(s_i, q)$ is calculated via LOS algorithm similar in [7].

An example of a probabilistic directional sensing model is presented in Fig. 2. We mention that the binary and omnidirectional sensing model is a special case of the probabilistic directional sensing model and could be included. We can maximize β_d, β_p, and β_t for the binary behaviour of the sensor. We can set $t_p = 180$ and $t_d = 90$ to achieve an omnidirectional field of view.

Moreover, the coordinate transformation technique introduced is flexible enough to allow parallel computing. To be specific, for one sensor, all the target points can be processed and treated as a batch. Then, the batch is calculated via coordinate transformation in the matrix form. The matrix computation can benefit from the underlying acceleration scheme in simulation software (e.g., *Numpy* package in Python, *Parfor* in Matlab).

C. Problem Formulation

Suppose we possess a set of sensors S, and s_i is one of them. Therefore, each target point q is covered by multiple sensors collaboratively. We define the collaboration of multiple sensors as follows:

$$\begin{aligned} CP^S(p) &= P\left\{\bigcup_{i=1}^{|S|} P(s_i, q)\right\} \\ &= 1 - P\left\{\bigcap_{i=1}^{|S|} \overline{P(s_i, q)}\right\} \\ &= 1 - \prod_{i=1}^{|S|}\left[1 - P(s_i, q)\right]. \end{aligned} \tag{5}$$

For a target set Q to be covered, in our settings, each target q in Q is associated with a coordinate (x_q, y_q, z_q) and a threat value w_q. Our objective in the deployment problem is to enable sensors to better coverage target points, thus minimizing the threat. To summarize, we formulate an MINLP model:

$$\min_{b_j, \theta_i, \varphi_i} \quad \sum_{q=1}^{|Q|} w_q \left(\prod_{i=1}^{|S|}\left[1 - P(s_i, q)\right]\right) \tag{6}$$

$$s.t.$$

$$b_j \in \{0, 1\}, \;\; \theta_i \in [-180^\circ, 180^\circ], \;\; \varphi_i \in [-90^\circ, 90^\circ].$$

We can make a preliminary analysis that the decision variables include both binary and continuous ones. To be more specific, it is a subset selection problem coupled with continuous optimization. Moreover, the objective function in Eq. (6) is highly non-convex and nonlinear, which is computationally intractable for traditional solvers. To efficiently solve this MINLP, we present a novel approach in the following section.

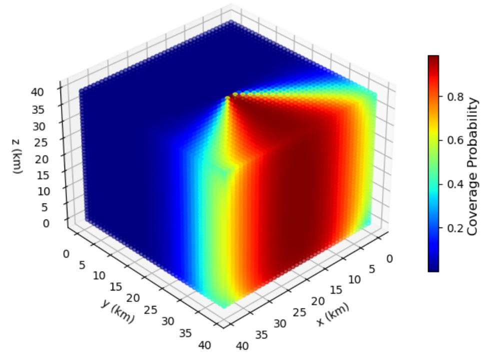


Fig. 2. An example of a probabilistic directional sensing model. We generate 50×50×50 target points in a cube and deploy a directional sensor in the coordinate $(25, 25, 25)$. Its parameters are $\beta_d = 0.5, \beta_p = 0.1, \beta_t = 0.15, t_d = 35, t_p = 50, t_t = 80$.

IV. Proposed Algorithms

A. Overview of Proposed Approach

We overview the proposed approach in Fig. 3. Major contributions are marked as **bold**. We detail three modules with different colors in the following paragraphs.

Blue Module describes the mixed-variable encoding scheme for MINLP and the flowchart of hybrid metaheuristic. An efficient encoding scheme is the basis of the metaheuristic. To handle MINLP, we apply binary coding for discrete part, and real number coding for continuous part. An example is given in Blue Module with 5 available candidate positions for deployment (i.e., blue dots) and 2 sensors to be deployed. The discrete variables encoding shows that the 2-nd and 4-th positions are selected for deployment (i.e., red dots). Since there are 2 sensors in this example, the dimensions of continuous variables are 2×2=4, where the first 2 dimensions represent pan angles and the last 2 dimensions represent tilt angles. Acting as an evolutionary approach, the proposed hybrid metaheuristic first initializes the population (i.e., solution pool), then iterates with reproduction, evaluation, and selection until the stopping criterion is met. Reproduction is conducted by s-PBIL and social learning particle swarm optimizer (SLPSO) in parallel and alignment. SLPSO is a widely used optimizer for continuous variables [23]. We select it for its success in various applications [24], [25] and its parameters efficiency.

Yellow Module specifies the design of the proposed s-PBIL. s-PBIL tries to sample while learning from good solutions and construct a probability model for discrete variables. s-PBIL could accurately learn the required information, and automatically learn a sparse distribution. We discuss s-PBIL in detail in Section IV-B.

Green Module introduces the mixed-variable surrogate with unifying space. The keys of surrogate design are the architecture of the surrogate model and the surrogate management. First, we apply Gaussian Process (GP) as the base surrogate

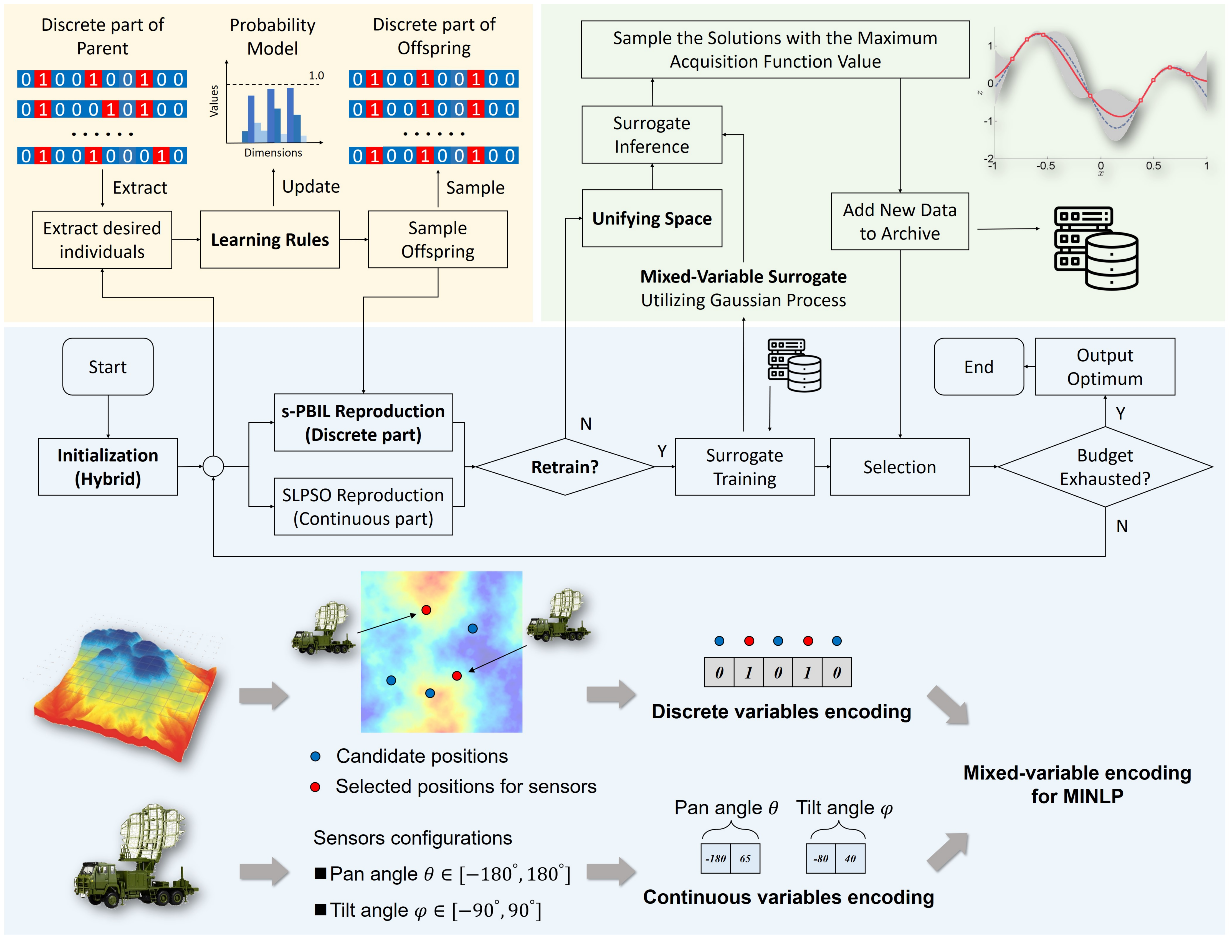

Discrete part of Parent
Probability Model
Discrete part of Offspring
Values
Dimensions
1.0
Extract
Update
Sample
Extract desired individuals
Learning Rules
Sample Offspring
Sample the Solutions with the Maximum Acquisition Function Value
Surrogate Inference
Unifying Space
Add New Data to Archive
Mixed-Variable Surrogate Utilizing Gaussian Process
Start
Initialization (Hybrid)
s-PBIL Reproduction (Discrete part)
SLPSO Reproduction (Continuous part)
Retrain?
N
Y
Surrogate Training
Selection
Budget Exhausted?
End
Output Optimum
Candidate positions
Selected positions for sensors
Discrete variables encoding
Mixed-variable encoding for MINLP
Sensors configurations
Pan angle θ ∈ [−180°, 180°]
Tilt angle φ ∈ [−90°, 90°]
Pan angle θ
Tilt angle φ
Continuous variables encoding

Fig. 3. Overview of the proposed approach. Blue Module describes the mixed-variable encoding scheme for MINLP and the flowchart of hybrid metaheuristic. Yellow Module specifies the design of the proposed s-PBIL. Green Module introduces the mixed-variable surrogate with unifying space. Major contributions are marked as **bold**.

model. However, GP always assumes that the input variables are real-valued [26] which is computationally intractable for mixed-variable. We utilize unifying space to handle this difficulty. Then, we apply Bayesian model management, which can also be treated as using an acquisition function for individual-based model management in evolutionary optimization [27]. We define the acquisition function via lower confidence bound (LCB). We discuss these issues in detail in Section IV-C.

B. *s-PBIL*

To handle subset selection problem (i.e., deploy a set of sensors in finite candidate positions), we design s-PBIL. s-PBIL tends to sample while learning from good solutions and construct a probability model for discrete part. From the perspective of learning a probability model, the key of our proposed s-PBIL is two-fold

- Accurately learn the required information
- Automatically learn a sparse distribution

as shown via heatmaps in Fig. 4. We visualize and compare the learned probability model by the proposed s-PBIL and PBIL. We can see that the probability model in Fig. 4(a) accurately learns the required information (i.e., 4 blue stripes) and automatically learns a sparse distribution. The probability model in Fig. 4(b) also reveals the 4 blue stripes, but its distribution is not sparse.

We present the learning rules consisting of rectification and mutation. Learning rules update the probability model iteratively. The rectification is as follows

$$P_i \leftarrow P_i \times (1.0 - LR) + p_i \times LR \tag{7}$$

where P_i denotes the value of i-th dimension of the probability model. We treat each dimension independently in s-PBIL. p_i indicates the value of i-th dimension of extracted individuals p in the parent. LR is the learning rate. Further, we generate a random float number between 0 and 1. If the number is less than the mutation probability MUT_{PROB}, the mutation is applied as

$$P_i \leftarrow P_i \times (1.0 - MUT) + random(0.0 \ or \ 1.0) \times (MUT) \tag{8}$$

where P_i denotes the value of i-th dimension of the probability model. $random(0.0 \ or \ 1.0)$ denotes a generated random number which is 0 or 1. MUT indicates the amount for mutation to affect the probability vector. Algorithm 1 summarizes the detailed procedure of the proposed s-PBIL. s-PBIL outputs

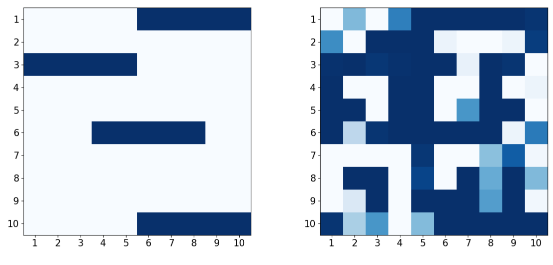


Fig. 4. Comparison between learned probability models. Both heatmaps represent 100-D probability models which are reshaped as 10×10. The darker the color, the larger the value. Take a customized toy problem as an example, $\max_{\boldsymbol{x}} f(\boldsymbol{x}) = \sum_{i=6}^{10} x_i + \sum_{i=20}^{24} x_i + \sum_{i=54}^{58} x_i + \sum_{i=96}^{100} x_i,\ x_i \in \{0,1\}$. It is not difficult to derive that when $f(x)$ reaches its optimum, the 6-D to 10-D, 20-D to 24-D, 54-D to 58-D, and 96-D to 100-D should be value 1. We can see that the probability model in (a) accurately learns the required information (i.e., 4 blue stripes) and automatically learns a sparse distribution. The probability model in (b) also reveals the 4 blue stripes, but its distribution is not sparse.

the updated probability model and the constructed discrete part of offspring. Since s-PBIL learns an accurate and sparse distribution, the decision maker could easily select a subset.

C. Mixed-variable Surrogate with Unifying Space

We recap that the keys of surrogate design are the architecture of the surrogate model and the surrogate management. We apply GP as the base surrogate model. However, GP always assumes that the input variables are real-valued which is computationally intractable for mixed-variable. To handle the mixed-variable issue with GP, we incorporate a transformation $T(\cdot)$ [26]. The input variables which correspond to a binary integer-valued variable are rounded to the closest integer value via $T(\cdot)$. The transformation is performed before computing the covariance function $\mathcal{K}(\cdot,\cdot)$, resulting in a new covariance function $\mathcal{K}'(\cdot,\cdot)$

$$\mathcal{K}'(\boldsymbol{x_i}, \boldsymbol{x_j}) = \mathcal{K}(T(\boldsymbol{x_i}), T(\boldsymbol{x_j})) \tag{9}$$

where $\boldsymbol{x_i}$ and $\boldsymbol{x_j}$ represent input data points. After establishing the mixed-variable surrogate utilizing GP with unifying space, we specify the LCB as an acquisition function

$$LCB(\boldsymbol{x}) = \mu(\boldsymbol{x}) - \beta\sigma(\boldsymbol{x}) \tag{10}$$

where $\boldsymbol{x}$ denotes an individual. $\mu(\boldsymbol{x})$ and $\sigma(\boldsymbol{x})$ are the mean and variance of the model at position $\boldsymbol{x}$. β is a parameter explicitly balancing the exploration and exploitation. When β is small, solutions that are expected to be high-performing are favored. On the contrary, when β is large, the exploration is conducted on currently uncharted areas in the search space.

Based on the acquisition function, Bayesian model management calculates its value for each individual in the population and chooses part of individuals with maximum values for sampling (i.e., fitness evaluation using the real objective function). In this way, the number of real fitness evaluation is intelligently reduced and much computational budget is saved. The individuals evaluated by real objective function are saved into an archive dynamically. The GP model is retrained using data in the archive in every given number of generations.

Algorithm 1: Proposed s-PBIL

```
Input: Hyper-parameters
       $\{P, k, r, LR, MUT_{PROB}, MUT\}$, where $P$
       denotes the population size, $k$ indicates the size
       of subset, and $r$ denotes a ratio number;
Output: The updated probability model $\boldsymbol{M}$, the
        constructed discrete part of offspring.
1  Initialize a $|Z|$ dimensions probability model $\boldsymbol{M}$. The
     value of each dimension is 0.5;
2  Sample $P$ individuals as parent in probability $\boldsymbol{M}$;
   // Extract desired individuals
3  for each individual $p$ of parent, $i = 1$ to $P$ do
4      Initialize a vector $temp$ with $|Z|$ dimensions. The
         value of each dimension is 0;
5      Take the dimensions of value 1 in $p$, and set the
         value of corresponding dimensions in $temp$ as 1;
6      if the dimensions of value 1 in $p$ are less than the
         size of subset $k$ then
7          Select the dimensions of value 0 in $p$ with
             larger indexes, until the number of selected
             dimensions equals to $k$;
8          Set the value of corresponding dimensions in
             $temp$ as 1;
9      end
10 end
11 Evaluate and sort the parent in descending order (i.e.,
     for $min$ problem);
12 Extract the best $\lfloor r \times P \rfloor$ individuals in parents as
     learning samples;
   // Learning Rules
13 for each individual $p$ of extracted learning samples,
     $j = 1$ to $\lfloor r \times P \rfloor$ do
14     for each dimension $i$ of $\boldsymbol{M}$, $i = 1$ to $|Z|$ do
15         Apply learning rules consisting of rectification
             using Eq. (7) and mutation using Eq. (8);
16     end
17 end
   // Sample Offspring
18 Sample the discrete part of offspring in updated
     probability $\boldsymbol{M}$;
19 Return the updated probability model $\boldsymbol{M}$, the
     constructed discrete part of offspring;
```

V. Experimental Studies

In this section, we present and analyse the performance of the proposed methods with extensive simulations. On the one hand, we build real-world deployment scenarios, scaling from small-size to large-size to compare several algorithms. On the other hand, we verify the effectiveness and efficiency of surrogate-assisted optimization.

A. Real-World Deployment Problem

The proposed approach is applied to solve a real-world deployment problem. We apply an open-source[2] DEM to

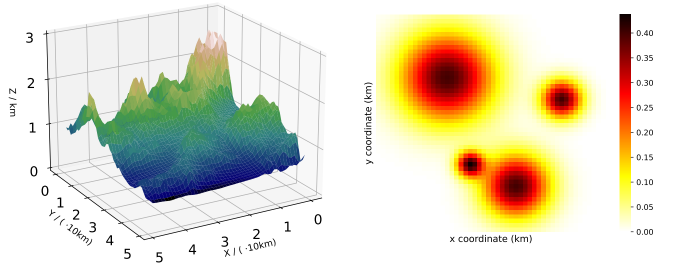
[2]The open-source DEM is acquired at https://earthexplorer.usgs.gov/.

TABLE II

RESULTS OF THE COMPARISON ON THE OBJECTIVE VALUE UNDER DIFFERENT PROBLEM SIZES

Problem Size	s-PBIL + SLPSO	r-EDA + SLPSO	Swap_opt + SLPSO	Random + SLPSO
Small-size	**0.0807** $\pm$ **0.0125**	0.1506 $\pm$ 0.0187 (+)	0.1014 $\pm$ 0.0054 (+)	0.1975 $\pm$ 0.0217 (+)
Medium-size	**0.0534** $\pm$ **0.0094**	0.1080 $\pm$ 0.0192 (+)	0.0816 $\pm$ 0.0149 (+)	0.1548 $\pm$ 0.0256 (+)
Large-size	**0.0640** $\pm$ **0.0135**	0.0962 $\pm$ 0.0089 (+)	0.0651 $\pm$ 0.0172 ($\approx$)	0.1182 $\pm$ 0.0169 (+)

Remark: Mean $\pm$ Std, and the best performance are **bold**. '+', '$-$', and '$\approx$' indicate that the results of the algorithm are significantly better than, worse than, and similar to the ones of s-PBIL + SLPSO by Wilcoxon's rank sum test with significance level 0.05.



(a) A real-world DEM (b) Heatmap of threat values

Fig. 5. A real-world deployment problem. (a) A 50$\times$50km open-source DEM discretized with 1km resolution. (b) Heatmap of threat values of targets. There exist four hot spots: $x = 15, y = 35$ with $\sigma = 7$, $x = 20, y = 15$ with $\sigma = 2$, $x = 30, y = 10$ with $\sigma = 5$, and $x = 40, y = 30$ with $\sigma = 3$.

conduct the simulation as shown in Fig. 5(a). The deployment space is 50$\times$50km. It is discretized with 1km resolution. We employ a probabilistic directional sensing model which is detailed in Section III-B. In our setting, we possess 10 sensors in total. The parameters of each sensors are $\beta_d = 1$, $\beta_p = 0.15$, $\beta_t = 0.15$, $t_d = 30$, $t_p = 60$, and $t_t = 60$.

1) Problem size: We make a preliminary analysis in Section III-C that the decision variables include both binary and continuous ones. To be more specific, the deployment problem is a subset selection problem coupled with continuous optimization. Since the inherent combinatorial property of subset selection is much more challenging and of our focus, we discriminate the small-size from large-size problem via the scale of subset selection problem. The details of the problem size are as follows

- Small-size problem: We possess 25 randomly generated candidate positions and 10 sensors.
- Medium-size problem: We possess 64 randomly generated candidate positions and 10 sensors.
- Large-size problem: We possess 100 randomly generated candidate positions and 10 sensors.

2) Targets generator: Target set Q denotes the coverage requirements, in which each target q is associated with a coordinate (x_q, y_q, z_q) and a threat value w_q. Since we consider a 3-D realistic scenario, we divide the space into 3 different levels with heights of 2km, 10km and 20km similar to [6]. The corresponding weights of 3 levels could be customized, while we define 1.0, 1.0, and 1.0 in this paper.

Further, we generate targets for each level similarly. In the real-world, there exist several important places such as command centres, transportation stations, and bridges. We treat them as hot spots. If a target is close to a hot spot, its threat value should be larger. To reasonably set their threat values, we introduce modified Gaussian distribution

$$t(d) = \frac{1}{\sqrt{2\pi}} exp\left(-\frac{d^2}{2\sigma^2} \right), \tag{11}$$

where d denotes the Euclidean distance between a target and a hot spot. σ denotes the importance of a hot spot. A larger σ indicates a more important hot spot. Several hot spots generate a joint distribution and reveal a heatmap of threat values, as shown in Fig. 5(b).

In each height level, we sample 100 targets that are evenly spaced. Therefore, the target set Q contains 300 incoming targets in total. We note that the number of targets does not have an adverse effect on the deployment approach, but the computational time needed for the deployment increases proportionally with more targets. We show the computational time and introduce a surrogate-assisted scheme to alleviate the low optimization efficiency with more targets in Section V-E.

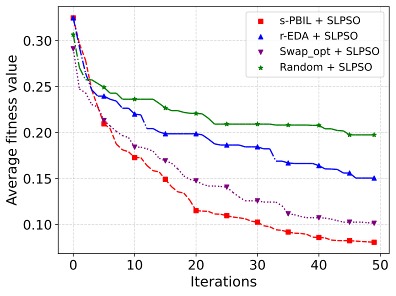
B. Algorithms setting

The population size P and the iteration number are set to 200 and 50, respectively. The last term in SLPSO representing learning from collective behaviour is not included for parameter efficiency in this paper. Latin-hypercube sampling [28] is used to initialize the population for SLPSO. The ratio r is set according to a self-adaptive mechanism, which is $r = P^{-\frac{1}{2}} = 200^{-\frac{1}{2}} \approx \frac{14}{200}$ empirically. The learning rate LR, the mutation probability MUT_{PROB}, and the amount for mutation MUT are set to 0.3, 0.1, and 0.05, respectively. GP incorporated in this paper is implemented based on SMT 2.0 [29] and its initial θ is set as 0.1. We conduct 10 independent runs and collect statistical results for each algorithm.

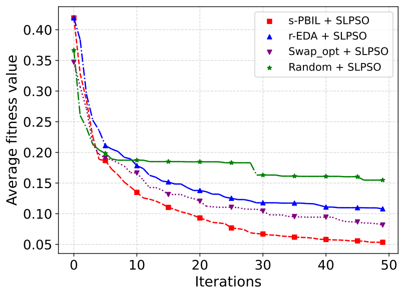
C. Comparison algorithms

We compare the proposed approach with three algorithms. We note that the settings of SLPSO are the same. The details of the compared methods are as follows

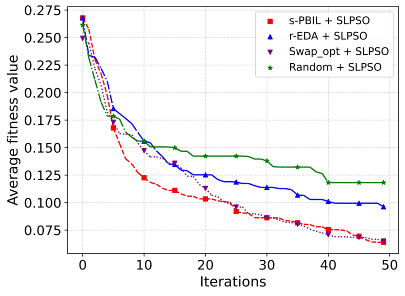
- Random + SLPSO: The positions of sensors are randomly selected in finite candidate positions. The other parameter settings are the same as the proposed approach.



(a) Small-size problem

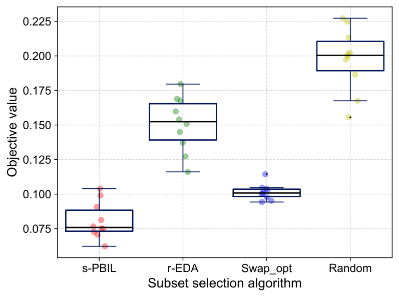


(b) Medium-size problem

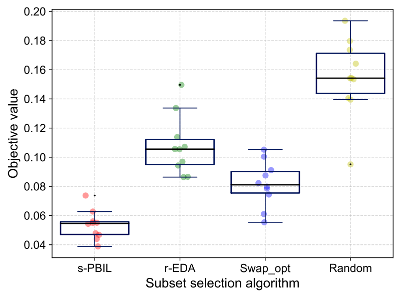


(c) Large-size problem

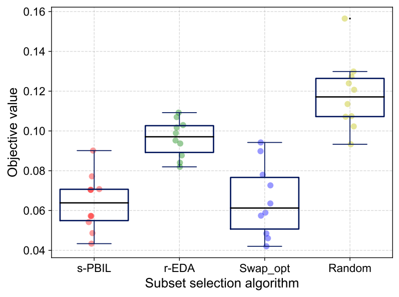
Fig. 6. Convergence profiles of the proposed approach and three comparison algorithms under different problem sizes.



(a) Small-size problem



(b) Medium-size problem



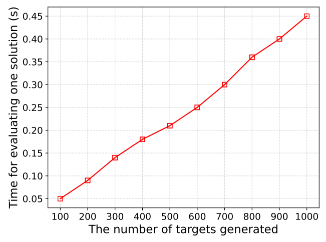
(c) Large-size problem

Fig. 7. Solution quality of the proposed approach and three comparison algorithms under different problem sizes.

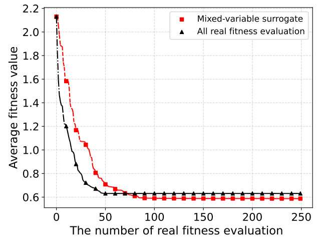
- Swap_opt + SLPSO: We also design an efficient heuristic approach swap_opt to handle subset selection problem. The swap_opt algorithm first randomly initializes a subset of size k in binary coding and then applies two strategies to improve the solution. The first strategy is to swap two positions iteratively, while the second strategy shifts the binary coding one bit to the right with a probability of 0.1. A population is maintained and the elitist is kept during the iterations. The other parameter settings are the same as the proposed approach.
- r-EDA + SLPSO: Inspired by [30], we customize roulette estimation of distribution (r-EDA) to solve subset selection problem. In each iteration, the learned probability vector is normalized via the sum of all values. Subsequently, a k-spins roulette wheel is used to select a subset of size k. The other parameter settings are the same as the proposed approach.

D. Experiment results

Fig. 6 presents the convergence profiles of the proposed approach and three comparison algorithms under different problem sizes. All subfigures presented in Fig. 6 show that the proposed approach outperforms compared schemes. The proposed method can converge faster and find solutions of better quality. Further, Table II and Fig. 7 show the statistical results of algorithms. We can observe that s-PBIL achieves the best average objective values under three problem sizes. The average values of Swap_opt under large-size problem is



(a) Time for evaluating one solution



(b) Convergence profiles

Fig. 8. More targets (i.e., 1,875 targets) in the small-size scenario.

relatively close to the values of s-PBIL. However, s-PBIL still gains improvement and achieves a low standard deviation.

E. Surrogate-assisted optimization with more targets

Fig. 8(a) shows the time for evaluating one solution versus the number of targets generated. We can see that the computational time needed for the deployment increases proportionally with more targets. If we have more targets, a larger population and more iterations, the optimization efficiency is insufficient for timely decision-making. We present a surrogate-assisted scheme to alleviate the low optimization efficiency with more targets (i.e., 1,875 targets generated in the small-size scenario). Fig. 8(b) shows that the mixed-variable surrogate under Bayesian model management converges to a better solution in the same number of real fitness evaluation. It converges slower in the previous step since the mixed-variable surrogate with unifying space is trained via initial low-quality individuals to depict the optimization landscape.

VI. Conclusion

In this paper, we present a surrogate-assisted hybrid metaheuristic for mixed-variable 3-D deployment optimization of DSNs. In detail, hybrid metaheuristic for mixed-variable, s-PBIL, and mixed-variable surrogate are designed for dealing with realistic MINLP model, subset selection, and the expensive computational cost. Experiment results show the effectiveness of the proposed algorithm.

In the future, the deployment problem in this paper could be enriched and benchmarked as a real-world expensive mixed-variable test suite. Since there exist various off-the-shelf techniques for continuous optimization, we focus more on efficient subset selection algorithms and surrogate-assisted schemes for high-dimensional problems [31]. The proposed s-PBIL and several comparison schemes could also provide the opportunity for the feature selection problem in the Machine Learning community [32]. In addition, the proposed approaches are potential for tackling real-world tasks, such as military surveillance [33] and smart city [34], [35] applications.

References

[1] S. Ding, C. Chen, Q. Zhang, B. Xin, and P. M. Pardalos, *Metaheuristics for resource deployment under uncertainty in complex systems*. CRC Press, 2021.

[2] M. R. Senouci and A. Mellouk, *Deploying wireless sensor networks: theory and practice*. Elsevier, 2016.

[3] T. Tanergüçlü, H. Maraş, C. Gencer, and H. Aygüneş, "A decision support system for locating weapon and radar positions in stationary point air defence," *Information Systems Frontiers*, vol. 14, no. 2, pp. 423–444, 2012.

[4] E. Yildiz, K. Akkaya, E. Sisikoglu, and M. Y. Sir, "Optimal camera placement for providing angular coverage in wireless video sensor networks," *IEEE Transactions on Computers*, vol. 63, no. 7, pp. 1812–1825, 2013.

[5] Y. Zou and K. Chakrabarty, "Sensor deployment and target localization based on virtual forces," in *IEEE INFOCOM 2003. Twenty-second Annual Joint Conference of the IEEE Computer and Communications Societies*, vol. 2, 2003, pp. 1293–1303 vol.2.

[6] X.-y. Lian, J. Zhang, C. Chen, and F. Deng, "Three-dimensional deployment optimization of sensor network based on an improved particle swarm optimization algorithm," in *Proceedings of the 10th World Congress on Intelligent Control and Automation*. IEEE, 2012, pp. 4395–4400.

[7] V. Akbarzadeh, C. Gagne, M. Parizeau, M. Argany, and M. A. Mostafavi, "Probabilistic sensing model for sensor placement optimization based on line-of-sight coverage," *IEEE Transactions on Instrumentation and Measurement*, vol. 62, no. 2, pp. 293–303, 2012.

[8] S. Ding, C. Chen, J. Chen, and B. Xin, "An improved particle swarm optimization deployment for wireless sensor networks," *Journal of Advanced Computational Intelligence and Intelligent Informatics*, vol. 18, no. 2, pp. 107–112, 2014.

[9] X. Zhang, Y. Zhou, Q. Zhang, V. C. Lee, and M. Li, "Problem specific moea/d for barrier coverage with wireless sensors," *IEEE Transactions on Cybernetics*, vol. 47, no. 11, pp. 3854–3865, 2016.

[10] B. Cao, J. Zhao, P. Yang, P. Yang, X. Liu, and Y. Zhang, "3-d deployment optimization for heterogeneous wireless directional sensor networks on smart city," *IEEE Transactions on Industrial Informatics*, vol. 15, no. 3, pp. 1798–1808, 2018.

[11] Y. Y.-H. Kim, "Maximizing the coverage of sensor deployments using a memetic algorithm and fast coverage estimation," *IEEE Transactions on Cybernetics*, pp. 6531–6542, 2022.

[12] A. S. R. S. Benyattou, "Toward a realistic approach for the deployment of 3d wireless sensor networks," *IEEE Transactions on Mobile Computing*, pp. 1508–1519, 2022.

[13] S. Ding, T. Zhang, C. Chen, Y. Lv, B. Xin, Z. Yuan, R. Wang, and P. M. Pardalos, "An efficient particle swarm optimization with evolutionary multitasking for stochastic area coverage of heterogeneous sensors," *Information Sciences*, p. 119319, 2023.

[14] Y. Liu and H. Wang, "Surrogate-assisted hybrid evolutionary algorithm with local estimation of distribution for expensive mixed-variable optimization problems," *Applied Soft Computing*, vol. 133, p. 109957, 2023.

[15] M. Gendreau, J.-Y. Potvin *et al.*, *Handbook of metaheuristics*. Springer, 2010, vol. 2.

[16] S. Baluja, "Population-based incremental learning. a method for integrating genetic search based function optimization and competitive learning," Carnegie-Mellon Univ Pittsburgh Pa Dept Of Computer Science, Tech. Rep., 1994.

[17] Z. Fei, B. Li, S. Yang, C. Xing, H. Chen, and L. Hanzo, "A survey of multi-objective optimization in wireless sensor networks: Metrics, algorithms, and open problems," *IEEE Communications Surveys & Tutorials*, vol. 19, no. 1, pp. 550–586, 2016.

[18] F. Zhou, J. Gao, X. Fan, and K. An, "Covering algorithm for different obstacles and moving obstacle in wireless sensor networks," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3305–3315, 2018.

[19] J. Kennedy and R. Eberhart, "Particle swarm optimization," in *Proceedings of ICNN'95-International Conference on Neural Networks*, vol. 4. IEEE, 1995, pp. 1942–1948.

[20] X. Wang, H. Zhang, S. Fan, and H. Gu, "Coverage control of sensor networks in iot based on rpso," *IEEE Internet of Things Journal*, vol. 5, no. 5, pp. 3521–3532, 2018.

[21] F. Wang, H. Zhang, and A. Zhou, "A particle swarm optimization algorithm for mixed-variable optimization problems," *Swarm and Evolutionary Computation*, vol. 60, p. 100808, 2021.

[22] B. Cao, J. Zhao, Z. Lv, and X. Liu, "3d terrain multiobjective deployment optimization of heterogeneous directional sensor networks in security monitoring," *IEEE Transactions on Big Data*, vol. 5, no. 4, pp. 495–505, 2017.

[23] R. Cheng and Y. Jin, "A social learning particle swarm optimization algorithm for scalable optimization," *Information Sciences*, vol. 291, pp. 43–60, 2015.

[24] C. Shen, K. Zhang, and J. Tang, "A covid-19 detection algorithm using deep features and discrete social learning particle swarm optimization for edge computing devices," *ACM Transactions on Internet Technology (TOIT)*, vol. 22, no. 3, pp. 1–17, 2021.

[25] G. Liu, X. Chen, R. Zhou, S. Xu, Y.-C. Chen, and G. Chen, "Social learning discrete particle swarm optimization based two-stage x-routing for ic design under intelligent edge computing architecture," *Applied Soft Computing*, vol. 104, p. 107215, 2021.

[26] E. C. Garrido-Merchán and D. Hernández-Lobato, "Dealing with categorical and integer-valued variables in bayesian optimization with gaussian processes," *Neurocomputing*, vol. 380, pp. 20–35, 2020.

[27] Y. Jin, H. Wang, and C. Sun, *Data-driven evolutionary optimization*. Springer, 2021.

[28] M. D. Shields and J. Zhang, "The generalization of latin hypercube sampling," *Reliability Engineering & System Safety*, vol. 148, pp. 96–108, 2016.

[29] P. Saves, R. Lafage, N. Bartoli, Y. Diouane, J. Bussemaker, T. Lefebvre, J. T. Hwang, J. Morlier, and J. R. Martins, "SMT 2.0: A surrogate modeling toolbox with a focus on hierarchical and mixed variables gaussian processes," *arXiv preprint arXiv:2305.13998*, 2023.

[30] J. L. Shapiro, "Drift and scaling in estimation of distribution algorithms," *Evolutionary Computation*, vol. 13, no. 1, pp. 99–123, 2005.

[31] X. Cai, L. Gao, and X. Li, "Efficient generalized surrogate-assisted evolutionary algorithm for high-dimensional expensive problems," *IEEE Transactions on Evolutionary Computation*, vol. 24, no. 2, pp. 365–379, 2019.

[32] I. Guyon and A. Elisseeff, "An introduction to variable and feature selection," *Journal of Machine Learning Research*, vol. 3, no. Mar, pp. 1157–1182, 2003.

[33] J. Chen, J. Sun, and G. Wang, "From unmanned systems to autonomous intelligent systems," *Engineering*, vol. 12, pp. 16–19, 2022.

[34] F. D. D. Y. C. Chen, "Wearable ubiquitous energy system," *Science China Information Sciences*, pp. 237–239, 2021.

[35] S. Yuan, H. Wang, and L. Xie, "Survey on localization systems and algorithms for unmanned systems," *Unmanned Systems*, vol. 9, no. 02, pp. 129–163, 2021.