International Journal of Smart Computing and Artificial Intelligence
International Institute of Applied Informatics
2023, Vol. 7, No. 2, IJSCAI793

An Extension of Particle Swarm Optimization to Identify Multiple Peaks using Re-diversification in Static and Dynamic Environments

Stephen Raharja [*], Toshiharu Sugawara [*]

Abstract

We propose an extension of the particle swarm optimization (PSO) algorithm for each particle to store multiple global optima internally for identifying multiple (top-*k*) peaks in static and dynamic environments. We then applied this technique to *search and rescue* problems of rescuing potential survivors urgently in life-threatening disaster scenarios. With the rapid development of robotics and computer technology, aerial drones can be programmed to implement search algorithms that locate potential survivors and relay their positions to rescue teams. We model an environment of a disaster area with potential survivors using randomized bivariate normal distributions. We extended the *Clerk-Kennedy PSO* algorithm as *top-k PSO* by considering individual drones as particles, where each particle remembers a set of global optima to identify the top-*k* peaks. By comparing several other algorithms, including the canonical PSO, Clerk-Kennedy PSO, and *NichePSO*, we evaluated our proposed algorithm in static and dynamic environments. The experimental results show that the proposed algorithm was able to identify the top-*k* peaks (optima) with a higher success rate than the baseline methods, although the rate gradually decreased with increasing movement speed of the peaks in dynamic environments.

Keywords: Meta-heuristic algorithm, Particle swarm optimization, Top-k multiple peaks, Search and rescue in disasters, Multiple optima

1 Introduction

As part of an integrated emergency response to disasters such as earthquakes, search teams must find and rescue survivors as soon as humanly possible, because their lives are typically in danger, i.e., survival rates decline within three days after a disaster. In particular, the first few hours after an incident are the most important time period [32]. In this context, autonomous aerial drones or robots can be applied to locate such survivors by searching the areas impacted by the disaster. For example, three types of aerial drones were developed during a project undertaken by Japanese universities and national research institutes. One type was designed to transmit local communication, one to perform early reconnaissance of

[*] Department of Computer Science and Engineering, Waseda University, Tokyo, Japan

the area, and another to locate survivors [25]. Such tasks are suitable for autonomous drones because they are not hindered by many types of obstacles such as rubble or debris that may be present after a disaster [1]. The aforementioned problem is called *urban search and rescue* (USAR).

Drones sufficiently sophisticated to perform such tasks are precise and expensive equipment, and a variety of technicians are required to operate and maintain them. Nonetheless, multiple drones must be ready at all times, because disaster or emergency situations may occur suddenly, with little or no warning. Furthermore, they should be deployed redundantly, because some units could be destroyed or rendered inoperable by the disaster. Therefore, less expensive and simpler drones should be developed that can be easily deployed in larger quantities to locate survivors with performance equivalent to or better than conventional advanced drones.

The movements of a group of cooperative drones are often modeled on schools of fish and foraging behaviors of insects such as those of honeybees in exploring an area to find locations with more food. This pattern of behavior is modeled by the *particle swarm optimization* (PSO) algorithm [17] [16]. PSO and its variants have been studied extensively for a long time and seem well suited for environments with a unitary optimal solution [31]; they have also shown good performance in static environments [26]. However, because of the cascading propagation of information in particle networks constructed by the generic PSO algorithm, its performance may not suffice in environments with multiple simultaneous solutions, such as the USAR problem with multiple rescue locations [7]. Moreover, the particles may converge prematurely and fail to find the true optimum in the environment, which is known as the local optima trap problem [22]. Thus, a simple and effective PSO algorithm capable of identifying multiple optimal values in an environment is needed.

In research on PSO, many studies have attempted to identify multiple optimal solutions using *niching techniques* [6] [19] [27]. Brits et al. [6], for example, proposed the *NichePSO* algorithm to exploit the *guaranteed convergence PSO* (GCPSO) algorithm [33] [34] to locate multiple or all optimal values (peaks) in an environment. However, neither of these methods are sufficiently simple for application in real robot swarms, nor are the probabilities of finding the required number of optimal solutions sufficiently high.

The proposed algorithm is a simple extension of the Clerk-Kennedy PSO [8], in which each particle stores a set of globally optimal values instead of a single globally optimal value. Particularly, using a set of globally optimal solutions, each particle randomly selects a convergence direction from a weighted set of alternatives, and information about the updated expected peak is communicated to other particles within a specified distance to retain diversity among particles in the swarm. If no improvement is achieved after a certain amount of time, a re-diversification strategy is introduced to re-randomize the positions of the particles. This method enables the diversification of the swarm and improved exploration of the environment. Once the terminal condition is reached, the swarm merges and processes all the values identified by all the particles and outputs the top k peaks, which correspond to the best k optimal solutions in the environment.

First, for a given environment, we model the problem using a *mixed bivariate normal distribution* with the means randomized. Based on this model, we conducted experiments to locate multiple peaks in a simulated environment with multiple static or

dynamically moving peaks. Then, we compared these results with three baseline methods, including NichePSO, Clerk-Kennedy PSO, and the canonical PSO. The experimental results show that the proposed algorithm enables particles to find the top-k peaks effectively with probability higher than the baseline methods in static environments and in dynamic environments although the rate of findings gradually decreased according to the increase of movement speed. Last, we discuss the strengths and weaknesses of our proposed method.

2 Related Work

The basic problem solved by PSO algorithms is that of identifying a single global optimum/solution in an environment. During that process, the swarm may encounter the local optimal trap problem, which has been studied extensively [35] [2] [28]. One of many approaches to solve the local trap problem is to use a collection of various PSO algorithms, as proposed by Engelbrecht in *heterogeneous PSO* (HPSO) [12]. At the initialization stage, each particle randomly chooses a PSO algorithm from a collection of PSO models, such as barebones PSO, modified barebones PSO [15], a social-only model [14], and a cognitive-only model. In the former, if the local optimum of particles have not been updated for some time, the particles would randomly rotate to a different algorithm from the given collection. In the latter, algorithms chosen by each particle remain unchanged. HPSO was also extended to handle dynamic environments [18].

In an optimization problem using *genetic algorithms* (GA), niching techniques have been studied extensively to identify multiple solutions, and the same tech- niques have also been applied in PSO algorithms to identify multiple optima. This approach was called NichePSO [6], in which a cognitive-only model is used by the main particles to locate initial optima in an environment, and multiple GCPSO sub-swarms [33, 34] are then formed around individual optima. If a particle or a sub-swarm moves within the radius of another sub-swarm, it is absorbed by the latter. Despite its performance, NichePSO has been shown to lose its diversity over time because all sub-swarms may eventually fuse into a single large swarm [10]. In contrast, comparing NichePSO, our proposed method top-*k* PSO retains the communication network within the swarm, which enables enhanced exploration of the environment.

In addition to niching techniques, many other approaches to identifying multiple optima in an environment have been developed, such as *galactic swarm optimization* (GSO) [24], and its extension using *whale optimization algorithm* (WOA) [13]. Exploration and exploitation of the environment in GSO is balanced by dividing the swarm into two levels, super-swarm and sub-swarm. During the first phase, each sub-swarm finds its global optimal value by performing exploration in the environment using the canonical PSO. Next, in the second phase, the super-swarm exploits the environment by using each global optimal value from the sub-swarms as seed values, which are then input to a separate canonical PSO algorithm. No- tably, the GSO algorithm is reported to be flexible, and any swarm-based algorithm can replace the canonical PSO used in the experiments.

Applying PSO to a dynamic environment has proven challenging due to many problems such as moving peaks, outdated memory, and loss of swarm diversity [4].

The *dynamic multi-swarm fractional-best particle swarm optimization* (DMSFPSO)[11] approach was developed to solve multi-model problems, and implemented both adaptive sub-swarm count and multi-swarm techniques. Moreover, when a change in the environment is detected, particle velocities are re-initialized to deal with the outdated memory problem. Furthermore, to retain diversity, sub-swarms repel each other. In contrast, considering that multiple optima might exist in a small area, we did not implement a repelling action in particles of the top-*k* PSO swarm. However, our proposed approach does implement a re-diversification method to reduce the probability of being affected by the local trap problem. Moreover, in contrast to both GSO and DMSFPSO, the objective of top-*k* PSO is to identify only a subset of all optima in the environment.

Finally, although we previously reported an extended PSO to find multiple peaks [29], we also demonstrate that the proposed method can find multiple peaks in a dynamic environment where some peaks gradually move over time.

3 Background and Problem Description

3.1 Clerk-Kennedy PSO

Here, we briefly explain the basis of our proposed method, which is the Clerk- Kennedy PSO algorithms [9] [37]. Clerk-Kennedy PSO is different from the canon- ical PSO in that instead of using acceleration constants, it uses a *constriction rate* to gradually convert particles from the task of exploration to exploitation, which is the same for both global and local attractors.

In Clerk-Kennedy PSO, the update formulas for the position $x_i(t+1) \in V$ of particle p_i and for its velocity $v_i(t+1)$ are defined by Eq. 2 and 1, respectively.

$$x_i(t+1) = x_i(t) + v_i(t+1) \text{ and,} \tag{1}$$

$$v_i(t+1) = \alpha(c \cdot r_1(t)(y_i(t) - x_i(t)) + c \cdot r_2(t)(g(t) - x_i(t))), \tag{2}$$

where $g(t)$ is the global best position, $y_i(t)$ is p_i's local best position, $x_i(t)$ is p_i's current position, $0 < r_1(t), r_2(t) < 1$ are random numbers at t, α is the constant of the constriction rate and c is the acceleration constant. Both constants, α and c, are positive with limitation of $0 < \alpha < 1$ and $c > 0$.

3.2 Model of Environment

We modeled a disaster area as a simulated environment with the assumption that all locations in the environment were equally important, because the potential num- ber of survivors is typically not known beforehand. Moreover, prior works have shown that survivors tend to move toward important locations from a social view- point after a disaster [21]. Tracking people's movement is also possible, such as by using mobile phone signals [3] or cellular base stations [8]. Assuming that viable methodologies are available to detect survivors, an aerial drone can sort areas based on their respective probability of containing survivors. Thus, the modeled problem is a two-dimensional (bivariate) normal distribution expressing the probability of finding a survivor within a certain area. Each of the center points in concentrated areas of survivors is represented as a mean $\boldsymbol{\mu}$ and the probability of existence of each peak is the value of the density function fi(μ). Its covariance matrix Σ expresses the density or the spread of each concentrated area. We describe it in more detail below.

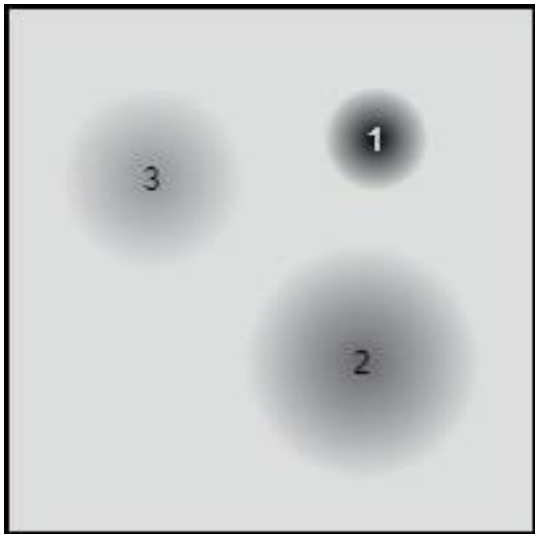

Figure 1: Top view of an environment with three peaks [29]

In our model, time is considered as discrete and calculated in unit of ticks. Let P be the n particles of the PSO swarm, i.e., the set of particles, where n is a positive integer. Each tick, a particle $p_i \in P$ in the PSO swarm moves within the environment to find peaks that correspond to the optimal solutions, by exchanging information with surrounding particles $p_j \in P$. A finite, square-shaped, two-dimensional space $V \subset R^2$ (where R is the set of real numbers) is used to represent the environment, with the length of the sides of the environment being $2 \cdot E$. Thus, any point $x \in V$ can be represented as $x = (z_1, z_2)$ such that $-E \leq z_1, z_2 \leq E$ for a positive number $E \in R$. N different Gaussian distributions $N(\mu_i, \Sigma_i)$ for $i = 1, \ldots, N$ are used to represent N peaks in V . For each peak, the center location of i-th peak is represented as vector $\mu_i = (\mu_{1,i}, \mu_{2,i})$ ($\in V$), and covariance between the x-axis and y-axis for i-th peak is represented by the 2×2 diagonal matrix Σ_i, the elements of which are in the closed interval of [0, 1]. Thus, the utility value of any location $x = (z_1, z_2) \in V$ to the i-th peak, or the probable number of survivors, can be formulated as

$$f_i(x) = \frac{1}{2\pi\sigma_{1,i}\sigma_{2,i}} e^{-\frac{1}{2}((\frac{z_1-\mu_{1,i}}{\sigma_{1,i}})^2+(\frac{z_2-\mu_{2,i}}{\sigma_{2,i}})^2)}, \tag{3}$$

where $\sigma_{1,i}$ and $\sigma_{2,i}$ are the standard deviations of two variables. The utility value of a single i-th peak can be calculated by $f_i(\mu_i)$.

Adding all distribution functions yields the utility value of the mixed distribution functions r(x) of $x \in V$, which is defined as

$$r(x) = \sum_{i=1}^{N} m_i \cdot f_i(x) \tag{4}$$

where $f_i(x)$ is the distribution function of the *i*-th value and $m_i > 0$ denotes its expected *area importance*; this represents special situations such as the urgency of a

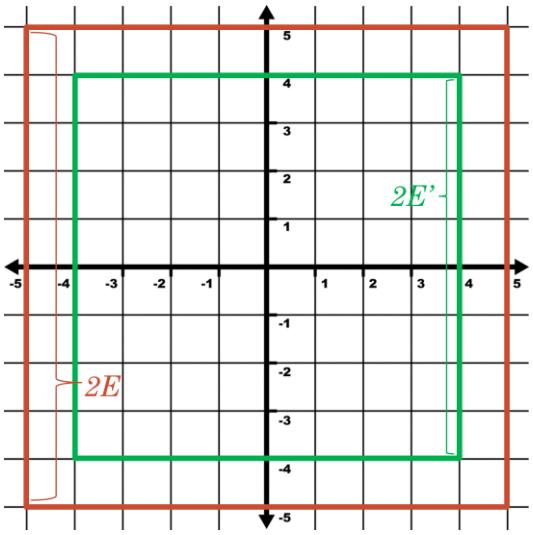

Figure 2: Visualization of areas in an environment with $E = 5$ and $E' = 4$.

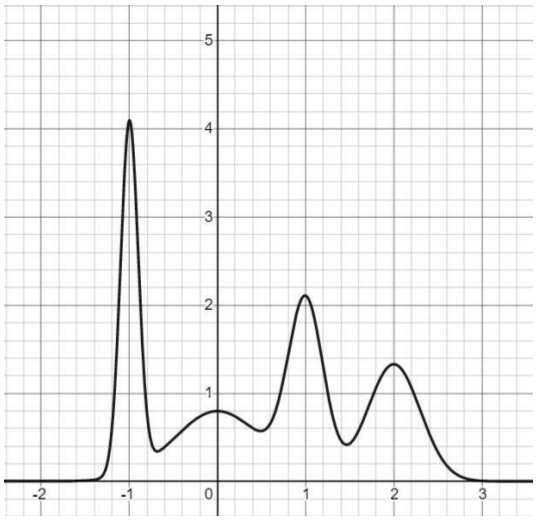

Figure 3: One-dimensional representation of utility values with four peaks [29]

given rescue or the expected number of survivors. If data regarding area importance in the environment are unavailable, all m_i are equalized. Fig. 1 shows a top view of a sample environment (Fig. 2) with 3 peaks, and a simplified visualization of the utility values in one-dimensional environment is shown in Fig. 3.

We introduce static and dynamic environments with the following natural assumptions to evaluate our proposed method.

- $|P| > N$, i.e., the number of particles is greater than the number of peaks in the environment.
- The *peak generation area* is smaller than the environment area and there is a margin between the boundaries of the environment and the peak generation area. This margin size is specified by E' ($\leq E$) as shown in Fig. 2.
- There is no/negligible communication overhead when particles propagate information to other particles within the swarm.

Note that the peak generation area is defined as the area where all peaks exist. The first assumption is intuitive given USAR is the target application. It is certainly plausible that our proposed algorithm would perform reasonably well without this assumption, but to simplify our experiment we decided to add this assumption because it would be difficult to find the minimum number of particles that would still guarantee good performance in any environment [20].

The second assumption was introduced to allow all peaks to be explored from all directions, because exploring a peak only a limited direction is difficult for par- ticles in a swarm, in all PSO algorithms, although reducing the margin between the two areas can increase the area that can be explored in the environment. Further discussion is provided in Section 5.3. The third assumption holds because in the premise of our proposed method, aerial drones would use wireless communication technologies to exchange information within the swarm. Considering that the infor- mation payload is small and the number of particles is not large, it is safe to assume that information overhead is negligible.

3.2.1 Static Environment

In static environments, the locations $\boldsymbol{\mu}_i$ and peak values $f_i(\boldsymbol{\mu}_i)$ of all peaks are considered to remain constant.

3.2.2 Dynamic Environment

In dynamic environments, only the peak values f_i ($\boldsymbol{\mu}_i$) of all peaks remain constant. The position of a peak at time t can be defined as $\boldsymbol{\mu}_i$ (t) = ($\mu_{1,I}$, $\mu_{2,i}$) ($\in V$). After each tick, each peak moves a random distance in random direction; the movement of the peaks is represented as a vector $\boldsymbol{\phi}_i$ with randomized elements in the range ($-E/\omega$, E/ω), where $\omega > 0$ is the *movement speed factor* to determine the movement speed of peaks. Thus, the next position of a peak can be given by $\boldsymbol{\mu}_i(t+1) = \boldsymbol{\mu}_i(t) + \boldsymbol{\phi}_i$.

We adopt a fully randomized movement of peaks in the environment to simulate the chaotic movements of survivors after a disaster, and the upper and lower bound of peak

movements are applied because survivors can only move up to a certain distance within a given time frame. However, the position of each peak must be within the boundary of the peak generation area of the environment $-E' \leq \mu_{1,i}, \mu_{2,i} \leq E'$ to ensure the margin $E - E' \geq 0$.

Given that the peaks move randomly, although the mean values $\boldsymbol{f}(\boldsymbol{\mu}_i(t))$ of each peak i at time t will always remain constant, the total utility value $r(x)$ that can be calculated in a position x will fluctuate unpredictably throughout the experiment. This could happen as two groups of survivors encounter each other, producing an area with higher probability of finding survivors within the join boundary for drones to identify. Even if those two groups of survivors merge into a larger one, the problem would be reduced to finding top-k peaks from $N-1$ peaks, and our proposed method would still be applicable.

3.3 Problem Formulation

Given that the main priority of USAR teams is to rescue as many people as possible without considering their social importance or other status, USAR teams need to quickly explore areas indicated by the aerial drones as having the highest probable concentrations of survivors. However, USAR teams may be forced to prioritize areas with the greatest probability of finding survivors owing to limited resources.

Let $L = \{\boldsymbol{\mu}_1, \ldots, \boldsymbol{\mu}_N\}$ be a set sorted in descending order based on the utility value $r(\boldsymbol{\mu}_i)$. For the purpose of our problem, the first k ($\leq N$) peaks from L need to be identified. A successful identification of a peak $\boldsymbol{\mu}_j$ at $x \in V$ by a particle p_i requires these two conditions: (1) the difference in utility value between an optimum found by p_i at x and $\boldsymbol{\mu}_j$ must satisfy

$$\frac{|r(\boldsymbol{\mu}_j) - r(x)|}{r(\boldsymbol{\mu}_j)} \leq \delta_u$$

and (2) the Euclidean distance must be small; to be precise, $dist(\boldsymbol{\mu}_j, x) \leq \delta_d$. Both parameters $\delta_u \geq 0$ and $\delta_d \geq 0$ represents threshold of closeness, and both are small positive numbers. The current location of p_i is represented as $\mathbf{x}(p_i)$.

4 Proposed Method

To solve the USAR problem, we extend the Clerk-Kennedy PSO to create a novel method *top-k PSO*. Clerk-Kennedy PSO is the basis of our method because it already includes a pair of constants known to be effective [9], which obviated the necessity of conducting additional experiments to find optimal constants. Instead of storing a single global optimal (peak) as in the canonical PSO, particle $p_i \in P$ in top-k PSO keeps the set of positions of possible global peaks, $G_i = \{g_{i,1}, g_{i,2}, \ldots, g_{i,k}\}$, where $g_i = (z_1, z_2)$ and the utility values of peaks in G_i are obtained by $r(g_i)$. Algorithm 1 describes the top-k PSO algorithm in pseudo-code. The main modifications in top-k PSO are the *global optima value step* (Lines 23-28) and the re-diversification method(Lines30-34)usedtoreduce susceptibility to the local trap problem. As top-k PSO is based on Clerk-Kennedy PSO, Lines 12-22 are almost identical to the base method. Exploration of the environment continues until $t = T_f$ is satisfied,

Algorithm 1 Top-k PSO Algorithm

1: // Initializing swarms
2: **for** each particle $p_i \in P$ **do**
3: x_i: current location;
4: y_i: local peaks location;
5: Randomize initial position x_i and velocity v_i;
6: Calculate $r(x_i)$;
7: $y_i = x_i$;
8: $G_i = \{y_i\}$; // Set of (current) global peaks in p_i;
9: $g_i = y_i$: chosen global attractor;
10: **end for**
11: // Exploration
12: **while** $t \leq T_f$ // Until reaching terminal condition **do**
13: **for** each particle $p_i \in P$ **do**
14: Calculate next velocity $v_i(t + 1)$ using Eq. 2;
15: Update position $x_i(t + 1)$ using Eq. 1;
16: **if** $r(y_i) < r(x_i(t + 1))$ **then**
17: $y_i = x_i(t + 1)$;
18: **end if**
19: **if** $r(g_i) < r(y_i)$ **then**
20: Substitute g_i in G_i for y_i.
21: Set $g_i = y_i$;
22: **end if**
23: **if** Particle p_i has no improvement in last γ_1 ticks **then**
24: Announce G_i to surrounding particles;
25: $G_i \leftarrow G$-update(p_i) in Alg. 2;
26: $g_i \leftarrow$ an element in G_i selected with probability distribution defined by Eq. 5 as new direction;
27: Randomize next velocity $v_i(t + 1)$;
28: **end if**
29: **end for**
30: **if** All $p_i \in P$ have no improvements of y_i in last γ_2 ticks **then**
31: **for** each particle $p_i \in P$ **do**
32: Randomize $x_i(t + 1)$ and $v_i(t + 1)$;
33: **end for**
34: **end if**
35: $t = t + 1$; // Also move each peak $\boldsymbol{\mu}_i$ in dynamic environment.
36: **end while**
37: Output top k peaks using function Merging in Alg. 3;

where T_f is the maximum time of the simulation and $T_f > 0$, which is a positive integer.

To ensure that the swarm can identify multiple peaks, we modified the global optimal value update step. Depending on difference in velocity compared to the previous step, $v(t + 1) - v(t)$, a particle may not exactly match with the center of the target peak μ_i. Hence, particle convergence is assumed only after γ_1 ticks with no improvement; then, particle p_i propagates G_i to its surrounding local particles (Line 23). Additionally, the requirement of convergence before propagating G_i is to prevent premature announcement of its current location x_i; premature announce- ment would attract surrounding local particles to a sub-optimal location. After propagation, a new exploration direction is selected by particle p_i (Line 26). In contrast to the canonical and Clerk-Kennedy PSOs, in our proposed top-k PSO the propagation of candidate peak values G_i is limited to nearby particles (particles the Euclidean distance of which is $\leq 2 \cdot E/k^2$). Limiting the G_i propagation area reduces the loss of swarm diversity and reduces the influence of distant peaks, which improve particles' ability to explore local areas.

There is a possibility that two peaks $g, g' \in G_i$ and $g \neq g'$, but $dist\ (g, g')$ is negligible. Thus, we consider two peaks g and g' as virtually identical, which is expressed by $g \sim g'$, if these conditions $|r(g) - r(g')| \leq \delta_{eqv}$ and $dist(g, g') \leq \delta_{eqv}$ are fulfilled, where δ_{eqv} is a small positive number determining the resolution of exploration and we set $\delta_{eqv} = 10^{-4}$ in our experiment below. Therefore, to maintain unique elements in set G_i, when $g \sim g'$, one of these elements is removed from set G_i and G in Algs. 1, 2, and 3.

Algorithm 2 Function G-update(p_i)

1: Let G be the list of global peaks G_j from close particles;
2: $G_i \leftarrow (G_i \cup_{G \in \mathrm{G}} G) \cup y_i$;
3: Sort G_i in descending order of $r(g_i)$.
4: **if** $|G_i| > k$ **then**
5: $G_i \leftarrow$ the first k elements in G_i;
6: **end if**
7: **return** G_i

Algorithm 3 Function Merging()

1: $G = \cup_{i=1}^{\eta} G_i$; // Retrieve all global peaks from all p_i.
2: Sort G in descending order of $r(g)$ for $g \in G_i$;
3: $R \leftarrow$ the first k elements in G;
4: Return R;

In top-k PSO, a particle p_i must identify the top k locations that are indistinguishable to the first k peaks in the sorted set L. During information propagation, a particle p_i receives potential global peaks G_j locations from surrounding close particles and merges them to update G_i; G_j is transmitted by converged particles p_j within a distance of $2 \cdot E/k^2$ to particle p_i with $i \neq j$ using function G-update(p_i) in Alg. 2 (Line 25 in Alg. 1). Next, a global peak candidate $g_i \in G_i$ is randomly selected by particle p_i as a new global attractor, which would substitute $g(t)$ in Eq. 2. The process of selecting a new global attractor $g_i \in G_i$ is randomized

Table 1: Experimental Setup.

Case	n (= $\|P\|$)	N	k	Env. size	Peak area size
1	30	3	3	$E = 5$	$E' = 4$
2	50	10	3	$E = 7$	$E' = 5.5$
3	50	15	5	$E = 10$	$E' = 8$

using the following probability function $p(g_i)$.

$$g_i \sim p(g_i) = \frac{\frac{r(g_i)}{dist(g_i, x_i)}}{\sum_{g_j \in G_i} \frac{g_j}{dist(g_j, x_i)}}, \tag{5}$$

where x_i is the location of particle p_i.

Selecting a new global attractor g_i for particle p_i would also randomize its next velocity $v(t+1)$ to remove the influence of the previous global attractor. To improve exploration of the environment, a re-diversification mechanism is implemented to cause the swarm to randomize the next positions $x(t + 1)$ and velocities $v(t + 1)$ of all particles if no particles were able to locate better positions in the last γ_2 ticks (Lines 30-34). However, this process does not randomize global peak positions G_i and local peak position y_i.

Positive integer parameters γ_1 and γ_2 in Line 23 and Line 30 denote the sensitivity of particles in detecting a convergence; lower values are preferred when peaks in an environment tend to be separated from each other. For the following experiments, values of $\gamma_1 = \gamma_2 = 5$ are used because they were deemed suitable.

The exploration continues until the termination condition of $t = T_f$, and then the candidate global peak set G_i of all particles are merged and duplicated values are removed; thus, the set of all candidate global peaks is $G = G_1 \cup \cdots \cup G_n$. Then, based on the utility value $r(g_i)$ of each element in the set G, elements are sorted in descending order. Finally, the top k values in the sorted set G are the top-k peaks in a given environment that are identified by the swarm (Line 37 in Alg. 1).

5 Experimental Evaluation and Discussion

5.1 Experimental Setup

Three experiments were conducted in both static and dynamic environments to evaluate the effectiveness of top-k PSO against several algorithms, including the NichePSO, canonical PSO, and Clerk-Kennedy PSO. We list the values of environmental parameters for each case (Cases 1, 2 and 3) in Table 1, where $(2E \geq) 2E' > 0$ is the length of the side of peak generation area, i.e., initial position of peak i is defined by $\boldsymbol{\mu}_i = (\mu_{1,i}, \mu_{2,i})$, and must satisfy $-E' < \boldsymbol{\mu}_i < E'$. In the initialization phase, all peaks are randomly generated inside the peak generation area of each experimental run. To insert a margin between the environment area and the peak generation area, E' was approximately 20% to 25% smaller than E. The value of area importance of each peak was set to $m_i = 1$ to represent all peaks having equal priority.

Table 2: Case 1 Experimental Result (in %)

Algorithm	1st Peak	2nd Peak	3rd Peak
top-*k*	64.5 ± 41.1	67.9 ± 42.2	71.1 ± 37.2
Canonical	29.4 ± 30.3	27.1 ± 19.0	17.9 ± 13.6
Clerk-Kennedy	31.3 ± 30.3	25.7 ± 17.0	18.3 ± 15.6
Niche	62.3 ± 32.6	13.4 ± 21.2	1.0 ± 2.4

Table 3: Case 2 Experimental Result (in %)

Algorithm	1st Peak	2nd Peak	3rd Peak
top-*k*	58.5 ± 27.7	67.3 ± 29.0	68.6 ± 32.5
Canonical	25.5 ± 24.9	27.4 ± 19.5	17.6 ± 12.3
Clerk-Kennedy	25.5 ± 25.9	26.3 ± 18.6	17.8 ± 15.5
Niche	65.5 ± 26.9	26.4 ± 25.5	2.5 ± 5.6

The threshold for closeness was the parameters $\delta_u \geq 0$ and $\delta_d \geq 0$, which are small positive numbers. For our experiments, we used the values $\delta_u = 0.05$, $\delta_d = 0.1$, $\delta_{eqv} = 10^{-4}$ and $T_f = 50,000$. For each experiment case, a total of 30 different environments were randomly generated, and 50 independent runs were executed. Experiment results are shown in tables with numbers denoting the means and their standard deviations of successfully identified peaks across the 30 randomly generated environments.

The experiments in dynamic environments used the same parameters as experiments in static environments, with only the randomized constant movement of peaks differing during the experiments, with a movement speed factor of $\omega = 50000$. A large constant was chosen to show notable but not drastic differences between static and dynamic environments, as a small movement speed factor such as $\omega = 100$ resulted in all algorithms failing to identify any peaks at all due to quick peak movements.

For the baseline method of canonical PSO, we set both local and global acceleration constants to $c_1 = 2$ and $c_2 = 2$. For top-*k* PSO and its base method Clerk-Kennedy PSO, we used two values from the original study [9], including a constriction rate of $\alpha = 0.729843788$ and an acceleration constant of $c = 2.05$. For NichePSO, a partitioning threshold of 10^{-3} and a merging distance for sub-swarms with swarm radius of $D = 0$ of 10^{-4} was adopted. As for the underlying GCPSO in NichePSO, the same constants as canonical PSO were used and the initial scaling factor was set to $\rho(t) = 0.1$. A detailed description of NichePSO and its base method are described in these references [6, 33, 34].

To show the performances of baseline methods, NichePSO, canonical PSO, and Clerk-Kennedy PSO in identifying top-*k* peaks, in each of those algorithms, an additional step of function Merging in Alg. 3 was added. Due to incompatibility when applying function Merging to the algorithms used for comparison, in canonical and Clerk-Kennedy PSO, Line 1 of Alg. 3 was replaced by $G = \bigcup_{i=1}^{n} y_i$, whereas in NichePSO $G = \bigcup_{i=1}^{n} g_i$.

Table 4: Case 3 Experimental Result (in %)

Algorithm	1st Peak	2nd Peak	3rd Peak	4th Peak	5th Peak
top-*k*	51.2 ± 32.2	64.8 ± 30.1	62.3 ± 35.4	65.7 ± 33.6	70.7 ± 32.8
Canonical	8.9 ± 10.3	15.4 ± 13.7	11.8 ± 10.4	12.7 ± 10.0	9.6 ± 8.8
Clerk-Kennedy	9.3 ± 9.8	16.3 ± 15.9	12.9 ± 11.6	12.5 ± 11.9	12.1 ± 8.9
Niche	43.7 ± 25.4	33.1 ± 25.2	11.9 ± 14.4	6.3 ± 13.1	2.9 ± 5.1

5.2 Experimental Results in Static Environment

Tables 2 to 4 show the results of the first experiment, i.e., the probabilities of identifying top-*k* peaks in Cases 1 to 3. These figures indicate that in identifying top-*k* peaks using these 4 algorithms, top-*k* PSO was the most successful with the exception of identifying the first peak compared to NichePSO in some cases.

In Case 1, the objective of the swarm was to identify all 3 peaks in the environment. Table 2 shows that top-*k* PSO was able to identify all of the 3 peaks with a probability greater than 0.5, while both the canonical and Clerk-Kennedy PSOs were only able to identify them with probability less than 0.4. In contrast, NichePSO was the second most successful at identifying the highest peak, but was also unable to identify the second and third peaks, with probabilities even lower than both canonical and Clerk-Kennedy PSOs. This issue is discussed further in Section 5.5.

Table 3 shows the experimental results of Case 2, which were similar to Case 1 but with $N = 10$ and $n = 50$, i.e., a swarm of fifty particles searching for the top 3 out of 10 peaks. This figure indicates that among the 4 algorithms used, top-*k* PSO was able to identify the top three peaks with the greatest probability. Particularly, top-*k* PSO was the only algorithm able to identify the third peak with an acceptable rate; the baseline methods exhibited lower probabilities in identifying peaks, especially the third peak, and NichePSO showed the lowest probability to identify the third one in the first experiment. However, NichePSO was the most successful in identifying the first peaks with the highest probability.

We conducted further experiment in a more complicated case with larger environments in Case 3, where 50 particles were used to identify the top 5 peaks from 15 peaks in the environment. The result of the experiment are shown in Table 4. This figure shows that similar to the previous two cases, top-*k* PSO was able to identify all peaks with high probabilities, and identified the all peaks at higher probabilities than baseline methods. However, unlike Case 2, top-*k* PSO outperformed NichePSO in identifying the first peak. However, repeating the experiment sometimes resulted in NichePSO performing better in identifying the first peak compared to top-*k* PSO by a small margin, vice versa. For more experimental results in static environments, please refer to our conference paper [29].

As for the standard deviation of results from all Cases, as shown in Tables 2 to 4, there was no notable difference between top-*k* PSO and the baseline methods, and each standard deviation of each experiment result for each *k*-th peak is relative to its mean. However, NichePSO did produce slightly smaller deviation compared to top-*k* PSO at identifying the first peak.

Table 5: Comparison of different peak area sizes ($E = 10$, in %)

E'	1st Peak	2nd Peak	3rd Peak	4th Peak	5th Peak
5	60.1 ± 31.5	82.6 ± 27.4	79.3 ± 27.5	73.3 ± 39.5	56.9 ± 43.0
6	71.3 ± 32.4	78.9 ± 22.8	82.4 ± 26.9	66.9 ± 40.0	65.0 ± 38.6
7	64.4 ± 28.7	63.0 ± 32.3	61.0 ± 33.1	78.9 ± 24.5	69.8 ± 34.2
8	50.7 ± 32.3	60.7 ± 34.9	57.5 ± 36.4	67.7 ± 33.1	56.4 ± 36.1
9	32.3 ± 34.7	40.8 ± 35.8	51.9 ± 41.8	54.3 ± 38.8	67.6 ± 39.6
10	41.1 ± 38.2	37.5 ± 39.8	45.0 ± 42.8	40.3 ± 37.3	49.6 ± 41.0

Table 6: Case 1 Dynamic Experimental Result (in %)

Algorithm	1st Peak	2nd Peak	3rd Peak
top-*k*	51.3 ± 38.1	51.5 ± 40.5	41.9 ± 34.3
Canonical	30.1 ± 23.5	18.6 ± 15.0	5.7 ± 5.2
Clerk-Kennedy	30.3 ± 25.1	17.8 ± 16.8	5.1 ± 7.4
Niche	49.9 ± 22.7	14.1 ± 14.9	1.4 ± 5.2

5.3 Effect of Peaks Near Edges of Environments

During the experiment, we discovered that peaks would sometimes be initialized near the edges of the environments, which negatively affected all algorithms on successfully identifying peaks. This was the basis for applying a margin of $E - E'$ between the edge of the environment and the edge of the peak generation area in our experiment. To examine this further, we performed six cases of the experiment with environmental sizes of 20 ($E = 10$) with varying margin sizes, i.e., $E' = 5, 6, 7, 8, 9, 10$. For other experimental settings, we used parameters from Case 3.

Table 5 shows the result of this comparison experiment.

From Table 5, it may be observed that when margin $E - E'$ of 20% (i.e., $E' \leq 8$) or greater, top-*k* PSO was able to find the top five peaks with approximately equal probability. However, the success rate noticeably decreased when the peak generation area size was $E' = 9$ and $E' = 10$. To improve the experimental result, we inserted a margin of about 20% in all Cases 1 to 3. Regardless, our proposed method top-*k* PSO produced better result with any margin, as shown in Table 5, than those by the baseline methods with $E' = 8$ (20% margin) in Table 4.

Table 7: Case 2 Dynamic Experimental Result (in %)

Algorithm	1st Peak	2nd Peak	3rd Peak
top-*k*	30.7 ± 25.5	32.1 ± 24.7	32.8 ± 29.8
Canonical	14.3 ± 14.2	11.6 ± 12.4	7.8 ± 8.4
Clerk-Kennedy	12.8 ± 9.2	10.3 ± 9.6	6.6 ± 7.7
Niche	34.9 ± 24.0	13.4 ± 14.0	3.7 ± 5.7

Table 8: Case 3 Dynamic Experimental Result (in %)

Algorithm	1st Peak	2nd Peak	3rd Peak	4th Peak	5th Peak
top-*k*	17.5 ± 16.1	26.8 ± 22.9	22.2 ± 14.3	32.9 ± 21.8	32.9 ± 24.3
Canonical	5.7 ± 8.3	5.7 ± 5.9	6.6 ± 5.7	4.1 ± 5.6	2.5 ± 3.4
Clerk-Kennedy	5.9 ± 8.0	7.6 ± 8.1	5.6 ± 7.5	4.7 ± 5.4	3.5 ± 5.6
Niche	14.3 ± 11.5	7.0 ± 7.9	2.7 ± 4.0	2.3 ± 5.4	0.7 ± 1.4

Table 9: Comparison of different movement speed factors (ω, in %)

ω	1st Peak	2nd Peak	3rd Peak	4th Peak	5th Peak
10000	1.4 ± 2.2	1.5 ± 2.0	1.5 ± 2.2	1.7 ± 3.1	1.5 ± 3.0
20000	7.3 ± 10.0	6.6 ± 7.8	3.4 ± 5.0	5.7 ± 6.1	6.7 ± 8.8
30000	9.3 ± 9.0	9.3 ± 9.2	13.3 ± 11.5	13.2 ± 14.9	10.7 ± 12.0
40000	14.5 ± 16.0	23.7 ± 21.7	25.5 ± 22.6	19.7 ± 17.9	15.3 ± 17.6
50000	17.5 ± 16.1	26.8 ± 22.9	22.2 ± 14.3	32.9 ± 21.8	32.9 ± 24.3
60000	22.2 ± 22.5	36.3 ± 27.1	34.0 ± 28.3	30.3 ± 27.6	34.5 ± 28.7
∞ (Static)	51.2 ± 32.2	64.8 ± 30.1	62.3 ± 35.4	65.7 ± 33.6	70.7 ± 32.8

5.4 Experimental Results in a Dynamic Environment

We conducted the second experiment in dynamic environments with the movement speed factor ω = 50, 000 to confirm whether our method can locate the moving top-*k* peaks. The results are shown in Tables 6, 7 and 8. In these experiments, we could observe the similar trends in Cases 1, 2, and 3 as of the static environments, that is, top-*k* PSO was the most successful in identifying all *k* peaks compared to other algorithms, although NichePSO showed better result in identifying only the first peak by a small margin in Case 2.

These results shown in Tables 6, 7 and 8 indicate that the probability of finding top-*k* peaks were slightly lower than those in the first experiment for all algorithms, although top-*k* PSO exhibited the acceptable probabilities of finding peaks. As for the standard deviation relative to its mean, we were not able to find noticeable difference with the experiments in static environment. These results suggested that the moving speed of peaks affect the performance of identifying top-*k* peaks.

Therefore, we investigated how moving speed affect the performance of top-*k* PSO by changing the movement speed factor to ω = 10000, 20000, 30000, 40000, 50000, 60000, and ∞. Note that $\omega = \infty$ corresponds to the static environment. This result is shown in Table 9. This graph indicates that the performance of top-*k*
PSO degraded if the movement was faster, i.e., the lower ω. We do not show the results, but we conducted the same experiments using the baseline methods and the results exhibited the same tendency, although their performance more quickly degrades than top-*k* PSO. This suggests that there is the trade of between moving speed and the frequency of calculation of particles, but frequent calculation requires more computational cost. This means that top-*k* PSO can reduce the required computational cost if the peaks are moving.

Slower peak movement minimizes impacts on performance because of one condition for positively identifying a peak, more precisely, the difference in Euclidean distance is less than 0.1. The higher the limit of peak movements, the higher the probability of the distance between the particle and its targeted peak being more than 0.1 in the last tick, which is reflected in the result in Table 9. This discussion also indicates the trade of

between moving speed and the frequency of calculation of particles. Note that even if ω = 50000 is fixed, the frequency of calculation can be increased by shortening the time length per unit for the simulated environment of moving particles.

5.5 Discussion

5.5.1 Static Environment

Our proposed method top-*k* PSO produced the best results in our experiments. The re-diversification introduced in our proposed method was the main component that supported this performance, because it makes particles in the swarm less susceptible to the local trap problem compared to the baseline methods.

Both the canonical and Clerk-Kennedy PSO algorithms exhibited a lower success rate in the simpler case of Case 1 with k = 3 and N = 3 (Table 2) was caused partially by gradually vanishing velocity update in both algorithms. In the canonical PSO, as a particle approached an optimum, the next velocity decreases, and eventually, the successive convergence would take infeasible amounts of time. A similar phenomenon was also observed in Clerk-Kennedy PSO; it is due to the constriction rate causing successive velocity updates to always be smaller than the previous velocity.

NichePSO could identify the first peak with similar probability compared top-*k* PSO, but due to the long simulation time, the sub-swarms of NichePSO exhibited a high probability of coalescing, causing the swarm to lose multiple tracked op- tima [10], and making the behavior of the swarm closer to that of the canonical PSO. In multi-modal problems such as in our case, this phenomenon is problematic, as shown in Tables 2 to 4, where NichePSO performed worse than both baseline methods when identifying second and subsequent peaks.

Comparing the result of Case 1 (Table 2) with that of Case 2 (Table 3), there was a significant decrease in the probability of identifying the top k peaks in the environment. This was most likely caused by the local trap problem, as number of available peaks N were much larger in Case 2 than in Case 1. Although Case 2 employed a larger number of particles n, the algorithms failed to overcome the local trap problem. An even larger number of particles may be required to obtain a similar result.

Then, comparing the result of Case 2 (Table 3) with that of Case 3 (Table 4), top-*k* PSO identified the top k peaks with a success rate similar to that of Case 2, despite more difficult parameters; Case 3 had a greater target peak count k and peak count N while using the same particle count n. One possible reason for this result is that information propagation is limited to particles within Euclidean distance of $2 \cdot E/k^2$; higher k results in more localized communications. Shorter communication distances causes the behavior of particles to be closer to that of a cognition-only model, which is suited for exploring local areas and thus identifying individual peaks. Due to the completely randomized generation of peaks in the environment, a peak could be generated very close to another peak with a small difference between their utility values. This phenomenon results in a tendency of particles to move toward peaks with higher utility values, reducing success rates for later peaks. This was observed in Case 1 and Case 2, in which probability rates fell the lower the position of the targeted peak is across all algorithms. However, this was not noticeable in Case 3 (Table 4) due to the larger k, because particles would be less affected by peaks identified by other particles.

Even at this current state, top-*k* PSO still requires many improvements for its convergence performance, including its re-diversification mechanism. For example, NichePSO identified the first peak better than top-*k* PSO across Case 1 to Case 3, which is mainly attributed to its base method of GCPSO [34]. In GCPSO, only the best particle (i.e., $\text{argmax}_{p_i \in P} r(\mathbf{x}(p_i))$) continues exploration of the environment while other particles update their positions using canonical PSO, reducing the susceptibility of its particles to the local trap problem. NichePSO has a tendency to coalesce [10] causing the swarm to behave more like GCPSO, which limits its ability to find multiple peaks. In contrast, although a re-diversification strategy is included in top-*k* PSO at the swarm level, a single converging particle suffices to pre- vent its activation. Considering the slowing velocity update rate of the underlying Clerk-Kennedy PSO, the probability of re-diversification mechanism being activated fell as time progressed. To improve both convergence accuracy and speed of top- *k* PSO, additional technologies should be implemented such as deep reinforcement learning [36], which remains as a possible topic for future research.

5.5.2 Dynamic Environment

In the dynamic environment of the second experiment, peaks moved randomly as time progressed, and thus, the utility value $r(x)$ at coordinate x at time t may differ at time $t + 1$, which would cause a problem of stale values stored in particles. All the algorithms used in the experiment, including our proposed top-*k* PSO, are unable to track time elapsed since utility values were calculated and thus, they could not perform at acceptable levels in dynamic environments. Moreover, as time progresses, swarms would eventually lose diversity, which is another serious issue when exploring dynamic environments [5].

Two main concerns in exploring dynamic environments were evident from the experimental results of Case 1 to Case 3, as shown in Tables 6 to 8. Compared to their counterparts in the static version, all algorithms in Case 1 to Case 3 produced lower successful identification rate of top *k* peaks. This was mainly due to the loss of diversity, because the swarms tended to converge over time, which rendered them unable to continuously explore the environment and thus caused them to lose track of the peak points. However, the relative trend in the results achieved by top-*k* PSO compared with those of other algorithms remained; top-*k* PSO was better at identifying second and later peaks than other algorithms, while similar of slightly worse than NichePSO at identifying the first peak.

Furthermore, the reduction in probability of identifying peaks between the dynamic and its static counterpart of Case 3 was larger than Case 1. This may be attributed to limit of peak movements calculated using the side length of the environment E; the greater the range of the movement, the more likely that a tracked peak may move outside the Euclidean distance limit of δ_d from the particle, causing the particle to be unable to identify the peak successfully. However, the opposite may also occur, albeit at lower probability, when a peak further than δ_d from a particle moves to within δ_d, enabling the particle to successfully identify the peak. To confirm this observation, we also experimented using Case 3 in dynamic environment but with different movement speed factors ω for the peaks, as shown in Table 9. The results of this experiment showed that the slower the movement of the peaks in the environment (higher ω), the higher the probabilities of identifying the top k peaks in the environment. Even though top-*k* PSO includes a re-diversification mechanism which should reduce the decrease in

performance when tracking faster moving peaks, the mechanism is triggered only when the particle remains converged after γ_1. Considering that peaks move constantly, this mechanism may be triggered far less often in faster environments. Further research would be required to refine this condition to allow top-*k* PSO to identify optimal points better in dynamic environments, especially with fast-moving peaks.

6 Conclusion

In this study, we modeled the problem of identifying areas with higher probability of survivors after a disaster as a two-dimensional environment, using a bivariate normal distribution to simulate areas with different concentrations of survivors. To solve this problem, we extended a PSO algorithm to develop our proposed top- *k* PSO, which is designed to identify top-*k* locations among all possible locations where survivors would be more likely to be found. After multiple experiments with different variables in a static environment and a comparison with several other algorithms, we found that top-*k* PSO was more effective in identifying a single peak (optimum) in most cases and in almost all cases when identifying top-*k* peaks. The re-diversification method that we introduced improved the convergence performance and also maintained swarm variation throughout the simulation.

When top-*k* PSO was tested in a dynamic environment, it was still able to identify the top-*k* locations at a satisfactory rate. Notably, our proposed algorithm was less affected in dynamic environments compared to the other algorithms compared. However, the performance of top-*k* PSO degrades quickly the faster the targets in the environment move.

From these experiments, we were able to identify several weak points that can be developed further; an example would be to enable top-*k* PSO to explore an environment with fewer particles than the number of peaks in the environment, mimicking the real-life situation of a limited number of aerial drones. The other weak point is that although we performed simulations in dynamic environments, a study on the aftermath of the 2010 Haiti earthquake showed that the movement of survivors is highly predictable [21], which suggests that our simplified model of randomized movements might not be fully applicable in the real world. Hence, a more accurate model of the problem is required to fully evaluate the effectiveness of our proposed algorithm in real applications. Furthermore, the performance of top-*k* PSO in dynamic environment is highly influenced by the movement speed of the dynamic peaks, and further research is required to enable top-*k* PSO to identify and track faster-moving peaks and peaks with changing utility values.

In addition to the need for additional investigation to realize application of top-*k* PSO in real world, collisions between *unmanned aerial vehicles* (UAVs) must also be avoided if top-*k* PSO were applied to actually search for potential survivors in a disaster area. Many different approaches can be applied to prevent collisions, such as using PSO [23] or using deep reinforcing learning [30]; combining top-*k* PSO with these methods could provide another potential direction for future research.

Acknowledgment

This work was partly supported by JSPS KAKENHI Grant Numbers 20H04245.

References

[1] Ollero A. Control and perception techniques for aerial robotics. In *7th IFAC Symposium on Robot Control*, pages 717–725, 2003.

[2] Ekkarat Adsawinnawanawa and Boontee Kruatrachue. Mutation variations in improving local optima problem of pso. 2020.

[3] Linus Bengtsson, Xin Lu, Anna Ekeus Thorson, Richard Garfield, and Johan von Schreeb. Improved response to disasters and outbreaks by tracking population movements with mobile phone network data: A post-earthquake geospatial study in haiti. *PLoS Medicine*, 8, 2011.

[4] Tim M. Blackwell. Particle swarm optimization in dynamic environments. In *Evolutionary Computation in Dynamic and Uncertain Environments*, 2007.

[5] Tim M. Blackwell and Peter John Bentley. Dynamic search with charged swarms. In *GECCO*, 2002.

[6] R. Brits, A. Engelbrecht, and F. van den Bergh. A niching particle swarm optimizer. In *Proceedings of the Fourth Asian-Pacific Conference on Simulated Evolution and Learning*, pages 692–696, 2002.

[7] R. Brits, A.P. Engelbrecht, and F. van den Bergh. Solving systems of unconstrained equations using particle swarm optimization. In *IEEE Conference on Systems, Man, and Cybernetics*, Tunisia, 2002.

[8] Shuang Cao, Yulong Wang, Xiaoxiang Wang, and Qi Li. A rapid assessment method for seismic intensity area and affecting field direction using mobile phone base stations. *2020 5th International Conference on Computer and Comunication Systems (ICCCS)*, pages 623–627, 2020.

[9] M. Clerk and J. Kennedy. The particle swarm: explosion, stability, and con- vergence in a multi-dimensional space. *IEEE transactions on Evolutionary Computation*, 6, 2000.

[10] Tyler Crane, Beatrice M. Ombuki-Berman, and Andries Petrus Engelbrecht. Nichepso and the merging subswarm problem. *2020 7th International Conference on Soft Computing & Machine Intelligence (ISCMI)*, pages 17–22, 2020.

[11] Simon Dennis and Andries Petrus Engelbrecht. Dynamic multi-swarm fractional-best particle swarm optimization for dynamic multi-modal optimization. *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 1549–1556, 2020.

[12] Andries P. Engelbrecht. Heterogeneous particle swarm optimization. *ANTS 2010, LNCS 6234*, pages 191–202, 2010.

[13] Issam Fares, Rizk Masoud Rizk-Allah, Aboul Ella Hassanien, and Snasel Vaclav. Multiple cyclic swarming optimization for uni- and multi-modal functions. 2020.

[14] J. Kennedy. The particle swarm: Social adaptation of knowledge. In *Proceedings of the IEEE International Conference on Evolutionary Computation*, pages 303–308, 1997.

[15] J. Kennedy. Bare bones particle swarms. In *Proceedings of the IEEE Swarm Intelligence Symposium*, pages 80–87, 2003.

[16] J. Kennedy and R.C. Eberhart. A new optimizer using particle swarm theory. In *Proceedings of the Sixth International Symposium on Micromachine and Human Science*, pages 39–43, 1995.

[17] J. Kennedy and R.C. Eberhart. Particle swarm optimization. In Proceedings of IEEE *International Conference on Neutral Networks*, volume ICNN '95, pages 1942–1948, Perth, Western Australia, December 1995.

[18] Barend J. Leonard, Andries Petrus Engelbrecht, and Andrich B. van Wyk. Heterogeneous particle swarms in dynamic environments. *2011 IEEE Symposium on Swarm Intelligence*, pages 1–8, 2011.

[19] Xiaodong Li. Niching without niching parameters: Particle swarm optimization using a ring topology. *IEEE Transactions on Evolutionary Computation*, 14:150–169, 2010.

[20] Andrei Lihu and Stefan Holban. A study on the minimal number of particles for a simplified particle swarm optimization algorithm. In *2011 6th IEEE International Symposium on Applied Computational Intelligence and Informatics (SACI)*, pages 299–303, 2011.

[21] Xin Lu, Linus Bengtsson, and Petter Holme. Predictability of population displacement after the 2010 haiti earthquake. *Proceedings of the National Academy of Sciences*, 109:11576 – 11581, 2012.

[22] M. Mitchell. *An Introduction to Genetic Algorithms*. MIT Press, 1996.

[23] Eslam Nabil Mobarez, Amr A. Sarhan, and Mohamed Ashry. Obstacle avoidance for multiuav path planning based on particle swarm optimization. *IOP Conference Series: Materials Science and Engineering*, 1172, 2021.

[24] Venkataraman Muthiah-Nakarajan and Mathew Mithra Noel. Galactic swarm optimization: A new global optimization metaheuristic inspired by galactic motion. *Appl. Soft Comput.*, 38:771–787, 2016.

[25] Masahiko Onosato, Satoshi Tadokoro, Hiroaki Nakanishi, Kenzo Nonami, Kuniaki Kawabata, Yasushi Hada, Hajime Asama, Fumiaki Takemura, Kiyoshi Maeda, Kenjiro Miura, and Atsushi Yamashita. *Rescue Robotics*, chapter 3. Springer, 2009.

[26] K. E. Parsopoulos and M. N. Vrahatis. Recent approaches to global optimization problems through particle swarm optimization. *Natural Computing*, pages 235–306, 2002.

[27] Alessandro Passaro and Antonina Starita. Particle swarm optimization for multimodal functions: a clustering approach. *Journal of Artificial Evolution and Applications*, 2008:8, 2008.

[28] Lukkana Poempool, Boontee Kruatrachue, and Kritawan Siriboon. Combine multi particle swarm in supporting trapping in local optima. *2018 International Conference on Engineering, Applied Sciences, and Technology (ICEAST)*, pages 1–4, 2018.

[29] Stephen Raharja and Toshiharu Sugawara. Identifying top-k peaks using an extended particle swarm optimization algorithm with re-diversification mechanism. *SCAI 2022*, pages 359–366, 2022.

[30] Mahsoo Salimi and Philippe Pasquier. Deep reinforcement learning for flocking control of uavs in complex environments. *2021 6th International Conference on Robotics and Automation Engineering (ICRAE)*, pages 344–352, 2021.

[31] Y. Shi and R.C. Eberhart. An empirical study of particle swarm optimization. In *Proceedings of the 1999 Conference on Evolutionary Computation*, pages 1945–1950, IEEE Service Center, Piscataway, NJ, 1999.

[32] Satoshi Tadokoro. *Rescue Robotics*, chapter 1. Springer, 2009.

[33] F. van den Bergh. *An Analysis of Particle Swarm Optimizers.* PhD thesis, Department of Computer Science, University of Pretoria, Pretoria, South Africa, 2002.

[34] F. van den Bergh and A. P. Engelbrecht. A new locally convergent particle swarm optimizer. In *IEEE Conference on Systems, Man, and Cybernetics*, Tunisia, 2002.

[35] Jianfang Wang, Ning Cheng, Zhidu Liu, and Changwang Liu. Applying fusion of pso-abc algorithm on the minimax location problem. *The Open Cybernetics & Systemics Journal*, 8, 2014.

[36] Pin Zhang, Haorong Li, Quang Phuc Ha, Zhen-Yu Yin, and Ren peng Chen. Reinforcement learning based optimizer for improvement of predicting tunneling- induced ground responses. *Adv. Eng. Informatics*, 45:101097, 2020.

[37] Zhe Zhang, Limin Jia, and Yong Qin. Modified constriction particle swarm optimization algorithm. *Journal of Systems Engineering and Electronics*, 26:1107– 1113, 2015.