Hindawi Publishing Corporation
Mathematical Problems in Engineering
Volume 2014, Article ID 187370, 10 pages
http://dx.doi.org/10.1155/2014/187370

Research Article

PSO-Based Robot Path Planning for Multisurvivor Rescue in Limited Survival Time

N. Geng, D. W. Gong, and Y. Zhang

School of Information and Electrical Engineering, China University of Mining and Technology, Xuzhou 221116, China

Correspondence should be addressed to D. W. Gong; dwgong@vip.163.com

Received 5 June 2014; Revised 5 August 2014; Accepted 19 August 2014; Published 25 September 2014

Academic Editor: Fang Zong

Copyright © 2014 N. Geng et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

Since the strength of a trapped person often declines with time in urgent and dangerous circumstances, adopting a robot to rescue as many survivors as possible in limited time is of considerable significance. However, as one key issue in robot navigation, how to plan an optimal rescue path of a robot has not yet been fully solved. This paper studies robot path planning for multisurvivor rescue in limited survival time using a representative heuristic, particle swarm optimization (PSO). First, the robot path planning problem including multiple survivors is formulated as a discrete optimization one with high constraint, where the number of rescued persons is taken as the unique objective function, and the strength of a trapped person is used to constrain the feasibility of a path. Then, a new integer PSO algorithm is presented to solve the mathematical model, and several new operations, such as the update of a particle, the insertion and inversion operators, and the rapidly local search method, are incorporated into the proposed algorithm to improve its effectiveness. Finally, the simulation results demonstrate the capacity of our method in generating optimal paths with high quality.

1. Introduction

Due to the influence of natural hazards and terrorism, various disasters frequently occur in our world. Since robots are very useful in dangerous or abominable environments where human could not reach certain targets, robot rescue in disaster remains an interesting and challenging subject for researchers [1, 2]. However, as one key issue in robot navigation, how to plan an optimal rescue path of a robot, along which the robot could rescue as many survivors as possible in limited time, has not yet been fully solved.

Up to now, much work has been done on robot path planning using various approaches, such as the cell decomposition [3], artificial potential field [4], visibility graph [5], probabilistic roadmap [6], quick random search tree [7], and intelligent search algorithm [8–11], to say a few. However, most of these approaches overlook the fact that survivors have only limited survival time because of atrocious circumstance. Actually, after a disaster, a survivor's strength will sharply decline, so she/he must be rescued within the short period of her/his survival time. In this case, it is of essence to plan a reasonable path so as to rescue as many survivors as possible in limited time.

This paper studies the problem of robot rescue path planning with time-constraint. Since a robot needs to rescue as many survivors as possible in limited time, the survival time of a survivor is introduced to reflect the feasibility of a path, and the number of rescued survivors is used to evaluate the quality of the path. Based on this, the above problem is formulated as a constraint optimization problem. To solve this problem, a modified PSO algorithm is presented. PSO was originally proposed by Kennedy and Eberhard [12]. Since it has a simple structure, fast convergence, and few parameters to be set, it has been widely applied in the field of robot path planning [13–15]. However, there have been no studies on robot rescue path planning with time-constraint.

The main contributions of this paper are as follows. (1) The mathematical model of the problem of robot rescue path planning is formulated, considering that a survivor's survival time is limited; (2) a modified integer PSO is proposed to solve the above model, and several new operators, such as the update of a particle, the insertion and inversion operators, and the rapidly local search method, are incorporated to enhance the capability of the proposed PSO; (3) a series of simulations are done to verify the effectiveness of the proposed method.

The remainder of this paper is organized as follows. Section 2 is a review on related work. The problem of rescue path planning is formulated in Section 3. The modified PSO algorithm used to solve the established model which is elaborated in Section 4. Section 5 is the simulation results. Section 6 draws the conclusion and provides some open questions.

2. Related Work

2.1. Rescue Path Planning of a Robot. There has been much meaningful work on the problem of rescue path planning. In order to search and rescue survivors in a building after a disaster, Kibler et al. constructed a flooding wireless protocol and a flooding fulfill search method and used an autonomous mobile robot as the development platform to verify the effectiveness of the wireless protocol and the search method [16]. Mandal et al. utilized a generalized local Voronoi graph to establish the models of a disaster site, coverage range, and team work and planned a robot path online based on information provided by sensors during rescue [17]. Ferdous et al. proposed a method of transferring people within the shortest time and the minimum cost, which focuses on disasters caused by flood, tsunami, and typhoon. The proposed method can also be used in other time-critical man-made disaster rescue operations [18]. In order to overcome the deficiency that a robot needs human help during the outdoor search and rescue, Doroodgar et al. presented a method of searching outdoors based on the semiautonomous human-computer interaction, which used information from sensors to guide a robot to perform rescue operations [19]. Aiming at the rescue problem in mines, Tian et al. developed a modified neural network to expand the Kalman filter and realized a method of robot path planning based on multisensor fusion [20]. Zhang et al. employed a modified ant colony algorithm (ACO) to the problem of rescue path planning [21]. Basilico and Amigoni introduced a strategy based on a multicriterion decision to solve the problem of robot path planning [22]. In addition, Chien et al. used a robot team to search, rescue, patrol, and do other military tasks, so as to seek for the divergence between human manual control and autonomous path planning [23].

These studies enrich the method of robot rescue path planning. However, the above work omits the fact that a survivor has limited time to wait for rescue, and there have been very few studies on formulating the mathematical model of the rescue problem. Moreover, various intelligent algorithms have been applied to robot path planning but rarely applied to the rescue problem.

2.2. PSO. PSO is a population-based stochastic global optimization technique, originated from the simulation of bird flock and fish school seeking food [12]. In PSO, a swarm of individuals (called particles) fly through the search space. Each particle represents a candidate solution, and particles move in the search space to search for the optimal solution(s) by updating the position of each particle based on the experience of its own and its neighboring particles. The best previous position of a particle is recorded as the personal best, called *pbest*, and the best position obtained by the swarm so far is the global best, called *gbest*. PSO searches for the optimal solution(s) by updating the position and velocity of each particle according to the following equations:

$$\begin{aligned} V_{ij}(T+1) &= \omega V_{ij}(T) + c_1 r_1 \left(p_{ij}(T) - X_{ij}(T)\right) \\ &\quad + c_2 r_2 \left(p_{gj}(T) - X_{ij}(T)\right), \\ X_{ij}(T+1) &= X_{ij}(T) + V_{ij}(T+1), \end{aligned} \tag{1}$$

where T denotes the number of generations during the evolution, ω is the inertia weight, c_1 and c_2 represent the acceleration constants, r_1 and r_2 are random values uniformly distributed in $[0,1]$, and p_{ij} and p_{gj} refer to the elements of *pbest* and *gbest* in the jth dimension.

2.3. Robot Path Planning Based on PSO. Masehian and Sedighizadeh presented two novel PSO-based algorithms for robot path planning and evaluated a path based on two criteria. The experiment results empirically verify the effectiveness of the proposed method. However, this method only works for continuous optimization problem [24]. Aiming at the simulated robot path planning, Adham and Somnuk proposed a modified PSO by extending the search space of a particle [25]. In order to obtain an optimal and smooth path, Lee et al. introduced an online planning method to completely cover a path based on PSO [26]. Considering danger sources in an environment, Zhang et al. took the danger degree and the path length as two objectives, formulated a mathematical model of the path planning problem, and presented an interval multiobjective PSO to seek for solutions of the problem [27]. Since there exist errors during the robot movement, Liu et al. proposed a path planning method of multirobot, avoiding collisions and satisfying the constraints on velocity and acceleration [28]. Tang and Eberhard presented a modified PSO by improving the method of updating a particle based on an enhanced Lagrangian operator and applied it to swarm-robot path planning [29]. In addition, Xue and Liu demonstrated a method of robot path planning based on PSO, which utilizes the danger degree grid to provide information related to an environment, and designed a fitness function based on the weighted sum of the danger degree and the path length [30].

These studies enrich the method of robot path planning. However, all these approaches do not consider the strength of a trapped person. So, they are not suitable for solving the problem of robot rescue path planning with constraints.

3. Formulation of the Problem of Robot Rescue Path Planning

As one key issue in robot rescue, the problem of rescue path planning with time constraint is investigated and formulated in this section.

3.1. Problem Descriptions. In order to describe the problem of rescue path planning, the following assumptions are made.

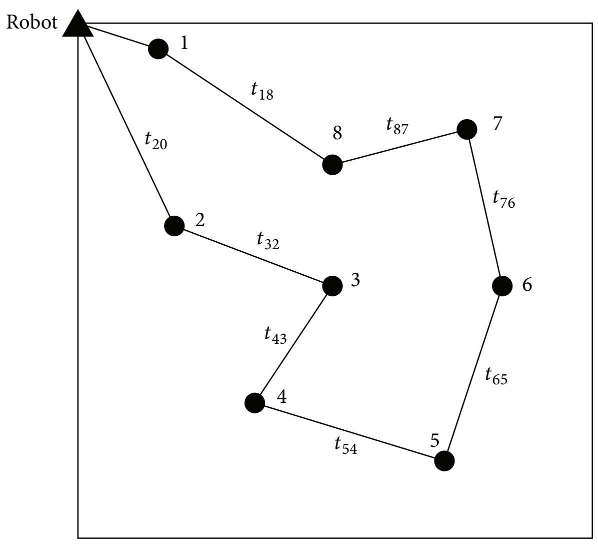

Figure 1: Robot rescue environment.

(1) There is only one robot and its power is enough to complete the whole rescue task.

(2) The position of a trapped person (target) is known in advance and fixed during rescue.

Given that the two-dimensional workplace contains many targets, the problem of robot rescue path planning can be stated as conducting the rescue task according to the planned path in order to rescue the most targets in limited time. If the strength of a trapped person is smaller than or equal to a threshold set in advance, it is of no necessity to rescue her/him.

Assume that there are N trapped persons, denoted as $1, 2, \ldots, N$, and the robot follows a certain sequence, represented as $s_1, s_2, \ldots, s_N$, to move along, where $s_i \in Z$, $1 \leq s_i \leq N$. If the initial position of the robot is regarded as a target, denoted as s_0, and is set to 0, a path, denoted as $s_0, s_1, s_2, \ldots, s_N$, can be obtained according to the above rescue sequence. The time sequence of the robot passing through two adjacent points is denoted as $t_{s_0 s_1}, t_{s_1 s_2}, \ldots, t_{s_{N-1} s_N}$, respectively, where $t_{s_i s_{i+1}}$ means the moving time from target s_i to s_{i-1}. Figure 1 depicts the positions of eight targets and a robot, where a target and a robot are represented by a solid circle and a solid triangle, respectively. If the robot follows the path given in Figure 1, the sequence is represented as 1, 8, 7, 6, 5, 4, 3, 2, and the corresponding robot rescue path is 0 1 8 7 6 5 4 3 2.

3.2. Mathematical Model. It is possible that the robot cannot rescue all targets when it follows the sequence given in Figure 1, since a target may die before the robot reaches it, as a result of the strength of this person smaller than the threshold. So, a reasonable path planning strategy is required to rescue the most alive targets. To this end, the number of rescued targets is utilized to evaluate a path, and the limited strength of a target should be considered to evaluate the feasibility of the path.

The rescue sequence is denoted as $s_1, s_2, \ldots, s_N$, which forms a rescue path, S. In order to formulate the objective of the problem, the limited strength of a target is first calculated when the robot reaches her/him; second, the strength is compared with the threshold, and if the former is larger than the latter, the number of the rescued targets increases one; otherwise, this number remains unchanged. If the number of the rescued targets is denoted as $F(S)$, it can be represented as follows:

$$F(S) = \sum_{i=1}^{N} f(s_i), \quad f(s_i) = \begin{cases} 1, & \sigma_{s_i} - \Delta\sigma > 0, \\ 0, & \text{otherwise}, \end{cases} \tag{2}$$

where $f(s_i)$ represents a sign that reflects whether target s_i is rescued or not, $\Delta\sigma$ is the threshold set in advance, and σ_{s_i} means the strength of target s_i when the robot reached s_i.

Generally, the value of σ_{s_i} at the current time can be estimated by the time spent in the robot moving and that spent in rescuing a target. Based on the method in [31], the value of σ_{s_i} can be estimated by the following equations:

$$\sigma_{s_i} = \sigma_{s_i}^{0} e^{-0.037 t_{s_i}}, \tag{3}$$

$$\sigma_{s_i}^{0} = \sigma_o \cdot \min\left\{\frac{d_{s_i}}{\mathrm{DS}}, 1\right\}, \tag{4}$$

$$t_{s_i} = \frac{P_{s_i}}{v} + \mathrm{ST}, \tag{5}$$

where t_{s_i} is the time spent by the robot in reaching target s_i from the start position. As formula (5) shows, it includes the following two parts: one is the time spent by the robot in moving from the start position to target s_i, equivalent to the path length from target s_i to the start position, P_{s_i}, divided by the velocity of the robot, v; the other is the time spent by the robot in rescuing a target, equal to ST.

In formula (4), σ_o is the average strength of a healthy person, $d_{s_i} = \sqrt{(x_{s_i} - x_{\mathrm{DS}})^2 + (y_{s_i} - y_{\mathrm{DS}})^2}$ means the distance between a trapped person and the source of the disaster, DS is the maximum radius of the region influenced by the disaster, and $\sigma_{s_i}^{0}$ refers to the initial strength (after the disaster) of target s_i. From formula (4), the smaller the distance between a trapped person and the source of the disaster is, the higher the injury degree of this person should be. If the trapped person is far from the source of the disaster, the injury will be slight, the strength will be larger, and the survival time will be longer.

Overall, the problem of rescue path planning can be formulated as follows:

$$\begin{aligned} &\max \ F(S) \\ &\text{s.t.} \quad \sigma_{s_i} > \Delta\sigma, \quad i = 1, \ldots, N. \end{aligned} \tag{6}$$

4. The Proposed PSO Algorithm

In order to solve the problem formulated as formula (6), an effective method is needed.

Since the problem is NP hard, various methods can be employed to solve it, especially genetic algorithm (GA) and PSO. Both PSO and GA are random search algorithms and suitable for solving the above problem. Though they both have a similar evolutionary process, PSO has memory but crossover and mutation operators which are owned by GA.

Compared with GA, PSO has a different mechanism in sharing information. In GA, information sharing is done among chromosomes. So the whole population moves toward the optimal areas uniformly. While in PSO, only *gbest* and *pbest* provide information to other particles, which is a one-way flow of information. Therefore, the search process follows the current optimal solution(s). As a consequence, compared with GA, all particles may converge to the optimal solution(s) more rapidly.

Besides, PSO has many advantages, such as simple structure, fast convergence, and few parameters to be set. So PSO is employed to solve the above problem in this study, with the purpose of expanding the application range of PSO.

There exist various well-established discrete PSO algorithms [32, 33]. However, most of them use the binary encoding method, which cannot reflect the rescue sequence, resulting in unsuitable for solving the above problem. A modified PSO algorithm is thus proposed in this study, and several operators are employed to improve the performance of the proposed algorithm.

Since the above problem is discrete, a new integer coding method is first introduced to encode a solution; then, a method of updating a particle that is suitable for the integer coding and an approach to updating *gbest* based on the insertion and inversion operators are presented, followed that a rapidly local search method is incorporated to search for a better solution in the neighbor of a particle; finally, an exchange operator is used to improve the diversity of the swarm.

4.1. Particle Coding. For the decision variable, S, in formula (2), its corresponding rescue sequence is $s_1, s_2, \ldots, s_N$, which is encoded as a number of integers, denoted as X, and different decision variables correspond to different coding strings. When PSO is employed to solve formula (2), different particles correspond to different coding strings. The length of a particle's coding string is the same as the number of targets, denoted as N. Assume that a rescue path is 1 2 4 3, and then the path indicates that the robot starts from target 1, goes through targets 2 and 4 sequentially, and at last reaches target 3. So, the number of targets to be rescued is four, and the corresponding particle is encoded as $x = (1, 2, 4, 3)$.

4.2. Particle Updating. Considering the swarm with n particles, its ith particle's position and velocity are $X_i(T) = (X_{i1}(T), X_{i2}(T), \ldots, X_{iN}(T))^T$ and $V_i(T) = (V_{i1}(T), V_{i2}(T), \ldots, V_{iN}(T))^T$, respectively. *pbest* and *gbest* are $p_i(T) = (p_{i1}(T), p_{i2}(T), \ldots, p_{iN}(T))$ and $p_g(T) = (p_{g_1}(T), p_{g2}(T), \ldots, p_{g_n}(T))$, respectively. As each element of a particle is an integer, $X_{ij}(T)$ is updated by the following formulas:

$$\begin{aligned} V_{ij}(T+1) = \big\lfloor \omega V_{ij}(T) + c_1 r_1 \left(p_{ij}(T) - X_{ij}(T)\right) \\ + c_2 r_2 \left(p_{g_i}(T) - X_{ij}(T)\right) \big\rfloor, \end{aligned} \quad (7)$$

$$X_{ij}(T+1) = \operatorname{Mod}\left(X_{ij}(T) + V_{ij}(T+1), N\right) + 1, \quad (8)$$

where $i = 1, 2, \ldots, n$, $j = 1, 2, \ldots, N$, $\lfloor\cdot\rfloor$ represents the rounding down function, and Mod $(\cdot,\cdot)$ means the mod function.

Traditional methods of updating a particle focus mainly on a continuous numerical optimization problem. Different from the traditional methods, formula (7) performs the rounding down function on the velocity of a particle, with the purpose of the velocity being an integer after updating. Furthermore, in formula (8), a mod operation is done to $(x_{ij}(T) + v_{ij}(T+1))$, leading to the gained number being an integer in the range of $[1, N]$. As a result, the updated particle can meet the coding requirement.

Although formulas (7) and (8) ensure that each element of the newly generated particle locates in the range of $[1, N]$, there exists the case that a part of elements reappear in a particle. Assume that a particle is $X_i(T) = (1, 2, 4, 3)$, and its newly generated position is $X_i(T+1) = (1, 3, 3, 4)$ after updating. It is obvious that this particle is infeasible.

The particle after update is denoted as $X_i(T+1) = (X_{i1}(T+1), X_{i2}(T+1), \ldots, X_{iN}(T+1))$. The following gives a method that makes an infeasible particle, $X_i(T+1)$, feasible by deleting those repeated elements, we call it the correction operator.

Assume that q elements of a particle are the same; in order to obtain a feasible rescue sequence, the same $(q-1)$ elements in particle $X_i(T+1)$ should be deleted, and integers within $[1, N]$ that have not been selected should be added to $X_i(T+1)$ in turn.

Taking the particle, $X_i(T+1) = (1, 3, 3, 4)$, as an example, it has two elements. So the particle should be modified. To this end, the last element with the same value in $X_i(T+1)$ is first deleted and $X_i(T+1) = (1, 3, 4)$. Then, an integer within $[1, 4]$ that has not been selected, that is, $D = (2)$, is selected and added to $X_i(T+1)$. Finally, $X_i'(T+1) = (1, 3, 4, 2)$.

4.3. Update of pbest. For the problem of robot rescue path planning, the number of optimal solutions may be more than one. Some optimal solutions may have the same number of rescued targets but with different rescue sequences. So they correspond to different rescue paths. In order to retain these optimal solutions, the strategy of updating *pbest*, given as follows, is different from the traditional one:

$$pbest\,(T+1) = \begin{cases} pbest\,(T), & F\left(X_i(T+1)\right) < F\left(pbest\,(T)\right) \\ \operatorname{rand}\left(pbest\,(T), X_i(T+1)\right), & F\left(X_i(T+1)\right) = F\left(pbest\,(T)\right) \\ X_i(T+1), & F\left(X_i(T+1)\right) > F\left(pbest\,(T)\right). \end{cases} \quad (9)$$

If the newly generated solution is better than *pbest*, *pbest* is updated to the newly generated one; if the newly generated solution is worse than *pbest*, *pbest* remains fixed; otherwise, *pbest* selects one from them at random.

4.4. Update of gbest. In traditional PSO, *gbest* plays a very important role in guiding the evolution of particles. The existing method selects the best one as *gbest* of particles among all *pbests* and the current *gbest*. However, given the fact that all particles select the same *gbest*, the swarm is prone to trap into local optima. In order to improve the capability of our algorithm in exploration, a new method of updating *gbest* is presented based on the insertion and inversion operators.

For the problem investigated in this paper, there may exist the case that different particles have the same fitness. On this circumstance, all particles with the best fitness are first kept in a set, called the optional set. Then, two operators, that is, the insertion and inversion operators proposed in [34], are performed to generate *cbest* based on the optional set. If *cbest* is better than *gbest*, *cbest* is selected as the new *gbest*; otherwise, *gbest* remains unchanged. The following is the detailed operations about the insertion and inversion operators.

Insertion Operator. For a solution, $p = \{p_1, p_2, \dots, p_i, \dots, p_j, \dots, p_N\}$, first, randomly select an element and a position, denoted as p_i and j. Then, insert p_i into the position after j. As a result, a new solution, denoted as $p' = (p_1, p_2, \dots, p_{i-1}, \dots, p_j, p_i, \dots, p_n)$, can be obtained.

Inversion Operator. For a solution, $p = (p_1, p_2, \dots, p_i, \dots, p_j, \dots, p_n)$, first, randomly select two inverse positions. Then, inverse all the elements between them. Based on this, a new solutionis generated, denoted as $p'' = (p_1, p_2, \dots, p_j, p_{j-1}, \dots, p_{i+1}, p_i, \dots, p_n)$, where i and j are the inverse positions and $i < j$.

Compared with the traditional method, the above strategy utilizes different *gbests* when updating different particles, which is a benefit for the swarm to jump out from local optima of the problem.

4.5. Exchange Operator. Like most PSO-based algorithms, the convergence of our algorithm has a close relationship with the diversity of the swarm. In order to improve the diversity of the swarm, an exchange operator is proposed. For convenient illustration, the concept of entropy is first defined and then used to calculate the diversity of the swarm.

Definition 1 (particle entropy [35]). If the probability of the jth particle occurrence in generation T is ρ_j, its particle entropy is calculated as

$$e_j = -\rho_j \log_2 (\rho_j), \quad j = 1, 2, \dots, n, \tag{10}$$

where $\rho_j = \text{count}(X_j(T))/n$ and $\text{count}(X_j(T))$ refers to the number of occurrences of $X_j(T)$ in generation T.

Definition 2 (swarm diversity [35]). The diversity of the swarm in generation T is calculated as follows:

$$E_T = -\sum_{j=1}^{n} \rho_j \log_2 (\rho_j), \tag{11}$$

when all particles in the swarm are the same and the entropy is minimum, that is, $E_T = 0$. If there are no identical particles in the swarm, the entropy is maximum, that is, $E_T = \log_2 n$.

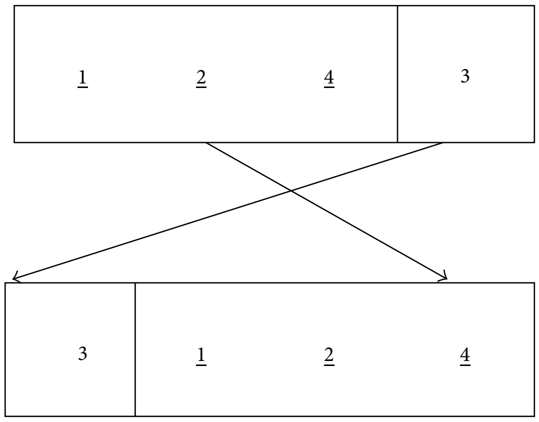

FIGURE 2: Exchange operation.

The diversity of the swarm can be reflected by the entropy value.

If the diversity of the swarm is smaller than a threshold, each particle is performing the exchange operator to generate a new particle. Taking particle $X_i(T) = (X_{i1}(T), X_{i2}(T), \dots, X_{iN}(T))^T$ as an example, first, an element of $X_i(T)$ is randomly selected. Then, the particle is divided into two parts. Finally, the left and the right parts are exchanged to generate a new particle.

Figure 2 shows an example with $X_i(T) = (1, 2, 4, 3)$. Suppose that the element randomly selected is 4; then the new particle after performing the exchange operator is $X_i(T) = (3, 1, 2, 4)$.

4.6. Algorithm Steps. Based on the above strategies, the PSO-based rescue path planning can be described as follows.

Step 1. Initialize the parameters, including the swarm size, n, the largest number of generations, $T_{\max}$, and the threshold of the diversity, ΔE. Randomly generate the initial swarm, calculate the fitness of each particle, and obtain *pbest* and *gbest*.

Step 2. Calculate the diversity of the swarm. If the diversity is smaller than ΔE, use the method proposed in Section 4.5 to improve its diversity.

Step 3. Update each particle using the method proposed in Section 4.2.

Step 4. Perform the rapid local search operator proposed in [34] to further exploit the best solution in the neighborhood of each particle.

Step 5. Update *pbest* of each particle by the method in Section 4.3.

Step 6. Select *gbest* adopting the method proposed in Section 4.4.

Step 7. Judge whether the swarm evolves for $T_{\max}$ generations. If yes, stop the evolution of the swarm, and output the optimal solution(s); otherwise, go to Step 2.

4.7. Algorithm Analysis. Solis and Wets [36] proposed a sufficient condition that a stochastic optimization algorithm converges to the global optimal solution with the probability of 1. For the purpose of analysis, the main conclusions are restated as follows.

Hypothesis 1. F s.t.$\{f(X^T)\}_{T=0}^{\infty}$ is an increasing function; that is, if $f(F(X,\xi)) \geq f(X)$ and $\xi \in U$, then $f(F(X,\xi)) \geq \max\{f(X), f(\xi)\}$. Where F is the iteration function which was used to generate new individuals, U is the feasible region for the variables.

Hypothesis 2. For a Borel subset of U, denoted as A, if its measure $\theta(A) > 0$, $\prod_{T=0}^{\infty}(1-\mu_T(A)) = 0$ is held, where $\mu_T(A)$ is the probability of gaining the element(s) in A by adopting some strategies.

Lemma 3. *Suppose that f is a measurable function, U a measurable subset, $\{X^T\}_{T=0}^{\infty}$ the resulting sequence generated by an algorithm, and Hypotheses 1 and 2 are held, then $\lim_{T\to\infty} p(X^T \in R_\varepsilon) = 1$ is held, where $p(X^T \in R_\varepsilon)$ is the probability of generating $X^T \in R_\varepsilon$ by the algorithm in the Tth generation and R_ε is the set of global best solutions.*

Lemma 4. *Assume that the objective, f, that PSO optimizes is measurable, and its solution space, U, is a measurable set, then the modified PSO can converge to the global best solution with the probability of 1.*

Proof. From the lemma, we need only to prove that the modified PSO can meet Hypotheses 1 and 2.

(1) The iteration function, F, of the modified PSO can be attributed to the following formula:

$$F\left(p_g(T), p_i(T)\right) = \begin{cases} p_g(T), & f\left(p_g(T) \geq p_i(T)\right), \\ p_i(T), & f\left(p_g(T) < p_i(T)\right), \end{cases} \quad (12)$$

where T is the number of generations and the resulting sequence of the algorithm is $\{p_g(T)\}_{T=0}^{\infty}$. It is clear that formula (12) meets Hypothesis 1.

(2) Set X_i^k as the kth result of particle X_i after the exchange, insertion, inversion, and the rapid local search operations. If these operations run all the time, the distribution of particles remains unchanged. From [36], the union set of particles must include U, that is, $U \subseteq \bigcup_{i=1}^{n} M_{i,k}$, where $M_{i,k}$ is the kth support set of particle X_i.

In the modified PSO, the velocity of a particle is convergent. These operations have no influences on the result in limited generations, so for $\forall A \subset U$, as long as its measure $\theta(A) > 0$, there is always a particle of the modified PSO that can reach A, which meets Hypothesis 2. □

Lemma 5. *The modified PSO is convergent to the global best solution with the probability of 1.*

Proof. Since the insertion and inversion operators perform on the set that is used to supply the global best solution, they have the role of transforming the global best solution, so it is enough to prove that the modified PSO converges to the global best solution with the probability of 1, which is equivalent to formula (7) satisfying $V_i(T) = 0(T \to \infty)$. Without loss of generality, the following assumption is given:

(1) only one swarm evolves in the modified PSO;

(2) the global best solution after the above operations is approximate to the current global best solution.

Based on the above assumptions, the velocity updating formula of particle can be written as follows:

$$\begin{aligned} V_{ij}(T+1) &= wV_{ij}(T) + r_1c_1\left(p'_{ij}(T) - X_{ij}(T)\right) \\ &\quad + r_2c_2\left(p_{gj}(T) - X_{ij}(T)\right). \end{aligned} \quad (13)$$

While $p'_{ij}(T) \approx p_{gj}(T-a)$, and the iteration function of F is bounded and $f(p_g(T)) \leq f(p_g(T+1))$, so

$$\lim\left[p_g(T) - p_g(T-a)\right] = 0, \quad (14)$$

and then

$$p'_{ij}(T) \approx p_{gj}(T) + \varepsilon(T), \quad (15)$$

where $\lim_{T\to\infty}\varepsilon(T) = 0$, so we can get

$$\begin{aligned} &V_{ij}(T+1) \\ &\quad = wV_{ij}(T) + (r_1c_1 + r_2c_2)\left(p_{gj}(T) - X_{ij}(T)\right) + \varepsilon(T). \end{aligned} \quad (16)$$

Now, obviously, the modified PSO algorithm can be approximated as the traditional PSO, and the particle's velocity of the traditional PSO model is convergent, so, the velocities of particles in the algorithm can converge to zero, if the number of generations is enough, i.e., $V_i(T) = 0(T \to \infty)$, As a result, the modified PSO thus converges to the global best solution with the probability of 1. □

5. Simulations

In order to verify the validity of the proposed method, simulations are done by using MATLAB software on a PC computer. The configuration of PC is P4 and 2.66 GHz, and the RAM memory is 512 M.

5.1. Parameters Setting. The related parameters of PSO are set as follows: $c_1 = c_2 = 2$, the linear inertia weight, $\omega = \omega_{\max} - (\omega_{\max} - \omega_{\min}) * T/T_{\max}$, where $\omega_{\max} = 0.9$, $\omega_{\min} = 0.4$ [37], the swarm size is 200, and the largest number of generations is 500. The environment is a two-dimensional plane of $100*100$, the strength threshold is 1.0, the velocity of the robot is 3, ST is equal to 5, the initial strength (before the disaster), σ_o, is equal to 30, the position of the disaster source is $D(65, 70)$, and the maximum radius of the region influenced by the disaster is equal to 150.

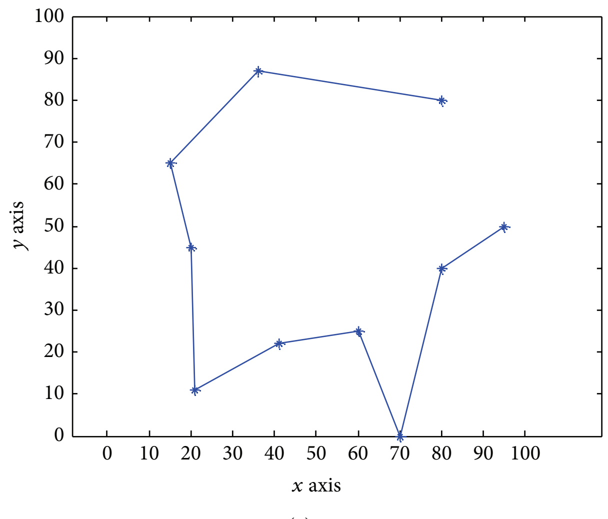

Figure 3: Two different rescue paths with the same number of rescued targets: (a) the rescue path corresponding to row one in Table 1, (b) the rescue path corresponding to row two in Table 1.

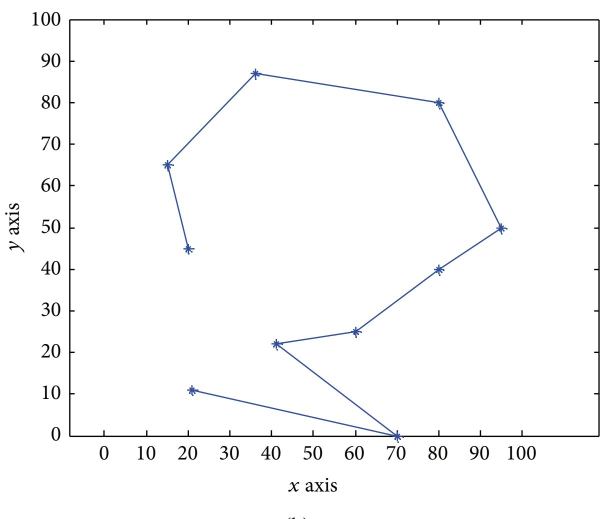

Table 1: Results of rescuing ten targets.

	1	2	3	4	5	6	7	8	9	10	F
1	4	7	6	5	10	8	2	1	3	9	7
2	5	6	7	4	9	3	2	8	1	10	7
3	6	7	4	9	3	2	8	1	5	10	7
4	7	6	5	8	3	2	9	1	4	10	7
5	7	6	5	10	8	3	2	1	4	9	7

Note: F means the number of rescued targets.

5.2. Simulation Results of Two Cases

(a) Case One: Ten Targets Need to Be Rescued. Consider the situation that ten targets need to be rescued, and their positions are (70, 0), (60, 25), (80, 40), (80, 80), (20, 45), (15, 65), (36, 87), (41, 22), (95, 20), and (2, 11). From formula (4), the initial strengths (after the disaster) of all these targets are 14.03, 9.05, 6.71, 3.61, 10.30, 10.05, 6.73, 10.73, 11.66, and 17.26, respectively. Employing the proposed method, five sequences of rescuing targets in this case are listed in Table 1, and the first two rescue paths are shown in Figure 3.

Table 1 reports that the robot can rescue seven targets at most in limited time, and there are five optimal rescue paths in total, which provide more than one selection when the secondary disaster occurs.

Figure 4 is the diversity of the swarm with and without the exchange operator. From the figure, at the beginning of the evolution, the two cases have a similar diversity. However, as the number of generations increases, the case with the exchange operator performs better than that without it. Furthermore, Table 2 shows the comparison between the above two cases. Owing to a better diversity, the case with the exchange operator has a larger number of rescue paths.

(b) Case Two: 100 Targets Need to Be Rescued. Consider that there exist 100 targets to be rescued, and their positions are listed in Appendix. Their initial strengths (after the disaster) calculation method is the same as case one. Employing the proposed method, the results are listed in Table 3.

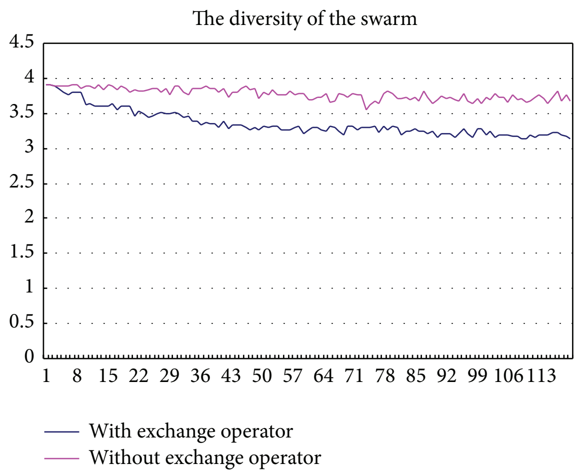

Figure 4: The diversity of the swarm with and without the exchange operator.

Table 2: Comparison between the cases with and without the exchange operator.

	F	Average number of rescue paths
With EO	7	6.3
No EO	7	2.6

Note: EO means the exchange operator, F refers to the number of rescued targets.

Table 3 reports that with the exchange operator, the robot can rescue 16 targets at most in limited time, and there is only 1.2 rescue path, whereas without the exchange operator, the maximum number of rescued targets is 11, and the average

Table 3: Comparison between the cases with and without the exchange operator.

	F	Average number of rescue paths
With EO	16	1.2
Without EO	11	11.9

Table 4: Comparison with and without the correction operator.

	F with the correction operator	F without the correction operator
Case 1	7	9
Case 2	16	21
Feasibility of solutions	Feasible	Infeasible

Table 5: Comparison between methods of updating the global best particle.

	F with the insertion and inversion operators	F without the insertion and inversion operators
Case 1	7	6
Case 2	16	14

number of rescue paths is 11.9. For the latter case, the number of rescue paths is larger; however, any path cannot complete the rescue task with the largest number of rescued targets.

5.3. Influences of Each Strategy on the Algorithm. Table 4 is the comparison between with and without the correction operation (described in Section 4.2). From Table 4, the maximum numbers of rescued targets for the above two cases are 7 and 16, respectively, whereas those without the correction operator are 9 and 21, respectively. It seems that without the correction operation is better for the two cases. However, some paths are infeasible since they have repeated targets, shown in Figure 5. In this figure, the rescue sequence for ten rescued targets without the correction operation is 10 8 9 1 7 6 7 3 2 10, suggesting that the robot rescues targets 7 and 10 twice. Therefore, it is essential to use the correction operation so as to get a feasible solution.

Table 5 is the comparison between with and without the insertion and inversion operators. Table 5 reports that the case with the insertion and inversion operators is better than its counterpart. The possible reason is that the insertion and inversion operators are added to the mechanism of updating *gbest*, and *gbest* is selected from the optional set.

Table 6 is the comparison of the above two cases between with and without the rapid local search. From Table 6, the result of the proposed method is better, which is the result that the algorithm can effectively search the neighbor of an individual by using the rapid local search. However, getting a better solution requires more time consumption.

Tables 7 and 8 are the comparison between the rapid local search and the common local search. From these two tables, both methods can gain the best solution for the above two cases. However, in terms of the time consumption, the proposed method has a greater advantage.

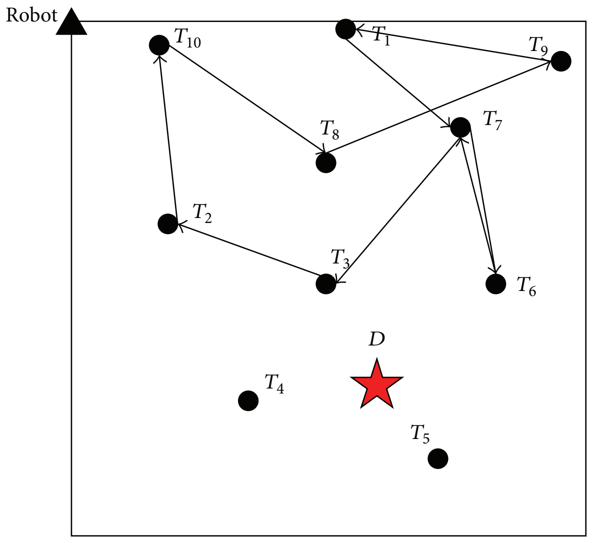

Figure 5: Rescue path without the correction operator for 10 targets.

Table 6: Comparison between with and without the rapid local search.

	F with the rapid local search	F without the rapid local search
Case 1	7	5
Case 2	16	13

Table 7: Comparison between local search methods.

	F of the proposed method	F of the common local search method
Case 1	7	7
Case 2	16	16

Table 8: Comparison of time consumption between local search methods (unit: s).

	Mean time of Case 1	Mean time of Case 2
Rapid local search	26.9	320.3
Local search	39.8	429.8

5.4. Algorithm Comparisons

(A) Comparison with Integer PSO. The proposed method and integer PSO are the same but the updating strategies, and their comparison results of the above two cases are shown in Table 9.

Table 9 tells that, in case one, the two algorithms are different in the number of rescued targets. The number of rescued targets of the proposed method is 7, whereas that of integer PSO is 6. In case two, their difference is 4, so the proposed method is better than integer PSO. Further, as the number of targets increases, the differences between these algorithms are larger.

(B) Comparison with GA. In order to verify the proposed method, the method proposed in this study is compared with

Table 9: Comparison between the proposed method and integer PSO.

	F of the proposed method	F of integer PSO
Case 1	7	6
Case 2	16	12

Table 10: Comparison between GA and the proposed method.

	F of the proposed method	F of GA
Case 1	7	5
Case 2	16	12

GA, which has the same encode and decode strategy, and the parameters of GA are set as follows: the mutation probability is 0.2, the crossover probability is 0.8, and other parameters are the same as the proposed PSO. These methods are run 30 times, and the average results are listed in Table 10.

From Table 10, the number of rescued targets of the proposed method is larger than that of GA in the two cases, since the proposed PSO algorithm preserves the optimal solution in each particle's memory and employs the rapid local search, whereas the optimal solution obtained by GA may be destroyed in the next generation.

6. Conclusions

For the problem of robot path planning for multisurvivor rescue in limited survival time after a disaster, a modified PSO is proposed to effectively solve it. When formulating the mathematical model of this problem, not only the number of rescued targets is considered but also the survivors' strength. In order to efficiently solve the established mathematical model by employing PSO, the formulas of updating a particle's velocity and position are designed. Also a new mechanism of updating *gbest* is presented, which can generate *gbest* with good performance. In addition, the rapid local search method is employed, with the purpose of helping a particle quickly search good solutions in its neighbor.

The proposed method is applied to two cases and compared with existing methods, and the simulation results show that the robot can quickly find better rescue paths by the proposed method.

The survivors' strength is considered when formulating the mathematical model of the problem, and it can reflect real-world rescue scenarios to some degree. However, some assumptions made in this study are too ideal; for example, the velocity of the robot is constant, the rescue environment remains unchanged before and after the disaster, and no obstacles exist in the rescue environment, to say a few, which would limit the application of the proposed method. So in the following research, it is essential to relax the above assumptions so as to better reflect a practical rescue scenario.

In addition, as a real-world rescue environment is much more complex than that considered in this study, some information and parameters are hard to be obtained, which makes the work done not mature enough to be published. It is another topic that needs to be further studied.

Appendix

The positions of the 100 targets in case two are listed as follows: (70, 0), (60, 25), (82, 40), (80, 80), (20, 45), (15, 65), (36, 87), (41, 22), (95, 50), (2, 11), (73, 10), (60, 35), (8, 50), (20, 90), (20, 55), (15, 76), (36, 70), (41, 12), (95, 52), (12, 17), (76, 8), (60, 15), (29, 30), (80, 70), (20, 35, (15, 55), (36, 47), (41, 12), (95, 40),(2, 31), (71, 20), (60, 70), (11, 90), (89, 20), (20, 2), (15, 40), (36, 65), (41, 6), (95, 40), (2, 15), (70, 27), (60, 10), (88, 40), (26, 60), (20, 30), (15, 32), (36, 65), (41, 60), (95, 80), (25, 10), (70, 35), (60, 42), (80, 28), (60, 0), (20, 45), (15, 50), (36, 15), (41, 10), (95, 9), (2, 22), (74, 28), (60, 18), (80, 69), (82, 28), (20, 40), (15, 18), (36, 23), (41, 94), (95, 21), (8, 14), (70, 22), (60, 23), (80, 24), (80, 30), (20, 26), (15, 75), (36, 76), (41, 15), (95, 71), (21, 85), (70, 14), (60, 63), (70, 55), (80, 80), (20, 14), (15, 99), (36, 60), (41, 11), (95, 100), (20, 52), (0, 19), (25, 71), (40, 51), (84, 80), (45, 45), (65, 65), (87, 87), (22, 22), (50, 50), and (11, 11).

Conflict of Interests

The authors declare no conflict of interests.

Acknowledgments

This research is jointly supported by the Fundamental Research Funds for the Central Universities (no. 2013XK09), the Jiangsu Planned Projects for Postdoctoral Research Funds (no. 1301009B), and the China Postdoctoral Science Foundation funded project (no. 2014T70557). Thanks to Dr. Edward C. Mignot, Shandong University, for linguistic advice.

References

[1] S.-H. Qian, S.-R. Ge, Y.-S. Wang, and C.-Q. Liu, "Research status of the disaster rescue robot and its applications to the mine rescue," *Robot*, vol. 28, no. 2, pp. 350–354, 2006 (Chinese).

[2] Z.-J. Cai, D.-B. Sun, Y.-Q. Qin, and N. Li, "Study of robot path planning based on framework space approach," *Computer and Digital Engineering*, vol. 4, no. 34, pp. 88–90, 2006.

[3] N. Ghita and M. Kloetzer, "Trajectory planning for a car-like robot by environment abstraction," *Robotics and Autonomous Systems*, vol. 60, no. 4, pp. 609–619, 2012.

[4] W. W. Charles, "Global path planning using artificial potential fields," in *Proceedings of IEEE International Conference on Robotics and Automation*, pp. 316–321, Scottsdale, Ariz, USA, 1989.

[5] D. Roy, "Study on the configuration space based algorithmic path planning of industrial robots in an unstructured congested three-dimensional space: an approach using visibility map," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 43, no. 2–4, pp. 111–145, 2005.

[6] L. E. Kavraki, P. Švestka, J.-C. Latombe, and M. H. Overmars, "Probabilistic roadmaps for path planning in high-dimensional configuration spaces," *IEEE Transactions on Robotics and Automation*, vol. 12, no. 4, pp. 566–580, 1996.

[7] W. J. Yang, "Anytime synchronized-biased-greedy rapidly-exploring random tree path planning in two dimensional complex environments," *International Journal of Control, Automation and Systems*, vol. 9, no. 4, pp. 750–758, 2011.

[8] Y.-Y. Han, Q.-K. Pan, J.-Q. Li, and H.-Y. Sang, "An improved artificial bee colony algorithm for the blocking flowshop scheduling problem," *The International Journal of Advanced Manufacturing Technology*, vol. 60, no. 9–12, pp. 1149–1159, 2012.

[9] A. Z. Mohamed, S. H. Lee, H. Y. Hsu, and N. Nath, "A faster path planner using accelerated particle swarm optimization," *Artificial Life and Robotics*, vol. 17, no. 2, pp. 233–240, 2012.

[10] M. F. Aly and A. T. Abbas, "Simulation of obstacles' effect on industrial robots' working space using genetic algorithm," *Journal of King Saud University—Engineering Sciences*, vol. 26, no. 2, pp. 132–143, 2014.

[11] X. Chen, Y. Kong, X. Fang, and Q. Wu, "A fast two-stage ACO algorithm for robotic path planning," *Neural Computing and Applications*, vol. 22, no. 2, pp. 313–319, 2013.

[12] J. Kennedy and R. C. Eberhart, "Particle swarm optimization," in *Proceedings of IEEE International Conference on Neural Networks (ICNN '95)*, pp. 1942–1948, Perth, Western Australia, December 1995.

[13] J. J. Liang, H. Song, B. Y. Qu, and Z. F. Liu, "Comparison of three different curves used in path planning problems based on particle swarm optimizer," *Mathematical Problems in Engineering*, vol. 2014, Article ID 623156, 15 pages, 2014.

[14] H.-C. Huang, "FPGA-based parallel metaheuristic PSO algorithm and its application to global path planning for autonomous robot navigation," *Journal of Intelligent & Robotic Systems*, 2013.

[15] Y.-K. Liu, M.-K. Li, C.-L. Xie, M.-J. Peng, and F. Xie, "Path-planning research in radioactive environment based on particle swarm algorithm," *Progress in Nuclear Energy*, vol. 74, pp. 184–192, 2014.

[16] S. Kibler and D. Raskovic, "Coordinated multi-robot exploration of a building for search and rescue situations," in *Proceedings of the 44th IEEE Southeastern Symposium on System Theory*, pp. 159–163, Jacksonville, Fla, USA, 2012.

[17] P. Mandal, R. K. Barai, M. Maitra, S. Roy, and S. Ghosh, "Autonomous robot coverage in rescue operation," in *Proceedings of the International Conference on Computer Communication and Informatics (ICCCI '12)*, pp. 1–5, Coimbatore, India, January 2013.

[18] A. H. M. I. Ferdous, S. M. Masudur Rahamn, M. S. H. Nizami, M. M. Abdul Quadir, and M. W. Ullah, "Path planning for automated robotic rescue system," in *Proceedings of the 16th IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD '12)*, pp. 709–712, Wuhan, China, May 2012.

[19] B. Doroodgar, M. Ficocelli, B. Mobedi, and G. Nejat, "The search for survivors: Cooperative human-robot interaction in search and rescue environments using semi-autonomous robots," in *Proceedings of IEEE International Conference on Robotics and Automation Anchorage Convention District (RAACD'10)*, pp. 2858–2863, Anchorage, Alaska, USA, May 2010.

[20] Z.-J. Tian, L.-Y. Zhang, and W. Chen, "Improved algorithm for navigation of rescue robots in underground mines," *Computers and Electrical Engineering*, vol. 39, no. 4, pp. 1088–1094, 2013.

[21] X. Zhang, M. Wu, J. Peng, and F. Jiang, "A rescue robot path planning based on ant colony optimization algorithm," in *Proceedings of the International Conference on Information Technology and Computer Science (ITCS '09)*, pp. 180–183, IEEE, Kiev, Ukraine, July 2009.

[22] N. Basilico and F. Amigoni, "Exploration strategies based on multi-criteria decision making for searching environments in rescue operations," *Autonomous Robots*, vol. 31, no. 4, pp. 401–417, 2011.

[23] S.-Y. Chien, H.-D. Wang, and M. Lewis, "Human vs. algorithmic path planning for search and rescue by robot teams," in *Proceedings of the 54th Human Factors and Ergonomics Society Annual Meeting (HFES '10)*, pp. 379–383, Jacksonville, Fla, USA, October 2010.

[24] E. Masehian and D. Sedighizadeh, "Multi-objective PSO- and NPSO-based algorithms for robot path planning," *Advances in Electrical and Computer Engineering*, vol. 10, no. 4, pp. 69–76, 2010.

[25] A. Adham and P.-A. Somnuk, "Applying area extension PSO in robotic swarm," *Journal of Intelligent and Robotic Systems: Theory and Applications*, vol. 58, no. 3-4, pp. 253–285, 2010.

[26] T.-K. Lee, S.-H. Baek, Y.-H. Choi, and S.-Y. Oh, "Smooth coverage path planning and control of mobile robots based on high-resolution grid map representation," *Robotics and Autonomous Systems*, vol. 59, no. 10, pp. 801–812, 2011.

[27] Y. Zhang, D.-W. Gong, and J.-H. Zhang, "Robot path planning in uncertain environment using multi-objective particle swarm optimization," *Neurocomputing*, vol. 103, pp. 172–185, 2013.

[28] S. Liu, D. Sun, and C. Zhu, "Coordinated motion planning for multiple mobile robots along designed paths with formation requirement," *IEEE/ASME Transactions on Mechatronics*, vol. 16, no. 6, pp. 1021–1031, 2011.

[29] Q.-R. Tang and P. Eberhard, "A PSO-based algorithm designed for a swarm of mobile robots," *Structural and Multidisciplinary Optimization*, vol. 44, no. 4, pp. 483–498, 2011.

[30] Y. Xue and H. Liu, "Optimal path planning in complex indoor environment based on improved PSO," *Journal of Computational Information Systems*, vol. 7, no. 6, pp. 2158–2165, 2011.

[31] Y. Kuwata and S. Takada, "Rescue ability for earthquake causality during the 1995 KOBE earthquake," http://www.lib.kobe-u.ac.jp/handle_kernel/00231322.

[32] J. Kennedy and R. C. Eberhart, "A Discrete binary version of the particle swarm algorithm," in *Proceedings of the IEEE International Conference on Systems, Man, and Cybernetics*, pp. 4104–4108, IEEE, Orlando, Fla, USA, October 1997.

[33] J.-C. Bansal and K. Deep, "A modified binary particle swarm optimization for knapsack problems," *Applied Mathematics and Computation*, vol. 218, no. 22, pp. 11042–11061, 2012.

[34] L. Wang, Q.-K. Pan, P. N. Suganthan, W.-H. Wang, and Y.-M. Wang, "A novel hybrid discrete differential evolution algorithm for blocking flow shop scheduling problems," *Computers & Operations Research*, vol. 37, no. 3, pp. 509–520, 2010.

[35] K.-Z. Tang, X. Xiao, J.-H. Jia, and X. Xu, "Adaptive particle swarm optimization algorithm based on discrete stimate strategy of diversity," *Journal of Nanjing University of Science and Technology*, vol. 37, no. 3, pp. 344–349, 2013.

[36] F. J. Solis and R. J. Wets, "Minimization by random search techniques," *Mathematics of Operations Research*, vol. 6, no. 1, pp. 19–30, 1981.

[37] Y. H. Shi and R. C. Eberhart, "A modified particle swarm optimizer," in *Proceedings of the IEEE International Conference on Evolutionary Computation (ICEC '98)*, pp. 63–79, Alaska, Alaska, USA, May 1998.

Copyright of Mathematical Problems in Engineering is the property of Hindawi Publishing Corporation and its content may not be copied or emailed to multiple sites or posted to a listserv without the copyright holder's express written permission. However, users may print, download, or email articles for individual use.