Intelligent Automation And Soft Computing, 2020
Vol.26,no.3, 447–454
DOI: 10.32604/iasc.2020.013921

# QRDPSO: A new optimization method for swarm robot searching and obstacle avoidance in dynamic environments

**Mehiar, D.A.F., Azizul, Z.H.* and Loo, C.K.**

Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, 50603 Kuala Lumpur, Malaysia

**ABSTRACT**

In this paper we show how the quantum-based particle swarm optimization (QPSO) method is adopted to derive a new derivation for robotics application in search and rescue simulations. The new derivation, called the Quantum Robot Darwinian PSO (QRDPSO) is inspired from another PSO-based algorithm, the Robot Darwinian PSO (RDPSO). This paper includes comprehensive details on the QRDPSO formulation and parameters control which show how the swarm overcomes communication constraints to avoid obstacles and achieve optimal solution. The results show the QRDPSO is an upgrade over RDPSO in terms of convergence speed, trajectory control, obstacle avoidance and connectivity performance of the swarm.

**KEY WORDS**: Swarm robotics, particle swarm optimization (PSO), quantum behaving particle, search and rescue simulations.

## 1 INTRODUCTION

SWARM robots is a highly popular robotics system that is composed of a group of interacting intelligent robots. The system takes advantage of the cooperation among individuals in a swarm in solving a particular task. Swarm robotics is influenced from bio-inspired algorithms such as the Particle Swarm Optimization (PSO). A common optimization task for swarm robots is the search and rescue problem where a swarm is deployed to find the global best (victims) and the performance is measured based on how fast the global best is discovered. A popular derivation of the PSO algorithm for swarm robots is called the Robotic Darwinian PSO (RDPSO).

The main aim of the RDPSO algorithm is to improve the efficiency of PSO-based algorithm to allow search to take place at a faster rate. The RDPSO extends the PSO algorithm using evolution to reduce overlapping search area so the robots move away from provincial (local) optima. Despite the significance of the RDPSO algorithm for multi-robot exploration, there remain important gaps for the searching capabilities such as premature and slow convergence in finding global best, and collisions between robots (Couceiro et al., 2014; Dadgar et al., 2017; Kumar et al., 2017; Sanchez et al., 2018). The quantum computing theory has several advantages that can improve the searching capabilities of these PSO-based algorithms.

The quantum-based PSO (QPSO) such as the derivation based on the delta potential well model of PSO (Sun et al., 2004 (June)) introduced wave function to represent quantum behavior in PSO-based algorithm. The algorithm has shown improved convergence speed and solution accuracy in continuous optimization problems. Two recent works adopted the method and showed improvement in global search ability for particles in optimal power flow problem (Yuan et al., 2015) and solved particle distribution and localization (Zuo et al., 2018). However, these works are particle-based and not robot-based. One robot-based work implemented QPSO to solve robot path selection (Tokgo & Li, 2014) however the work focused in free environment (no obstacles) and as such not suitable in search and rescue simulations.

In this paper, we introduce the Quantum Robot Darwinian PSO (QRDPSO) algorithm to optimize swarm robot behavior in search and rescue simulation. We show how the algorithm is formed by extending main RDPSO parameters representing quantum behavior in the form of wave function. To show robustness of the algorithm, we examine the global best convergence and robot collision occurrences for different quantities of robots in a MATLAB

CONTACT Azizul, Z.H. zati@um.edu.my

simulation. The experimental results show the QRDPSO is more efficient, stable and faster to reach optimal solution in comparison to the RDPSO algorithm.

The paper is structured as follows. In the next section, we present the literature review of the work. In section 3, we propose a method of parameters control for the QRDPSO algorithm. Experiment procedures are described in Section 4. Section 5 discusses the experiment results and the paper is concluded in Section 6.

## 2 RELATED WORK

THIS section presents the technical background of the PSO approach. In this review, we analyze the mathematical models and show how they work to optimize the search and obstacle avoidance in a swarm system. We begin by reviewing the traditional PSO, followed by the RDPSO and finally the QPSO.

### 2.1 Classical PSO

The PSO (Kennedy & Eberhart, 1995) is an optimization algorithm which models a set of potential problem solutions as a swarm of particles moving about in a virtual search space. Most importantly, the PSO contains particles that search for global optima. Each of the particle has position $x_n[t+1]$ and velocity $v_n[t+1]$, which depends on several vectors, the position $x_n[t]$, velocity $v_n[t]$, local best $X_n[t]$, global best $G_n[t]$, and the performance vector $F(x_n(t))$. The path of the particle is measured following Dadgar et al. (2016):

$$\begin{aligned} v_n[t+1] = {} & WV_n[t] + \\ & c_1 r_1 (g_n^{\sim}[t] - x_n[t]) + \\ & c_2 r_2 (x_n^{\sim}[t] - x_n[t]) \end{aligned} \tag{1}$$

$$x_n[t+1] = x_n[t] + v_n[t+1] \tag{2}$$

The trend to adopt the PSO in solving swarm searching is majorly due to the ease of implementation where only few parameters require adjustment. However, the approach poses several disadvantages such as the inability to work out the problems of scattering and optimization, premature convergence and suffers from partial optimism causing inexact regulation of speed and direction (Cai et al., 2013). These limitations mean the PSO algorithm will work in some problem to get optimized solution but fail in others and get sub-optimal solutions. For this reason, the RDPSO as an enhancement of PSO offers solutions for robots to escape local optima.

### 2.2 Extension to RDPSO

The RDPSO have been a popular choice after a report took note it is faster and more accurately converges than other optimization approaches in finding optimal solutions (Couceiro et al., 2014). In terms of equation, the RDPSO has the same coefficients as the PSO but introduces new coefficients for static and dynamic obstacle avoidance. The use of robots (instead of particles) means that each robot can have obstacle sensor with a detection radius $r_s$ and a sensing function $g(x_n[t])$ defined upon sensor data collection (Nakisa et al., 2015). Hence, the equation (1) of velocity is extended as follows:

$$\begin{aligned} v_n[t+1] = {} & WV_n[t] + \\ & c_1 r_1 (g_n^{\sim}[t] - x_n[t]) + \\ & c_2 r_2 (x_n^{\sim}[t] - x_n[t]) - \\ & c_3 r_3 \left(\widetilde{x_n^{g}}[t] - x_n[t]\right) \end{aligned} \tag{3}$$

The $c_3$ and $r_3$ from (3) are the obstacle's capability weight and random, and $\widetilde{x_n^{g}}[t]$ is the position of robot $n$ which optimizes the increasing or decreasing of the monotonic and positive sensing function for solution convergence (Cecconi & Campenni, 2010). The advantages of the RDPSO are that it is adaptable to huge population of robots, include parameters of the real surroundings for obstacle avoidance and observe faster and accurate convergence compared to other approaches. Nevertheless, collision between robots is still an issue (Couceiro et al., 2013 (June)) and the reviewer of modern heuristic algorithms can find more comprehensive algorithms that can reach the global best in a shorter time. This inspired supplanting the PSO with more sophisticated approach such as the quantum-behaving particles (QPSO).

### 2.3 Quantum Delta Potential Well Model of PSO (Quantum-behaved PSO)

The QPSO describes the state of particles, which design allows potential for better global search ability. In quantum space-time, the quantum state of a particle is shown by the wave function $\Psi(\bar{x}, t)$ rather than position $\bar{x}$ and velocity $\bar{v}$. This is due to the dynamism of the particles' behaviors, which is developed in different direction from that in PSO, that the exact values of $\bar{x}$ and $\bar{v}$ cannot be determined. We can only assume the probability of particle $s$ showing up in position $\bar{x}$ from the partial differential equation $|\Psi(\bar{x}, t)|^2$ which depends upon the potential field in which the particle lies (Sun et al., 2004 (December)):

$$X_{i,n+1}^{j} = p_{i,n}^{j} \pm \alpha \left| X_{i,n}^{j} - p_n^{j} \right| \ln\left(\frac{1}{u_{i,n+1}^{j}}\right) \tag{4}$$

$$X_{i,n+1}^{j} = p_{i,n}^{j} \pm \alpha \left| X_{i,n}^{j} - C_n^{j} \right| \ln\left(\frac{1}{u_{i,n+1}^{j}}\right) \tag{5}$$

In (4) and (5), $C$ is the $m$ best positions vector and $n$ is defined as the number of iterations. $j$ is the component of the position of particle $i$ where $j^{th} (1 \leq j \leq N)$ for the particle $i (1 \leq i \leq M)$ at the $(n+1)$ position, where $N$ is space dimensions and $M$ refers to

number of particles. $P_n^j$ is the centre of the $N$-dimension Hilbert space with a δ potential well. It is the best previous position, the position giving the best objective function value of fitness value, of the particle $i$ (also refers to as *Personal Best*).

$$p_{i,j} = \varphi . p_{i,j}(t) + (1 - \varphi) . G_j(t) \quad (6)$$

In (6), $G(t) = (G_1(t), (G_2(t), \ldots, (G_D(t))$ describes the optimal position vector of the group's particle in space with dimension $D$ (also refers to as *Global Best*). The $\varphi$ refers to population size, $p_{i,j}$ is the local attractor of each particle, and $\mu$ is a uniformly distributed random number between 0 and 1 (Sun et al., 2011). (6) can be rewritten as:

$$p_{i,j}(t+1) = G_j(t) + \varphi.(p_{i,j}(t) - G_j(t)) \quad \text{where } (1 \le i \le N, 1 \le j \le D) \quad (7)$$

It can be observed from (4) and (7) that the local attractor $p_{i,j}(t+1)$ is associated with the difference between the best position in the swarm $G_j(t)$ and the best position of the current particle $P_{i,j}(t)$. Its position $x_{i,j}(t+1)$ is associated with the difference between the average positions of current particles $c_i(t)$ and the position of the particle itself $x_{i,j}(t)$.

The usage of these position vectors ensure stable convergence between particles which promotes faster and stronger searchers. The only problem is, it is not directly usable to describe multi-robot applications, unlike the RDPSO. In the next section, we show how we adopted the QPSO in our proposed PSO derivation for swarm robot application.

## 3 THE PROPOSED QRDPSO ALGORITHM

IN this section we show how the proposed QRDPSO is derived. The aim of this algorithm is to propose a new cost or fitness function in a manner that it would guide the robot to the global best while avoiding obstacles. Maintaining communication is key (Baghaei & Agah, 2013) so when a robot moves from any position to the target position, it is able to avoid both static and dynamic obstacles in the environment.

For obstacle avoidance, we assume every robot is equipped with sensors suitable for finding obstacle location within a finite sensing radius $r_s$. The sensing function $q(x_i[t])$ is also defined. This function describes the data collected from the sensor such as the distance from robot to obstacles or detected objects from the surrounding.

### 3.1 Individual robot-obstacle susceptibility rate

The trajectories of individual robots passing an obstacle are given by the standard deviation value for the obstacle susceptibility $\sigma(q_i(t))$ and the standard deviation value for the current position of the robot $\sigma(x_i(t))$, respectively. At any point, the individual robot's susceptibility is defined as follows:

$$is_{i,j}(t) = is_{i,1}(t), is_{i,2}(t), \ldots, is_{i,D}(t) = \frac{\sigma\left(q_{i,1}(t)\right)}{\sigma\left(x_{i,1}(t)\right)}, \frac{\sigma\left(q_{i,2}(t)\right)}{\sigma\left(x_{i,2}(t)\right)}, \ldots, \frac{\sigma(q_{i,D}(t))}{\sigma(x_{i,D}(t))} (1 \le i \le N) \quad (8)$$

For optimization, ideally the value of $is_i(t)$ and susceptibility should be directly proportional to each other, where $(0 < is_i(t) \le 1)$. If the value of $is_i(t)$ remains 1, it means the robot successfully avoided all obstacles. In other words, when the robot $i$ reaches $\sigma\left(q_{i,1}(t)\right)/\sigma\left(q_{i,1}(t)\right) = 1$, it means the robot has changed positions without hitting any obstacle.

The information regarding the obstacle in the surrounding would come from range finders such as an ultrasonic or laser sensor mounted on the robot. These sensors work by transmitting sound waves or light and wait for its reflectance (time-of-flight rules). Following the *speed=distance/time* equation, these sensors can easily measure the distance between the robot and any solid obstacles or objects. To improve the detection, we join readings of the sensors on one robot during its movement and make comparison of the output value with a predefined threshold. Thus, the final value of (8) may lie in the interval (0,1) which is useful to determine the trajectories of the robots.

### 3.2 Communication rate between the robots

For the swarm robot to maintain communication, we describe the connectivity between robots as a link matrix $L = \{l_{i,f}\}$ which can be calculated as functions of either distance $d_{max}$ or signal quality $z_{min}$ or both. Together, they form the adjacency matrix $A = \{a_{if}\}$ and can be defined as follows:

$$a_{ij} = \begin{cases} 1, nodes\ i\ and\ f\ connected \\ 0, no\ connection \end{cases} \quad (9)$$

Using the hop distances, i.e., the minimum number of hops to interact with non-adjacent (far away) robots, the zero-valued off-diagonal entries in the adjacency matrix can be manipulated to create a multi-hop connectivity matrix $C^k = \{c_{i,f}^k\}$, where the entry $(i, f)$ represent the least number of hop count needed to connect nodes $i$ and $f$. $k$ represents the iteration which varies with the number of hops the network can handle. The calculation of the connectivity matrix can be defined as follows:

$$c_{ij}^k = \begin{cases} h, i\ connected\ to\ f\ by\ h \le k\ hops \\ 0, otherwise \end{cases} \quad (10)$$

In the case where each robot corresponds to a node, to overcome the no connectivity between them, the desired position of each robot $X_{i,n+1}^{j}$ must be controlled since it influences the link matrix. One way to ensure the full connectivity is to *force* each robot to communicate with its nearest neighbor that has not chosen it as its nearest neighbor. Since the connectivity depends on the distance or signal quality, connectivity between the nodes is determined by computing the minimum or maximum value of each line of the adjacency matrix $A$, after excluding zeros and the $(i, f)$ pairs previously chosen. Therefore, a connectivity function $m(x_i(t))$ is defined.

### 3.3 Individual robot connectivity rate

The connectivity rate of individual robots within a swarm are given by the standard deviation value of the connectivity function $\sigma(m_i(t))$ and the standard deviation value of the current position of the robot $\sigma(x_i(t))$, respectively. At any point, the individual robot connectivity is defined as follows:

$$im_{i,j}(t) - \left(im_{i,1}(t), im_{i,2}(t), \dots, im_{i,D}(t)\right) = \frac{\sigma\left(m_{i,1}(t)\right)}{\sigma\left(x_{i,1}(t)\right)}, \frac{\sigma\left(m_{i,2}(t)\right)}{\sigma\left(x_{i,2}(t)\right)}, \dots, \frac{\sigma(m_{i,D}(t))}{\sigma(x_{i,D}(t))} (1 \le i \le N) \quad (11)$$

For optimization, ideally the value of $im_i(t)$ and the susceptibility should be directly proportional to each other, where $(0 < im_i(t) \le 1)$. If the value of $im_i(t)$ remains 1, it means the swarm of robots successfully connected to each other. In other words, when the particle $i$ reaches $\sigma\left(m_{i,1}(t)\right)/\sigma\left(x_{i,1}(t)\right) = 1$, it means the robot is in a position where it has connectivity with its neighbor. This sets the connectivity constraints of the robots' movements, while at the same time reducing the calculation overhead so that the robots can plan their movements considering the communication constraint, based on one value only. Moreover, this equation is adaptable with different communication techniques to permit the robots to correspond with one another.

### 3.4 Formulating the QRDPSO

Based on the inertia-weighted parameters for the QPSO algorithm mentioned in previous sub-sections, if we include parameters $is_{ij}(t)$ and $im_{ij}(t)$ into the term $(X_{i,n}^{j} - C_n^j)$ of (5), we obtain the QRDPSO as follows:

$$X_{i,n+1}^{j}(t+1) = P_{i,n}^{j} \pm \left(\alpha_1 \left|X_{i,n}^{j} - C_n^j\right| + \alpha_2 \left|X_{i,n}^{j} - im_n^j\right| + \alpha_3 \left|X_{i,n}^{j} - is_n^j\right|\right) \ln\left(\frac{1}{u_{i,n+1}^{j}}\right) \quad (12)$$

From (12), it can be noticed that the values of $im$ determines the movement of the robot. Based on the mounted sensors' readings and the communication signals' strength between the robots, each one has the option to move in search for a better objective function, but that movement is bounded within the limitations of these communication constraints. However, benefiting from the searching capabilities of the quantum-behaving particles in QPSO, hypothetically speaking, the robots can avoid any local optimal solution and reach the global optimal solution within a shorter time.

### 3.5 The QRDPSO control architecture design

We propose a control architecture design for the QRDPSO (see Figure 1). In the architecture, the low level control (LLC) receives the desired position $x_n^d[t+1]$ and computes the kinematic model. The output of the LLC is represented by the rotation $\theta_n[t+1]$ and the distance $h_n[t+1]$. These outputs are useful so the robot can turn to face the target (robot-target spatial alignment) and move the distance calculated towards it. Whenever a new position is calculated, the current robot position $x_n[t+1]$ is updated. This new position and the corresponding value from the objective function $f(x_n[t+1])$ needs to be shared between connected robots in the swarm so cooperation can emerge.

We propose that this QRDPSO control architecture can be useful for testing with different types of robotic systems. If the hardware changes, only the LLC in the QRDPSO control architecture needs to be replaced.

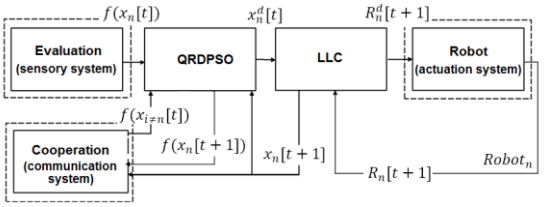


**Figure 1.** QRDPSO control architecture

The following section describes the setup and experiment done taking into consideration the scope of the search and rescue task and the control parameters proposed to compare the performance of QRDPSO against its predecessor, the RDPSO.

## 4 EXPERIMENT DONE

IN this section we describe the experiment done to condition and determine suitable coefficients for the QRDPSO. We then describe the environment setup to measure the performance of QRDPSO in comparison to its predecessor, the RDPSO. For the experiments, we define a swarm of robots in a MATLAB simulator running on a high performance workstation Lenovo

W530 with Intel iCore7, 2.67GHz processor and 16GB of RAM.

### 4.1 Parameter control

Conditioning the parameters within the QRDPSO is important so we ran a preliminary numerical evaluation for the coefficients that controls the swarm's susceptibility to change $\mu$, and the important cognitive coefficients $\alpha_1$, $\alpha_2$ and $\alpha_3$ which represent the swarm robot's convergence, trajectory and connectivity, respectively.

Figure 2 shows the behavior of the swarm is susceptible to changes $\mu$ when posed with obstacles to avoid. When $\mu$=2 (248 positions in 788 iterations), the path of the swarm is linear for exploration indicating the swarm is stable and converges to an optimal solution. When higher $\mu$=2.5 is used (270 positions in 788 iterations), the path of the swarm is linear at first but not stable at later iterations extending the time to reach the global solution. When smaller $\mu$=1.5 is used (250 positions in 550 iterations), the swarm moves slowly and finds it difficult to converge on a solution. Consequently, the swarm gets stuck in a sub-optimal solution with this value. Thus, the value $\mu$=2 is selected for the swarm's susceptibility to changes

Figure 3 shows how the parameter $\alpha_1$ influences the speed of convergence for the swarm. When $\alpha_1$=1.5 (250 positions in 780 iterations), superior performance is observed on the speed of convergence and sub-optimal solution avoidance. When higher $\alpha_1$=2 is used (260 positions in 780 iterations), the swarm takes longer time to find the victim (optimal solution) because of unstable movement behavior. The swarm did complete the task but not directly. The instability of the movement made the swarm to lose the optimal path before reaching the victim. When smaller $\alpha_1$=1 is used (280 positions in 850 iterations), the swarm shows very instable behavior and lost its way in a sub-optimal solution. The swarm did not show potential to complete its task to reach the victim. Thus, the value $\alpha_1$=1.5 is selected to promote the swarm's speed of convergence.

Figure 4 shows an analysis on how the parameter $\alpha_2$ influences the swarm's trajectory in getting around the obstacles. When $\alpha_2$=2.5, the trajectory behavior is worst as the swarm is trapped in sub-optimal solution and may not reach the victim. Similarly when $\alpha_2$=1.5, the swarm is moving chaotically and are not able to avoid obstacles. Only when $\alpha_2$=2 the robots in the swarm are able to move around the obstacles found in its surrounding, then maintain its course to reach the victim. The value $\alpha_2$=2 allows the swarm to successfully avoid obstacles in its path thus selected to condition the equation proposed.

Parameter $\alpha_3$ influences the communication behavior of the swarm. With each robot considered as a network node, the required position of the robot $x_n[t+1]$ must be controlled since it affects the adjacency matrix A. The adjacency matrix depends on the maximum interaction range $d_{max}$ or minimum signal quality represented by the link matrix $L = \{l_{i,j}\}$ for an N-node network where each entry represents the link between the robot $i$ and $j$ (Li et al., 2015). It is important for all robots to stay connected so the swarm is cooperative. Nevertheless, the swarm may lost communication while performing the search and rescue task.

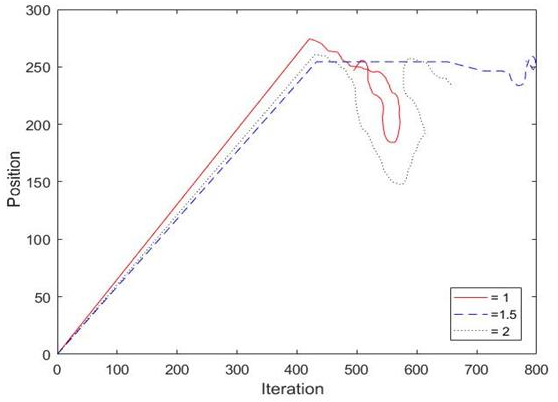


**Figure 2.** Analysis of swarm trajectory parameter μ

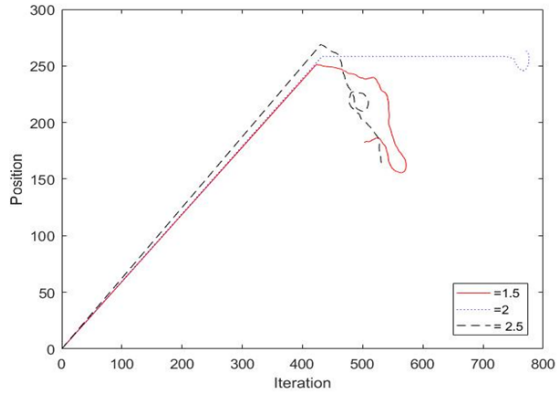


**Figure 3.** Analysis of swarm convergence parameter $\alpha_1$

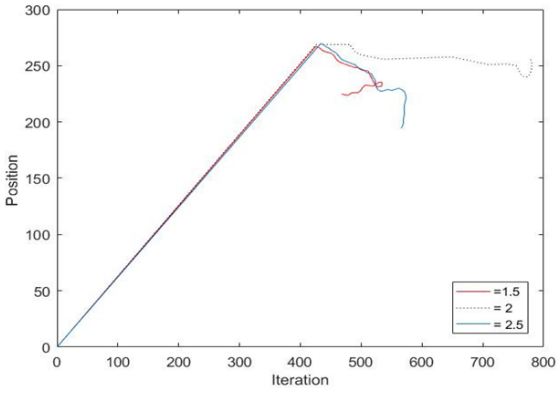


**Figure 4.** Analysis of swarm obstacle avoidance parameter $\alpha_2$

Losing communication means a particular robot has gone out of its communication range. The robot is not lost per se and can regain communication if it is able to regain the range and pick up signals from neighbouring robots. In Figure 5, when $\alpha_3$=2.5, the swarm lost communication before reaching the optimal solution. It gets close to the solution but cannot reach it. When $\alpha_3$=1.5, the swarm has unstable trajectory and gets stuck in sub-optimal solution. When $\alpha_3$=2 (260 positions in 780 iterations), the swarm moves directly towards the optimal solution with all the robots maintaining positive connectivity between them. Since the success of a search and rescue simulation depends highly on the cooperative behavior of the swarm, the value $\alpha_3$=2 is selected.

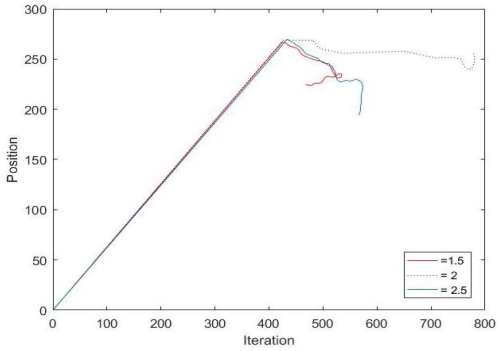


**Figure 5. Analysis of swarm convergence parameter $\alpha_3$**

### *4.2 Environment setup*

The aim of the experiment is to measure the performance of the proposed QRDPSO in comparison to its predecessor, the RDPSO in terms of a swarm's cooperation in searching optimal solution while performing obstacle avoidance and maintaining robot connectivity. In particular, we observe the convergence time and number of robot lost for a search and rescue task with the following scopes for the experimental setup:

- The obstacles' locations are unknown for the robots and are randomly spread in the environment
- The shape and the occupied area of each obstacle are random and vary between the obstacles
- There is only one target in the environment
- The location of the target is unknown to the robots

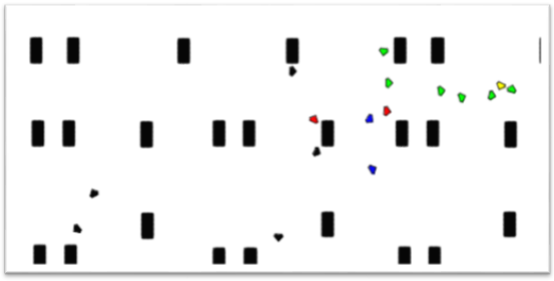

**Figure 6. A 300x300m environment used in experiment with random obstacles and a single target (victim)**

Figure 6 shows the environmental setup used for the experiment. In Figure 6, the rectangular blocks represent random obstacles generated. The triangular markers represent robots. The triangular-shaped in yellow represents the victim at a random location unknown to the robots. The green triangles denote robots which successfully located the victim and is proceeding towards it. In blue and red, the robots are facing some trouble navigating around obstacles with potential to get stuck in local optima. In this example, several black triangles are depicted to be far away from other robots (on their own at random positions respectively). These are the robots that have lost communication with the swarm and is moving randomly in the hope to regain the communication range with the swarm. These robots may get back on track towards victim if they are able to receive signals from other robots.

## 5 RESULTS

WE present the experiment results in this section. We ran both the RDPSO and the QRDPSO algorithms following the environment setup described in section 4.2 and experiment with a group of 5, 10, 15 and 20 robots in a swarm. Figure 7 shows the results when the QRDPSO is compared with the RDPSO in terms of speed of convergence in searching the optimal solution (victim). The chart shows that when the number of population of robots increases (i.e. 5, 10, 15 and 20), the time needed to find the optimal solution while maintaining connectivity and performing obstacles avoidance decreases for both QRDPSO and RDPSO. However, the robot in the QRDPSO are faster than the robots in the RDPSO. For example, in the QRDPSO, 5 robots can reach the optimal solution in 319 iterations but the RDPSO requires 418 iterations to reach the victim. The trend is similar with 10, 15 and 20 robots for the QRDPSO. This means the QRDPSO swarm rescues the victim faster and consumes less energy in doing so.

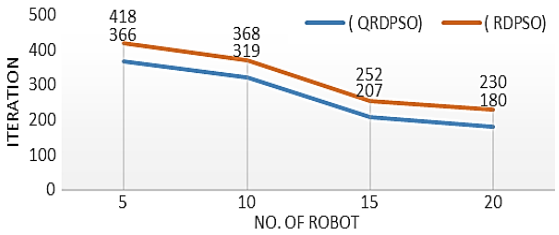


**Figure 7. Comparison between the QRDPSO and RDPSO convergence performance**

Figure 8 shows the number of robot lost after the QRDPSO and the RDPSO swarms completed their task. When the number of robot population increases (i.e. 5, 10, 15 and 20), the number of robot lost increases but the time needed to find the optimal solution decreases. When 5 robots are deployed, the QRDPSO completed the task in 366 iterations with 2 lost robots but the RDPSO lost 2 robots in 418 iterations. When 10 robots are deployed, the QDPSO lost 2 robots in 319 iterations but the RDPSO lost 4 robots in 368 iterations. When 15 robots are deployed, the QRDPSO lost 4 robots in 207 iterations but the RDPSO lost 6 in 252 iterations. Similarly when 20 robots are deployed, the QRDPSO lost 6 robots in 180 iterations but the RDPSO lost 8 robots in 230 iterations. Robots are lost because they are outside of the communication range (not connected to other robots). This is due to the performance of the objective function, in which no improvement to the minimize cost function is observed for both algorithms. However, if the lost robots do not get trapped in local optima and continue searching, they could somehow regain communication and may be able to regroup and reach optimal solution.

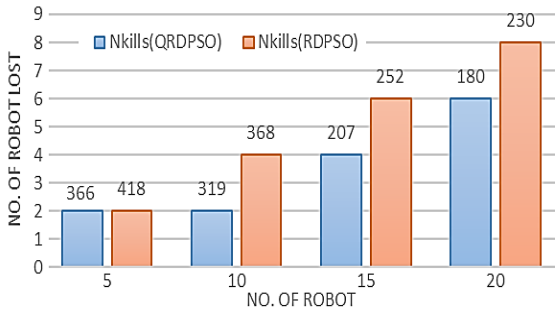


**Figure 8. Comparison between QRDPSO and RDPSO robot lost performance**

## 6 CONCLUSION

THIS paper shows how the QPSO, a quantum-based particle behaving algorithm is adopted onto the robot-based algorithm the RDPSO, to produce a new PSO derivation for swarm robotic application, the QRDPSO. In this paper, details on the algorithm formulation in particular a new cost or fitness function, and parameter conditioning are included. The proposed cost or fitness function aims to guide the robot swarm to overcome communication constraints and avoid getting trapped at obstacles (local optima) so the swarm may reach the victim (global best) in search and rescue simulations.

The paper then compared the QDPSO and the RDPSO algorithms on a MATLAB simulator. Following similar setup for both algorithms, the experiment showed that the QRDPSO model has a linear convergence of the whole population (robots) when reaching the global best solution and showing lesser number of robot lost in comparison to the RDPSO. In addition, the QRDPSO performs better in both speed and energy consumption in comparison to the RDPSO.

Communication is important for the swarm to maintain cooperation and we report improvement in terms of connectivity among individual robots in the QRDPSO swarm over the RDPSO. Nevertheless, there is still much to explore in regards to enhancing the QRDPSO swarm communication for robot energy conservation and prolonged lifetime during search and rescue exploration. One particular approach is to look into identifying partitions in the wireless sensor network for a more coordinated swarm movement (Cheng et al., 2015).

## 7 ACKNOWLEDGMENT

THIS work is fully supported by the Geran Bantuan Kecil Penyelidikan from the University of Malaya, Malaysia with grant number BKP058-2014.

## 8 REFERENCES

Baghaei, K. R., & Agah, A. (2003). Task allocation and communication methodologies for multi-robot systems. *Intelligent Automation & Soft Computing, 9*(4), 217-226.

Cai, Y., Chen, Z., & Min, H. (2013, October). Improving particle swarm optimization algorithm for distributed sensing and search. In *P2P, Parallel, Grid, Cloud and Internet Computing (3PGCIC), 2013 Eighth International Conference on* (pp. 373-379). IEEE.

Cecconi, F., & Campenní, M. (2010). PSO (particle swarm optimization): One method, many possible applications. *Agent-Based Evolutionary Search*, 229-254.

Cheng, L., Wang, Y., Wu, C., & Han, Q. (2015). A Pso-Based Maintenance Strategy in Wireless Sensor Networks. *Intelligent Automation & Soft Computing, 21*(1), 65-75.

Couceiro, M. S., Martins, F. M., Rocha, R. P., & Ferreira, N. M. (2014). Mechanism and convergence analysis of a multi-robot swarm approach based on natural selection. *Journal of Intelligent & Robotic Systems, 76*(2), 353-381.

Couceiro, M. S., Rocha, R. P., Ferreira, N. M., & Vargas, P. A. (2013, June). Darwinian robotic

swarms for exploration with minimal communication. In *Evolutionary Computation (CEC), 2013 IEEE Congress on* (pp. 127-134). IEEE.

Dadgar, M., Jafari, S., & Hamzeh, A. (2016). A PSO-based multi-robot cooperation method for target searching in unknown environments. *Neurocomputing*, *177*, 62-74.

Eberhart, R., & Kennedy, J. (1995, October). A new optimizer using particle swarm theory. In *Micro Machine and Human Science, 1995. MHS'95., Proceedings of the Sixth International Symposium on* (pp. 39-43). IEEE.

Kumar, A. S., Manikutty, G., Bhavani, R. R., & Couceiro, M. S. (2017, September). Search and rescue operations using robotic darwinian particle swarm optimization. In *Advances in Computing, Communications and Informatics (ICACCI), 2017 International Conference on* (pp. 1839-1843). IEEE.

Li, B., Li, D., Zhang, Z., Yang, S., & Wang, F. (2015). Slope stability analysis based on quantum-behaved particle swarm optimization and least squares support vector machine. *Applied Mathematical Modelling*, *39*(17), 5253-5264.

Nakisa, B., Rastgoo, M. N., Nasrudin, M. F., & Nazri, M. Z. A. (2015). A multi-swarm particle swarm optimization with local search on multi-robot search system. *Journal of Theoretical and Applied Information Technology*, *71*(1), 129-136.

Sànchez, N. D. G., Vargas, P. A., & Couceiro, M. S. (2018, July). A Darwinian Swarm Robotics Strategy Applied to Underwater Exploration. In *2018 IEEE Congress on Evolutionary Computation (CEC)* (pp. 1-6). IEEE.

Sun, J., Feng, B., & Xu, W. (2004, June). Particle swarm optimization with particles having quantum behavior. In *Evolutionary Computation, 2004. CEC2004. Congress on* (Vol. 1, pp. 325-331). IEEE.

Sun, J., Lai, C. H., & Wu, X. J. (2011). *Particle swarm optimisation: classical and quantum perspectives*. CRC Press.

Sun, J., Xu, W., & Feng, B. (2004, December). A global search strategy of quantum-behaved particle swarm optimization. In *Cybernetics and Intelligent Systems, 2004 IEEE Conference on* (Vol. 1, pp. 111-116). IEEE.

Tokgo, M., & Li, R. (2014, September). Estimation method for path planning parameter based on a modified QPSO algorithm. In *International Conference on Artificial Intelligence: Methodology, Systems, and Applications* (pp. 261-269). Springer, Cham.

Yuan, X., Wang, P., Yuan, Y., Huang, Y., & Zhang, X. (2015). A new quantum inspired chaotic artificial bee colony algorithm for optimal power flow problem. *Energy conversion and management*, *100*, 1-9.

Zuo, T., Min, H., Tang, Q., & Tao, Q. (2018). A Robot SLAM Improved by Quantum-Behaved Particles Swarm Optimization. *Mathematical Problems in Engineering*, 2018.

## 9 DISCLOSURE STATEMENT

NO potential conflict of interest was reported by the authors.

## 10 NOTES ON CONTRIBUTORS

**Duaa Abdel Fattah Mehiar** is a Ph.D. candidate at the Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, Malaysia. Her research interest includes simulation of multi-robot cooperative system for exploration. Ms. Mehiar receives her M.A. in Computer Science from the Al-Balqa Applied University, Jordan.

**Zati Hakim Azizul** is a lecturer at the Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, Malaysia. She focuses mainly on intelligent robotics testing cognitive strategies of navigation and mapping on autonomous robots. She received her Ph.D. from Auckland University of Technology, New Zealand.

**Loo Chu Kiong** is a professor at the Department of Artificial Intelligence, Faculty of Computer Science and Information Technology, University of Malaya, Malaysia. He focuses generally on soft computing such as quantum-inspired soft computing and intelligent robots. He received his Ph.D. in Computer Science from the University of Science, Malaysia.