Quantum-behaved Particle Swarm Optimization with Qubit's Bloch Coordinates Coding

Fuhua Shang
School of Computer & Information Technology
Northeast Petroleum University
Daqing 163318, China
Email: shangfh {at} 163.com

Abstract—**To enhance the optimization ability of quantum-behaved particle swarm optimization algorithm, some improvement measures are proposed. First, we propose a encoding approach based on qubits described on Bloch sphere. This approach makes each particle contain three groups of Bloch coordinates of qubits, and all three groups of coordinates are regarded as approximate solutions describing optimization result. Then we propose an updating approach of particles based on quantum computing. As the optimization process is performed in *n*-dimensional hypercube space [-1,1]^n, which has nothing to do with the specific issues, hence, the proposed approach has good adaptability for a variety of optimization problems. The experimental results show that the proposed algorithm is superior to the original one in optimization ability.**

Keywords- Quantum computation; Bloch coordinates; Particle swarm optimization; Algorithm design

I. INTRODUCTION

Particle swarm optimization (PSO) is a new global optimization algorithm proposed by Eberhart and Kennedy in 1995 [1]. As an important optimization tool, PSO has been successfully applied to combinatorial optimization [2] and numerical optimization [3]. For PSO performance, the following improvement strategies have been addressed such as the modification of design parameters [4-6], the modification of update rule of the particle location and velocity [7-8], the integration with other algorithms [9-12], and the quantum PSO (QPSO) based on quantum mechanics [13-16]. These improvements enhanced the PSO's performance in varying degrees. Quantum computing is an emerging computing technology, its integration with intelligent computing has broad application prospects. At present, QPSO has obtained some successful applications, however its theoretical studies is relatively few. The basic principles of QPSO is that the search mechanism is established by simulating that particles in the potential field always move towards the point with the lowest potential energy. Namely, the optimization space is regarded as the potential field (potential well) in quantum mechanics, the global optimal solution is regarded as the point with the lowest potential energy (potential well center), and the optimization process is regarded as the particles' moving to potential well center. This method has some advantages. However, in QPSO, the particles are encoded by real number, and for each variable, the search range is an interval on a number axis, which the search efficiency is not ideal.

This work was supported by the National Natural Science Foundation of China (Grant No. 61170132).

In order to enhance the optimization ability of QPSO, with help of the integration of quantum computation, we propose an improved quantum-behaved particle swarm optimization algorithm. In proposed algorithm, all particles are encoded by the coordinates of qubits described on the Bloch sphere. Since each qubit has three coordinate values, each particle has three locations, and each of locations represents an optimization solution, which can expand the search scope of each variable from an interval on the number axis to an area of the Bloch sphere, and can accelerate search process.

The remainder of the paper is structured as follows. Section 2 gives a brief survey on QPSO. Section 3 detailedly describes the QPSO improvement based on quantum computing. The implementation scheme of proposed approach is presented in Section 4. In Section 5, we test our algorithm through 4 benchmark problems. In addition, the experiment results are compared with the QPSO and the classical PSO. Section 6, is devoted to conclusions and future work.

II. THE QPSO MODEL

A. The PSO Model

Suppose the PSO consist of M particles in n dimension space, where the i^{th} particel's location X_i, velocity V_i, the own best location P_i^L, the global best location P_g are described as follows: $X_i = (x_{i1}, x_{i2}, \cdots, x_{in})$, $V_i = (v_{i1}, v_{i2}, \cdots, v_{in})$, $P_i^L = (p_{i1}^L, p_{i2}^L, \cdots, p_{in}^L)$, $P_g = (p_{g1}, p_{g2}, \cdots, p_{gn})$. The update strategy can be written as

$$V_i(t+1) = wV_i(t) + c_1 r_1 (P_i^L - X_i(t)) + c_2 r_2 (P_g - X_i(t)) \quad (1)$$

$$X_i(t+1) = X_i(t) + V_i(t+1) \quad (2)$$

where $i = 1,2,\cdots,m$, w denote the inertia factor, c_1 denotes the self factor, c_2 denote the global factor, r_1 and r_2 are two uniformly distributed random numbers in (0,1).

For ease of description, the Eq.(1) is rewritten as follows

$$V_i(t+1) = wV_i(t) + [\Phi](P_i - X_i(t)) \tag{3}$$

where P_i and $[\Phi]$ are defined as

$$\begin{aligned} P_i = &\text{diag}\left(\frac{c_1 r_1^1}{c_1 r_1^1 + c_2 r_1^2}, \cdots\cdots, \frac{c_1 r_n^1}{c_1 r_n^1 + c_2 r_n^2}\right) P_i^L \\ &+ \text{diag}\left(\frac{c_2 r_1^2}{c_1 r_1^1 + c_2 r_1^2}, \cdots\cdots, \frac{c_2 r_n^2}{c_1 r_n^1 + c_2 r_n^2}\right) P_g \end{aligned} \tag{4}$$

$$[\Phi] = \text{diag}(c_1 r_1^1 + c_2 r_1^2, \cdots\cdots, c_1 r_n^1 + c_2 r_n^2) \tag{5}$$

[17] pointed out that, in order to ensure PSO convergence, all particles must approximate P_i defined by Eq.(4).

B. The QPSO Model

In quantum mechanics, particle dynamic behavior comply with the following Schrodinger equation.

$$j\hbar \frac{\partial}{\partial t} \Psi(r,t) = (-\frac{\hbar^2}{2m} \nabla^2 + V(r)) \Psi(r,t) \tag{6}$$

where $\hbar$ denotes Planck's constant, m denotes particle quality, $V(r)$denotes energy distribution function.

In Schrodinger equation, the unknown is the wave function $\Psi(r,t)$. According to the statistical interpretation of wave function, the square of the magnitude of this function denotes the probability density. Taking the *delta* potential well for example, the design of QPSO is described as follows.

The potential energy distribution function of *delta* potential well can be expressed as

$$V(r) = -\gamma \delta(r) \tag{7}$$

where γ denote the potential well depth.

Substituting Eq.(7) into Eq.(6), we can obtain particle's wave function as follows

$$\Psi(r) = \frac{1}{\sqrt{L}} e^{-|r|/L} \tag{8}$$

where $L = \frac{\hbar^2}{m\gamma}$ denotes the characteristic length of *delta* potential well.

Therefore, particle's probability density function can be written as

$$Q(r) = |\Psi(r)|^2 = \frac{1}{L} e^{-2|r|/L} \tag{9}$$

To increase the particles' probability of moving towards potential well centre, Eq.(9) must satisfy the following relationship

$$\int_{-|r|}^{|r|} Q(r) dr > 0.5 \tag{10}$$

From Eqs.(9,10), the characteristic length L must meet the following equation

$$L = \frac{|r|}{g \ln(\sqrt{2})} \tag{11}$$

where g >1.

In the potential well, the dynamic behavior of the particles obeys the Schrodinger equation, in which the particles' locations are random at any time. However, the particles in ordinary PSO obey Newtonian mechanics, where the particles must have definite locations at any time. This contradiction can be satisfactorily resolved by means of the collapse of the wave function and Monte Carlo method. We first take random number u from the range of (0,1), and then let $u = e^{-2|r|/L}$, and finally obtain the following results

$$|r| = \frac{L}{2} \ln(\frac{1}{u}) \tag{12}$$

According to Eqs.(11,12), it can be derived that $|r_{k+1}| = \frac{\ln(1/u)}{2g \ln\sqrt{2}} |r_k|$. Let $r_k = x_k - P_k$, using some algebra it is possible to derive $x_{k+1} = P_{k+1} \pm \frac{\ln(1/u)}{g \ln 2} |x_k - P_k|$. Let $\alpha = \frac{1}{2g \ln\sqrt{2}}$, the following result may be obtained by

$$x_{k+1} = P_{k+1} \pm \alpha |x_k - P_k| \ln(\frac{1}{u}) \tag{13}$$

The above equation is the iterative equation QPSO [16].

III. THE QPSO BASED ON BLOCH COORDINATES OF QUBITS

In this section, we propose a Bloch Coordinates-based quantum-behaved particle swarm optimization algorithm, called BQPSO.

A. The Spherical Description of Qubits

In quantum computing, a qubit is a two-level quantum system, described by a two-dimensional complex Hilbert space. From the superposition principles, any state of the qubit may be written as

$$|\varphi\rangle = \cos(\theta/2)|0\rangle + e^{i\phi}\sin(\theta/2)|1\rangle \tag{14}$$

where $0 \le \theta \le \pi$, $0 \le \varphi \le 2\pi$.

Therefore, unlike the classical bit, which can only be set equal to 0 or 1, the qubit resides in a vector space parameterized by the continuous variables θ and ϕ. Thus, a continuous of states is allowed. The Bloch sphere representation is useful in thinking about qubits since it provides a geometric picture of the qubit and of the transformations that one can operate on the state of a qubit. Owing to the normalization condition, the qubit's state can be represented by a point on a sphere of unit radius, called the Bloch sphere. This sphere can be embedded in a three-dimensional space of Cartesian coordinates ($x = \cos\phi\sin\theta$, $y = \sin\phi\sin\theta$, $z = \cos\theta$). By definition, a Bloch vector is a vector whose components (*x*, *y*, *z*) single out a point on the Bloch sphere. We can say that the angles θ and ϕ define a Bloch vector, as shown in Fig.1.

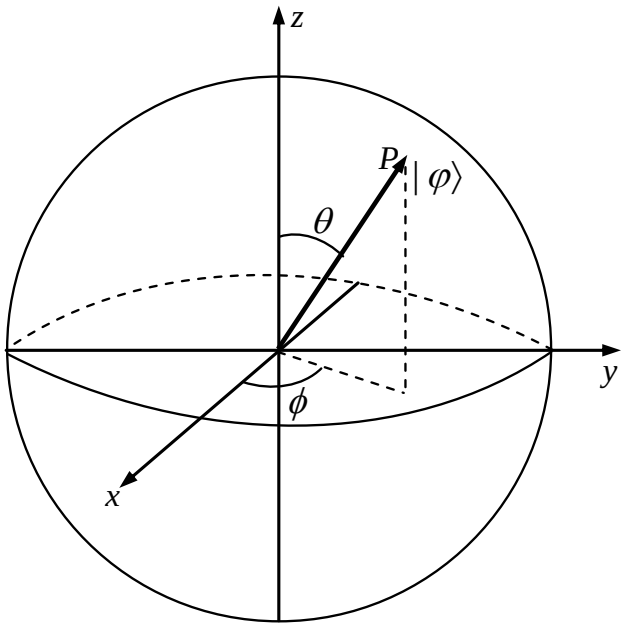

Figure 1. A qubit description on Bloch sphere.

B. The BQPSO Encoding Method

In BQPSO, all particles are encoded by qubits described on Bloch sphere. Set the swarm size to *m*, and the space dimension to *n*. Then the i^{th} particle is encoded as

$$p_i = \begin{vmatrix} \cos\phi_{i1}\sin\theta_{i1} & \cdots & \cos\phi_{in}\sin\theta_{in} \\ \sin\phi_{i1}\sin\theta_{i1} & \cdots & \sin\phi_{in}\sin\theta_{in} \\ \cos\theta_{i1} & \cdots & \cos\theta_{in} \end{vmatrix} \tag{15}$$

where $0 \le \theta_{ij} \le \pi$, $0 \le \phi_{ij} \le 2\pi$, $i = 1,2,\cdots,m$, $j = 1,2,\cdots,n$

As the optimization is performed in $[-1,1]^n$, which has nothing to do with the specific issues, hence, the proposed method has good adaptability for a variety of optimization problems.

In BQPSO, the Bloch coordinates of each qubit are regarded as three paratactic location components, each particle contains three paratactic locations, and each of locations represents an optimization solution. Therefore, in unit space $[-1,1]^n$, each particle simultaneously represents three optimization solutions, which can be described as follows

$$\begin{cases} \boldsymbol{p}_{ix} = [\cos\phi_{i1}\sin\theta_{i1}, \cdots, \cos\phi_{in}\sin\theta_{in}] \\ \boldsymbol{p}_{iy} = [\sin\phi_{i1}\sin\theta_{i1}, \cdots, \sin\phi_{in}\sin\theta_{in}] \\ \boldsymbol{p}_{iz} = [\cos\theta_{i1}, \cos\theta_{i2}, \cdots, \cos\theta_{in}] \end{cases} \tag{16}$$

where $\boldsymbol{p}_{ix}$ is known as *X* solution, $\boldsymbol{p}_{iy}$ is known as *Y* solution, and $\boldsymbol{p}_{iz}$ is known as *Z* solution. The particle encoded by qubits is shown in Fig.2.

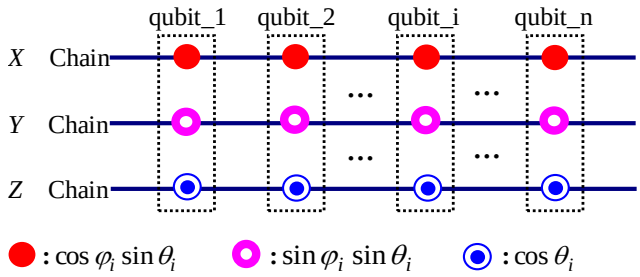

Figure 2. The structure of encoding particle.

Such encoding method as Eq.(15) has some advantages as follows. Since three solutions are synchronously updated in each optimization step, for same colony size, such encoding method can extend search range, accelerate optimization process, and increase the number of global optimum solutions and the success probability.

C. Solution Space Transformation

In BQPSO, each particle contains 3*n* Bloch coordinates of *n* qubits that can be transformed from unit space $[-1,1]^n$ to solution space of the continuous optimization problem. Each of Bloch coordinates corresponds to an optimization variable in solution space. Let the j^{th} variable of optimization problem

$X_j \in [A_j, B_j]$, and (x_{ij}, y_{ij}, z_{ij}) denote the coordinates of the j^{th} qubit on the i^{th} particle. Then the corresponding variables (X_{ij}, Y_{ij}, Z_{ij}) in solution space are respectively computed as follows

$$X_{ij} = \frac{1}{2}[A_j(1-x_{ij}) + B_j(1+x_{ij})] \quad (17)$$

$$Y_{ij} = \frac{1}{2}[A_j(1-y_{ij}) + B_j(1+y_{ij})] \quad (18)$$

$$Z_{ij} = \frac{1}{2}[A_j(1-z_{ij}) + B_j(1+z_{ij})] \quad (19)$$

where $i = 1,2,\cdots,m$, $j = 1,2,\cdots,n$.

D. The Optimal Solutions Update

Substituting three solutions $[X_{i1}, X_{i2},\cdots, X_{in}]$, $[Y_{i1}, Y_{i2},\cdots, Y_{in}]$, $[Z_{i1}, Z_{i2},\cdots, Z_{in}]$ described by the i^{th} particle into the fitness function, respectively, we may compute its fitness, where $i = 1, 2,\cdots, m$. Let $gfit_{best}$ denote the best fitness so far, and gp_{best} denote the corresponding best particle, $cfit_i$ denote the own best fitness of the i^{th} particle, and cp_i denote the corresponding best particle, $fit(p_i) = \max(fit(X_i), fit(Y_i), fit(Z_i))$, $fit_{best} = \max\limits_{1\le i\le m} fit(p_i)$. If $cfit_i < fit(p_i)$ then $cfit_i = fit(p_i)$ and $cp_i = p_i$. If $gfit_{best} < fit_{best}$ then $gfit_{best} = fit_{best}$ and $gp_{best} = p_{best}$.

E. Particle Locations Update

For the i^{th} particle, let P_{ij} denote the current location of the j^{th} qubit $|\varphi_{ij}\rangle$ on the Bloch sphere, P_{ij}^L and P_{gj} denote its own best location and global best location on the Bloch sphere. According to Ref.[16], for P_{ij}, the two potential well centers in Eq.(13) can be respectively obtained by

$$P_{ij}^c(k) = \frac{\sum_{i=1}^m P_{ij}^L / m}{\| \sum_{i=1}^m P_{ij}^L / m \|} \quad (20)$$

$$P_{ij}^c(k+1) = \frac{rP_{ij}^L + (1-r)P_{gj}}{\| rP_{ij}^L + (1-r)P_{gj} \|} \quad (21)$$

where m is the number of particles, r is a random number uniformly distributed in (0,1), and k denote the iterative steps.

Let O denote the centre of Bloch sphere, $\beta_{ij}(k)$ denote the angle between $\overrightarrow{OP_{ij}}$ and $\overrightarrow{OP_{ij}^c(k)}$. From the QPSO's iteration equation, in order to make P_{ij} move to $P_{ij}^c(k+1)$ \$P_{ij}^{c}(k+1)\$, such an angle $\beta_{ij}(k+1)$ need to be rotated on the Bloch sphere that

$$\beta_{ij}(k+1) = \pm\alpha \ln(1/u)|\beta_{ij}(k)|, \ \alpha < 1 \quad (22)$$

Let the qubit corresponding to point $P_{ij}^c(k)$ is $|\varphi_{ij}^c\rangle$, from the above equation, we know that, the new location of $|\varphi_{ij}\rangle$ is actually the location of $|\varphi_{ij}^c\rangle$ after it is rotated through an angle $\beta_{ij}(k+1)$ towards $|\varphi_{ij}\rangle$.

Let $|\varphi_{ij}^c\rangle = \cos(\theta_{ij}^c/2)|0\rangle + e^{i\phi_{ij}^c/2}\sin(\theta_{ij}^c/2)|1\rangle$ and $|\varphi_{ij}\rangle = \cos(\theta_{ij}/2)|0\rangle + e^{i\phi_{ij}/2}\sin(\theta_{ij}/2)|1\rangle$. The location parameters θ_{ij} and ϕ_{ij} of particle p_i can be updated as follows

$$\begin{cases} \theta_{ij} = \theta_{ij}^c + \beta_{ij}(k+1) \\ \phi_{ij} = \phi_{ij}^c + \beta_{ij}(k+1) \end{cases} \quad (23)$$

F. Particle Location Mutation

In order to increase individual's diversity and prevent premature convergence, a variety of evolutionary algorithms introduce mutation. Most of the current quantum optimization algorithm use the quantum NOT gate (namely, Pauli matrix σ_x) to perform the mutation, in which two probability amplitudes of a qubit are exchanged, and only a qubit's parameter θ is changed. In BQPSO, we propose a new mutation based on the *Hadamard* gate that is defined as follows.

$$\boldsymbol{H} = \frac{1}{\sqrt{2}}\begin{bmatrix} 1 & 1 \\ 1 & -1 \end{bmatrix} \quad (24)$$

This gate is an important unitary operator in quantum computing, it can be written as the linear combinations of two Pauli matrices, and has the following property.

$$-i\boldsymbol{H} = \cos\frac{\pi}{2}\boldsymbol{I} - i\sin\frac{\pi}{2}\left(\frac{\boldsymbol{\sigma}_x}{\sqrt{2}} + 0\boldsymbol{\sigma}_y + \frac{\boldsymbol{\sigma}_z}{\sqrt{2}}\right) \quad (25)$$

From this equation it is clear that the *Hadamard* gate is a rotation through an angle $\delta = \pi$ about the axis $\vec{n} = [1/\sqrt{2},\ 0,\ 1/\sqrt{2}]$. Indeed, the above equation coincides $R_{\vec{n}}(\pi)$, up to an overall phase. This

transformation rotates the *x*-axis to *z*-axis and vice versa. As this rotation have nothing to do with the potential well center, and the rotation angle is great, it can avoid premature convergence and increase diversity of particles. For each particle in swarm, first generate a random number, if this number is less than the mutation probability, then randomly select a qubit of this particle, and apply *Hadamard* gate to perform its mutation.

IV. THE IMPLEMENTATION SCHEME OF BQPSO

Step1 Particle swarm initialization. Include: swarm size *m*, space dimension *n*, mutation probability p_m, iterative steps *G*, control parameter α , According to Eq.(15), generate the initial swarm, and set the current iterative step $t = 0$.

Step2 Perform solution space transformation according to Eqs.(17-19), compute each particle's fitness, and update the own best locations and the global best location.

Step3 For each quit in each particle, compute the potential well center by Eqs.(20,21), the rotation angle by Eq.(22), and perform the rotation by Eq.(23). Mutate particles by *Hadamard* gates according to mutation probability.

Step4 Set $t = t+1$, if $t > G$ then save the optimization results and stop, else go back to **step2**.

V. EXPERIMENTAL RESULTS AND ANALYSIS

A. Function Extremum Optimization

Many benchmark numerical functions are commonly used to evaluate and compare optimization algorithms. If a function has more than one local optimum, this function is called multimodal. Multimodal functions are used to test the ability of algorithms getting rid of local minima. If the exploration process of an algorithm is poor that it cannot search whole space efficiently, this algorithm gets stuck at the local minima. Another group of test problems is separable/non-separable functions. A p-variable separable function can be expressed as the sum of p functions of one variable. Non-separable functions cannot be written in this form. These functions have interrelation among their variables. Therefore, non-separable functions are more difficult than the separable functions.

In this section, the performance of proposed BQPSO is evaluated on 4 standard, unconstrained, multimodal, non-separable benchmark functions with different characteristics. Definition, search ranges, and optimum values of the test functions are given as follows.

(1) Shaffers F5 function

$$f(x) = \frac{1}{500} + \sum_{j=1}^{25} \frac{1}{j + \sum_{i=1}^{2} (x_i - a_{ij})^6} \tag{26}$$

where $x_i \in [-65.536, 65.536]$

$$(a_{ij}^k) = \begin{pmatrix} -32 & -16 & 0 & 16 & 32 \\ -32+16k & -32+16k & -32+16k & -32+16k & -32+16k \end{pmatrix}$$

$(a_{ij}) = (a_{ij}^0 \;\; a_{ij}^1 \;\; a_{ij}^2 \;\; a_{ij}^3 \;\; a_{ij}^4)$, $i = 1,2$, $j = 1,2,\cdots,25$, $k = 0,1,\cdots,4$.

This function has multiple local maximum points, and the global maximum point is (-32,-32), the global maximum is 1.002. When the optimization result is greater than 1.000, the algorithm is considered convergence.

(2) Shubert function

$$f(x,y) = \{\sum_{i=1}^{5} i\cos[(i+1)x + i]\} \times \{\sum_{i=1}^{5} i\cos[(i+1)y + i]\} \tag{27}$$

where $x, y \in [-10, 10]$. This function has 760 local minimum points, and the global minimum is -186.73090882259. This function can easily fall into local minimum -186.34. When the optimization result is less than -186.34, the algorithm is considered convergence.

(3) Branin function

$$f(x,y) = (x - \frac{5.1}{4\pi^2} y^2 + \frac{5}{\pi} y - 6)^2 + 10(1 - \frac{1}{8\pi})\cos y + 10 \tag{28}$$

where $x \in [0, 15]$, $y \in [-5, 10]$.

This function has a global minimum 0.3979, and a local minimum 0.4004 that is very close to the global minimum. When the optimization result is less than 0.4000, the algorithm is considered convergence.

(4) Camel function

$$f(x,y) = (4 - 4x^2)x^2 - xy - (4 - 2.1y^2 + \frac{y^4}{3})y^2 \tag{29}$$

where $x, y \in [-10, 10]$.

This function has a global maximum 1.031628. When the optimization result is greater than 1.000, the algorithm is considered convergence.

For the above four functions, we use respectively BQPSO, QPSO, PSO to perform optimization. In order to reflect the fairness of comparing results, three algorithms use the same colony size and iterative steps. These parameters are set as:

colony size m=20, iterative steps G=100, control parameter $\alpha = 0.8$, mutation probability $p_m = 0.001$.

In order to manifest the objectivity of the comparison results, for the above four functions, we use each algorithm to optimize 1000 times, and record convergence times, average iterative steps, average results, and variance of results. For the case of convergence, we also record the times that converges to X solutions, Y solution, and Z solutions, respectively. The experimental results are shown in Tables 1-4.

TABLE I. THE OPTIMIZATION RESULT CONTRASTS OF SHAFFER'S F5.

Alg.	*Con. times*	*X*	*Y*	*Z*	*Avg. of step*	*Avg. of result*	*Var. of result*
BQPSO	784	311	280	193	66.1220	0.8970	0.0502
QPSO	723	—	—	—	68.7330	0.9044	0.0473
PSO	328	—	—	—	89.2410	0.7534	0.1024

TABLE II. THE OPTIMIZATION RESULT CONTRASTS OF SHUBERT.

Alg.	*Con. times*	*X*	*Y*	*Z*	*Avg. of step*	*Avg. of result*	*Var. of result*
BQPSO	944	343	310	291	61.6470	-186.48	0.1860
QPSO	543	—	—	—	76.5250	-186.07	0.7106
PSO	334	—	—	—	87.1210	-184.81	56.1972

TABLE III. THE OPTIMIZATION RESULT CONTRASTS OF BRANIN.

Alg.	*Con. times*	*X*	*Y*	*Z*	*Avg. of step*	*Avg. of result*	*Var. of result*
BQPSO	996	321	286	389	49.8420	0.3990	8.6e-07
QPSO	826	—	—	—	58.9750	0.3995	2.5e-06
PSO	489	—	—	—	80.0120	0.4018	7.8e-05

TABLE IV. THE OPTIMIZATION RESULT CONTRASTS OF CAMEL.

Alg.	*Con. times*	*X*	*Y*	*Z*	*Avg. of step*	*Avg. of result*	*Var. of result*
BQPSO	999	451	431	117	36.5520	1.0143	7.2e-04
QPSO	947	—	—	—	44.6050	1.0109	0.0021
PSO	658	—	—	—	75.8260	-8.5867	1.2e+04

It can be seen from tables 1-4 that, for the above four functions, three algorithms have the same sort in the optimization ability, from high to low in turn for BQPSO, QPSO, PSO.

B. Fuzzy Controller Parameters Optimization

In the fuzzy control system, the performance of the controller has a significant impact on the performance of system. Fuzzy controller performance to a large extent depends on the fuzzy control rules and its scalability. Therefore, we can introduce an adjustable parameter to adjust the control rules, so that the controlled object can obtain satisfactory control performance. This is the design problem of the fuzzy controller with a group of adjustable fuzzy rules. In this kind of fuzzy controller design, the control action depends on the error and error change. To adapt to different requirements of the controlled object, by introducing an adjustment factor, we may obtain a kind of fuzzy control rule with analytical description as follows

$$u = -\langle \alpha E + (1-\alpha)EC \rangle, \quad \alpha \in (0,1) \tag{30}$$

By adjusting the size of α, we can achieve varying degrees of weighting error and error change. When the error is large, the main control task is to eliminate error, at this time; we should increase the error's weighting. On the contrary, when the error is small, the main control task is to reduce the overshoot in order to stabilize the system as soon as possible. Therefore, the different error levels need to introduce different weighting factors in order to achieve self-adaptive adjusting of control rules. Taking the following second-order system for the controlled object, and using the step signal as input, we investigate the optimization ability of BQPSO.

$$F(s) = \frac{20}{(2s+1)(4s+1)} \tag{31}$$

The domain of error, error change, and control size is selected as $\{E\} = \{EC\} = \{u\} = \{-3, -2, -1, 0, 1, 2, 3\}$. Taking into account the fuzzy control system in different system states should have different requirements for the control parameter α, in this experiment, the α is divided into three levels.

$$u = \begin{cases} -\langle \alpha_1 E + (1-\alpha_1)EC \rangle & E = 0, \pm 1 \\ -\langle \alpha_2 E + (1-\alpha_2)EC \rangle & E = \pm 2 \\ -\langle \alpha_3 E + (1-\alpha_3)EC \rangle & E = \pm 3 \end{cases} \tag{32}$$

Therefore, this study need to optimize the six fuzzy controller parameters such as quantization factor k_e, k_{ec}, scale factor k_u, adjustment factor α_1, α_2, α_3. With help of the **ITAE** integral performance index, the evaluation function is designed as follows

$$f = \frac{1}{a + J(\mathrm{ITAE})} \tag{33}$$

where $J(\mathrm{ITAE}) = \int_0^\infty t \,|\, e(t) \,|\, dt$, a is a small positive number so that the denominator is not zero.

According to experience, the initial scopes of six controller parameters are given by $k_e, k_c, k_u \in (0, 10)$, $\alpha_1 \in (0, 0.4)$, $\alpha_2 \in (0.4, 0.8)$, $\alpha_3 \in (0.8, 1.0)$. These six parameters are respectively optimized by three algorithms. The colony size m=15, space dimension n=6, iterative steps G=50, the other setting is the same as the previous experiment. The

optimization results of three algorithms are shown in Table 5, the **ITAE** performance comparisons are shown in Fig.3, and the system response for step signal input under the control action of three fuzzy controllers are shown in Fig.4.

TABLE V. OPTIMIZATION RESULT CONTRASTS OF FUZZY CONTROLLER.

Alg.	k_e	k_{ec}	k_u	α_1	α_2	α_3	**J(ITAE)**
BQPSO	4.5036	2.7263	9.9703	0.2468	0.4769	0.9194	3.9503
QPSO	6.1607	5.6757	6.2983	0.0751	0.4728	0.8917	4.8329
PSO	4.2562	3.7794	4.1931	0.3712	0.5059	0.9967	5.4465

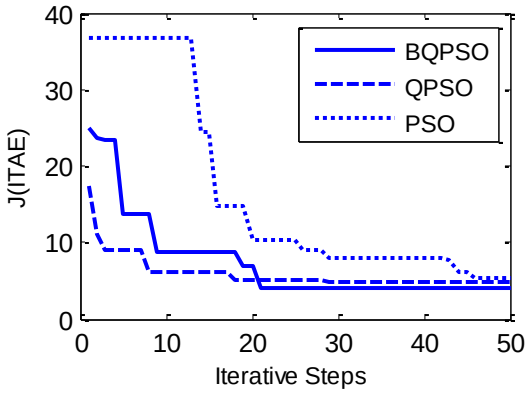

Figure 3. **ITAT** integral index comparison.

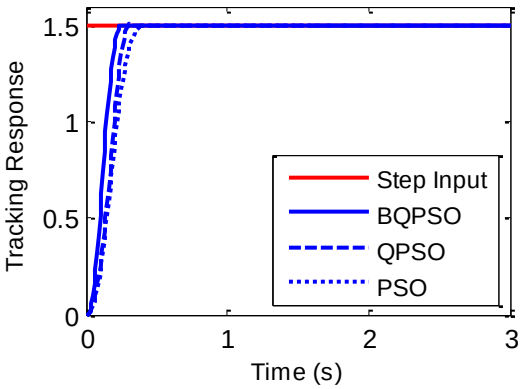

Figure 4. Tracking response curve comparisons of Fuzzy controllers.

Table 5 shows that, in the same colony size and iterative steps, the BQPSO's **ITAE** is the smallest, followed by QPSO and PSO. From Fig.4, it is clear that the BQPSO has obtained the minimum 3.9503 after about 20 iterative steps, while, the QPSO and PSO have respectively obtained 4.8329 and 5.4465 after 50 iterative steps. Fig.5 shows that, the controller optimized by BQPSO has a faster tracking speed and a less tracking time than that optimized by QPSO and PSO, which shows that BQPSO obtains more excellent controller parameters combination than QPSO and PSO, and makes the fuzzy controller have the excellent control performance. The experimental results show that the optimization ability of BQPSO indeed better than that of QPSO and PSO.

VI. CONCLUSIONS

This paper presents an improved quantum-behaved particle swarm optimization algorithm. The experimental results reveal that the encoding method based on coordinates of qubits can better simulate the quantum behavior, the search method based on qubits' rotating can improve search efficiency, and the integration of the following four aspects of the PSO, *delta* potential well, the encoding method based on Bloch coordinates of qubits, and the qubits' rotation on Bloch sphere, can really enhance the optimization ability of the classical PSO. The advantages and disadvantages of the Bloch coordinates-based encoding method and the integration of this method with other intelligent optimization algorithms are two issues which we need to study in future.

REFERENCES

[1] J. Kennedy , R. C. Eberhart, "Particle swarms optimization," in: Proceedings of IEEE international conference on Neural Networks, 1995, pp.1942-1948.

[2] W. Z. Guo, G. L. Chen, S. J. Peng, "Hybrid Particle Swarm Optimization Algorithm for VLSI Circuit Partitioning," Journal of Software, 2011, 22(5), pp.833-842.

[3] S. W. Lin, K. C. Ying, S. C. Chen, et al., "Particle swarm optimization for parameter determination and feature selection of support vector machines," Expert Systems with Applications, 2008, 35(4), pp.1817-1824.

[4] X. J. Cai, Z. H. Cui, J. C. Zeng, et al., "Dispersed particle swarm optimization," Information Processing Letters, 2008, 105(6), pp.231-235.

[5] F. Bergh, A. P. Engelbrecht, "A study of particle swarm optimization particle trajectories," Information Science, 2006, 176(8), pp.937-971.

[6] A. Chatterjee, P. Siarry, "Nonlinear inertia weight variation for dynamic adaptation in particle swarm optimization," Computers & operations research, 2007, 33(3), pp.859-871.

[7] Z. S. Lu, Z. R. Hou, "Particle swarm optimization with adaptive mutation," Acta Electronica Sinica, 2004, 32(3), pp. 416-420.

[8] Y. Liu, Z. Qin, Z. W. Shi, et al., "Center particle swarm optimization," Neurocomputing, 2007, 70(4-6), pp.672-679.

[9] B. Liu, L. Wang, Y. H. Jin, et al., "Improved particle swarm optimization combined with chaos," Chaos Solitons & Fractals, 2005, 25(5), pp.1261-1271.

[10] Q. Luo, D. Y. Yi, "A co-evolving framework for robust particle swarm optimization," Applied Mathematics and Computation, 2008, 199(2), pp.611-622.

[11] Y. J. Zhang, S. F. Shao, "Cloud mutation Particle Swarm Optimization Algorithm Based on Cloud Model," Pattern Recognition & Artificial Intelligence, 2011, 24(1), pp.90-95.

[12] H. M. Zhu, Y. P. Wu, "A PSO algorithm with high speed convergence," Control and Decision, 2010, 25(1), pp.20-24.

[13] J. Sun, B. Feng, W. B. Xu, "A Global Search Strategy of Quantum-Behaved Particle Swarm Optimization," IEEE Conference on Cybernetics and Intelligent Systems, 2004, 01, pp.111-116.

[14] J. Sun, W. B. Xu, W. Fang, "Quantum-behaved particle swarm optimization algorithm with controlled diversity,"International Conference on Computational Science, 2006, 03, pp.847-854.

[15] M. L. Xia, J. Sun, W. B. Xu, "An improved quantum-behaved particle swarm optimization algorithm with weighted mean best position," Applied Mathematics and Computation, 2008, 205(2), pp.751-759.

[16] W. Fang, J. Sun, Z. P. Xie, W. B. Xu, "Convergence analysis of quantum-behaved particle swarm optimization algorithm and

study on its control parameter," Acta Physica Sinica, 2010, 59(6), pp.3686-3693.

[17] M. Clerc, J. Kennedy, "The particle swarm: explosion, stability, and convergence in a multi-dimensional complex space," IEEE Transactions on Evolutionary Computation, 2002, 6(1), pp.58-73.