Potential and Dynamics-based Particle Swarm Optimization

Hyungmin Park and Jong-Hwan Kim

***Abstract*— The Particle Swarm Optimization (PSO) algorithm is a robust stochastic evolutionary computation technique based on the movement and intelligence of swarms. This paper proposes a novel PSO algorithm, based on the potential field and the motion dynamics model. It is assumed that particles form potential fields and each particle has its own mass. The potential filed and mass are modeled by the particles' fitness value. By using these fitness based models, the proposed algorithm performs well, in particular, in avoiding the local minima compare to the original PSO. The proposed PD-PSO successfully solves minimization problems of complex test functions.**

I. Introduction

THE Particle Swarm Optimization (PSO) algorithm, first proposed by J. Kennedy and R. Eberhart in 1995, is an optimization paradigm which simulates the ability of human society to process knowledge [1]. The algorithm is first designed to solve the neural network weight optimization problem, but it also works well with other optimization issues. The algorithm is used in many fields such as electromagnetic optimization problem [2], human tremor analysis [3], and so on. Like other stochastic optimization techniques such as evolutionary programming (EP), evolution strategies (ES), genetic algorithm (GA), the PSO algorithm is a stochastic search algorithm with no gradient information of objective function and its behavior can be easily controlled. Furthermore, the PSO algorithm and its variants are applicable to solve the multi-objective optimization problems [4] and constrained optimization problems [5] like Two-phase EP [6], Evolian [7], etc.

The fundamental principle behind the PSO algorithm is the evolutionary mechanism which leads to information sharing among individuals, named particles, through generation. Each particle is randomly deployed in an *n-dimensional* search space. The particles update their position in the search space according to a simple stochastic rule known as the *velocity rule*. In the original PSO algorithm, each particle looks for the global best particle in each generation and its personal best position throughout the generation. As a result, particles converge to the global best position like a bird flock or a fish flock as if they have swarm intelligence.

A number of variant algorithms of PSO have been suggested by updating the velocity rule. Hu and Eberhart presented a method to re-evaluate the global-best and the second-best before every iteration [8]. Carlisle and Dozier also introduced a similar algorithm by monitoring randomly "sentry" particle which was randomly chosen at each iteration [9]. Hu *et al.* updated the velocity rule according to the diversity of population [10]. Blackwell and Bentley modified the velocity rule by adding additional term based on the electrical force between two charged particles [11].

In this paper, phenomena in nature system are considered to update the velocity rule. Nature system tends to optimize itself into equilibrium state in which the environment has the minimum potential energy. Considering electric potential, i.e. voltage, charged particles move to the direction that minimizes the voltage difference between the particles. Also the movement of particles follows the Newton's motion dynamics law. It means that the mass and accelerated of particles should be considered to calculate the physical movement. By introducing these two natural phenomena to the concept of the original PSO algorithm, a novel algorithm, called potential and dynamics-based particle swarm optimization (PD-PSO) can be proposed. Proposed algorithm is based on potential field model and the motion dynamics of particle with mass. Thus, the driving force for the particles is formed by the potential field and the movement of particles follows motion dynamics rules. The effectiveness of the proposed algorithm is demonstrated by carrying out experiments for well known test functions. In particular, it is robust for the multi-modal functions which have many local minima.

The remainder of this paper is organized as follows. Section II describes problem formulation and terminology. In Section III, the novel PSO algorithm based on potential model and motion dynamics is proposed. Section IV presents experiment results on test functions such as De Jong functions and Griewangk function.

II. Problem Formulation and Terminology

A. Generic optimization problem

The generic optimization problem is defined as follows: find the solution $\mathbf{x}^*$ which gains the global minimum value $f(\mathbf{x}^*)$ of a given real-valued function $f : D \subseteq \mathbf{R}^n \rightarrow \mathbf{R}$, such that

$$f^* = f(\mathbf{x}^*) \leq f(\mathbf{x}), \quad \forall \mathbf{x} \in \mathbf{D} \tag{1}$$

where $\mathbf{D}$ is the allowable (bounded) *n-dimensional* search space.

B. Terminology

- A **particle** is a candidate solution defined in n-dimensional space and assumed to have its own mass. A particle p_i's location represents an input vector $\mathbf{x}_i$ for objective function f.
- A **population** is the set of all the particles. The size of population is *N*.

Authors are with EECS Department, KAIST, Guseong-dong, Yuseong-gu, Daejeon, 305-701, Republic of Korea (e-mails: {hmpark, johkim} @rit.kaist.ac.kr)

978-1-4244-1823-7/08/$25.00©2008 IEEE

Authorized licensed use limited to: Korea Advanced Inst of Science & Tech - KAIST. Downloaded on July 15,2021 at 14:24:40 UTC from IEEE Xplore. Restrictions apply.

- For a given minimization problem, a particle p_i is **better** than the other particle p_j if

$$f(\mathbf{x}_i) < f(\mathbf{x}_j).$$

- A particle p_i is called the **best** particle if p_i is better than the another particle p_j for all j = 1, 2, ..., N and $j \neq i$.
- The **fitness** F_i of a particle p_i is defined between 0 and 1 as follows:

$$F_i = NM(f(\mathbf{x}_i))$$

where *NM* is a normalization and mapping function by which the minimum value of f(·) is mapped into 1 as a best particle's fitness, while the maximum value of f(·) is mapped into 0 as a worst particle's fitness.

III. Proposed algorithm

A. *Definition of mass of a particle*

The definition of mass of a particle should satisfy the following conditions:

- A particle's mass is bounded by $[M_0, M_B]$, where M_0 is the minimum mass and M_B is the maximum mass. Without upper-bound restriction, a very heavy particle may appear and it would not move at all even if it is not located at the global optimal point. Lower-bound restriction prevents the appearance of zero weight particle.
- The better particle is heavier than the worse one, which makes the particle with lower fitness value move faster than those with higher fitness value. This affects the exploration feature and convergence speed.

Any mass function which satisfies above conditions is acceptable to represent the mass of a particle in $[M_0, M_B]$. In this paper, the mass of a particle p_i is defined as follows:

$$m_i = M_0 + (M_B - M_0)e^{-C(F_{best} - F_i)} \qquad (2)$$

where F_i is from $\mathbf{x}_i$, the location vector of p_i, F_{best} from $\mathbf{x}_{best}$, the location vector of the best particle of current generation and C is a positive nonzero constant.

To satisfy the second condition, the mass of a particle should depend on its fitness such that if a particle is better than the other particle it should be heavier than the worse one. For this purpose, the exponential function in (2) is employed.

Also the exponential function has a property to make the slope steeper as F_i approaches F_{best}. This leads to amplify the mass difference between particles, $\mathbf{x}_{best}$ and $\mathbf{x}_i$ when p_i moves close to the best one, p_{best}. The difference of objective values of the particles may be small, but the difference of their masses would be much bigger, compared to other linear mass function. For example, in Figure 1, assume that $F_{AB} = F_B - F_A$ is equal to $F_{CD} = F_D - F_C$, where F_is are the fitness of each particle. As the figure shows, however, mass difference between particle A and B is much bigger than that between the other pair.

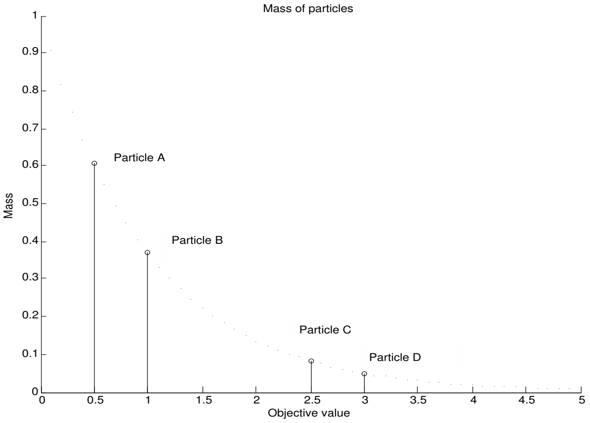

Fig. 1. Masses of four particles. The interval between particle A and B in x-axis is equal to the interval between particle C and D.

B. *The definition of potential field and the force applied to a particle*

In differential based searching algorithm, the potential field is defined by the objective function and the test vector is updated by gradient of the potential field. This approach has two problems. One is that the objective function is not continuous or not differentiable. In this case, the gradient operator is not usable. Second, gradient operator only looks near the test vector and it leads the test vector to the local optimum point.

To avoid these two problems, PD-PSO algorithm proposes a novel definition of a *potential field* determined by the particles. The particle with higher fitness value is placed near the minimum point. The minimum point of low potential which attracts the other particles in higher potential. When a particle is trapped in the local minimum point, if a fitter particle appears, it takes a position in lower potential and pulls out the trapped particle. This property minimizes the risk of converging into a local minimum.

The potential function for the optimization problem should satisfy following conditions:

- The potential function P should be a real-valued function, $P : \mathbf{R}^n \rightarrow \mathbf{R}$.
- The better particle should form lower potential than the worse.
- The difference of potential between two particles should depend on the distance of the two in the decision space.

Any function which satisfies the conditions, can be used as a potential function. In this paper, for a particle p_i, the potential difference P_{ij} caused by the particle p_j with $\Delta F_{ij} = F_j - F_i$ is modeled as follows:

$$P_{ij} = -P_0 \Delta F_{ij} ||\mathbf{r}_{ij}|| + P_{Base} \qquad (3)$$

where P_0 is a positive constant, P_{Base} is an offset term for potential value for each particle, $\mathbf{r_{ij}}$ is a distance vector from p_i to p_j and the minus sign is used to make the better particle attain lower potential. If the particle p_j is better than p_i, the

Authorized licensed use limited to: Korea Advanced Inst of Science & Tech - KAIST. Downloaded on July 15,2021 at 14:24:40 UTC from IEEE Xplore. Restrictions apply.

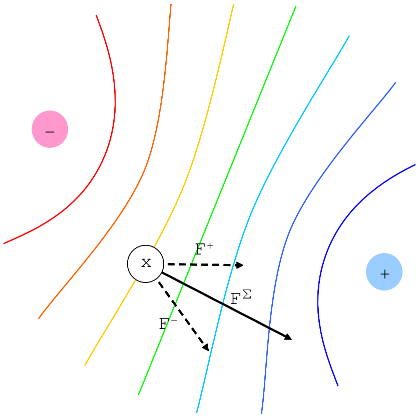

Fig. 2. Potential field and the force caused by the field. **F-** is a repulsive force applied to the particle "**x**" caused by the worse particle "**-**", while **F+** is an attractive force caused by the better particle "**+** ". The curves represent equipotential surface.

difference of fitness between two particles p_i and p_j, ΔF_{ij} would be positive and the potential would decrease when p_i approaches p_j.

At the test particle p_i, the total potential P_i can be written as follows:

$$P_i = \sum_{j=1}^{N} P_{ij} = -P_0 \sum_{j=1}^{N} (\Delta F_{ij} ||\mathbf{r}_{ij}||) + N \cdot P_{Base}. \quad (4)$$

The *force* caused by the potential field is defined by the negative of gradient of the potential. Therefore, the force applied to a particle is defined as follows:

$$\mathbf{F}_i = -\nabla P_i = P_0 \sum_{j=1}^{N} \Delta F_{ij} \hat{\mathbf{r}}_{ij} \quad (5)$$

where $\hat{\mathbf{r}}_{ij}$ is a unit vector from p_i to p_j.

The force $\mathbf{F}_{ij}$ applied to p_i which is caused by p_j can be written as

$$\mathbf{F}_{ij} = P_0 \Delta F_{ij} \hat{\mathbf{r}}_{ij}. \quad (6)$$

When ΔF_{ij} is positive, the force can be expressed as an attraction force. In other case, i.e., when $\Delta F_{ij} < 0$, the force is a repulsive force.

Assuming that a target particle is affected by all of the other particles, the repulsive forces formed by the worse particles help avoid the local optimum point, but it may cause negative effect from the viewpoint of the convergence speed. If a target particle is affected only by the better particles, it improves the convergence speed but its exploration feature may be deteriorated.

C. Update mechanism

Update mechanism for a particle's location follows Newton's dynamics model. For each time generation, a particle's position is calculated based on its current location and velocity. The particle's velocity is determined by its current velocity and the forces applied to the particle.

Basically, Newton's second law of dynamics is described as follow:

$$\mathbf{F} = m\mathbf{a} \rightarrow \mathbf{a} = \mathbf{F}/m \quad (7)$$

where **a** is the acceleration of mass *m* for applied force **F** .

Assuming that the particles' location is sampled at every Δt seconds, the movement of particles is modeled using (2), (5) and (7) as follows:

$$\mathbf{a}_i(t) = rand\,(\mathbf{F}_i/m_i) - b\mathbf{v}_i(t). \quad (8)$$

$$\mathbf{v}_i(t+\Delta t) = \mathbf{v}_i(t) + \mathbf{a}_i(t)\Delta t \quad (9)$$

$$\mathbf{x}_i(t+\Delta t) = \mathbf{x}_i(t) + \mathbf{v}_i(t)\Delta t. \quad (10)$$

Comparing (8) with (7), there exist two additional terms: *rand* and $-b\mathbf{v}_i(t)$. The first term, $rand \in [0,1]$ is added to make the procedure stochastic. Without this term, only the initial population determines the final solution. This means that the final solution is decided at the initialization process and the exploration feature of entire algorithm is weakened. The second term, $-b\mathbf{v}(t)$ term is the viscosity friction and b is a positive constant smaller than 1. When a particle is accelerated, its potential energy is converted into kinetic energy. Thus, as the particle approaches the global optimum point, where the particle has the minimum potential energy and maximum kinetic energy, it would have higher kinetic energy. This means that it moves very fast near the optimum point. This phenomenon may result in poor exploitation feature. Hence, the particle's kinetic energy should be drained to avoid this happening by the viscosity friction term which is in proportion to the particle's velocity.

Considering Δt as unit time for simplicity, above equations can be written as following update rules for PD-PSO:

$$\mathbf{v}_i \leftarrow (1-b)\mathbf{v}_i + rand\,(\mathbf{F}_i/m_i) \quad (11)$$

$$\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i. \quad (12)$$

D. Difference between the original PSO algorithm and the proposed algorithm

The original PSO algorithm's update formula is written as follows:

$$\mathbf{v}_i \leftarrow w\mathbf{v}_i + C_0\,rand\,(\mathbf{x}_{i,best} - \mathbf{x}_i) + C_1\,rand\,(\mathbf{x}_{best} - \mathbf{x}_i) \quad (13)$$

$$\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i \quad (14)$$

where $\mathbf{x}_{best}$ is the location of the global best or local best particle and $\mathbf{x}_{i,best}$ is the location of personal best of each particle.

Comparing original PSO and PD-PSO, two update equations, (12) and (14) are the same. However, velocity update equations, (11) and (13) are different from each other. The velocity update in (13) is done by considering the relative position information of related particles in the decision space. On the other hand, in PD-PSO it is carried out based on its acceleration which depends on its fitness (F_i) and the force ($\mathbf{F}_i$) from other particles.

Authorized licensed use limited to: Korea Advanced Inst of Science & Tech - KAIST. Downloaded on July 15,2021 at 14:24:40 UTC from IEEE Xplore. Restrictions apply.

TABLE I

GENERIC PSEUDO-CODE FOR PD-PSO

```
Initialize the population.
Repeat
    Evaluate each particle
    designate the best particle.
    For each particle $p_i$ do (loop #1)
        Set $\mathbf{a}_i \leftarrow \mathbf{0}$
        For each particle $p_j \neq p_i$ do
            Calculate $\mathbf{F}_{ij}$
            Update $\mathbf{a}_i \leftarrow \mathbf{a}_i + \mathbf{F}_{ij}/m_i$
        End for
    End for
    For each particle $p_i$ do (loop #2)
        Update $\mathbf{v}_i \leftarrow (1-b)\mathbf{v}_i + rand\,\mathbf{a}_i$
        Update $\mathbf{x}_i \leftarrow \mathbf{x}_i + \mathbf{v}_i$
    End for
Until stopping criterion is met
```

When a particle is trapped in the local minimum point, proposed PD-PSO provides it with more chance to escape from the trap. In original PSO, a particle is affected by itself and the other one, p_{gbest}. If p_{gbest} is located at the local optimum point, it is liable to converge to the local minimum point as well. There is no chance to avoid such a local minimum point until the other particles find the better location. On the other hand, in PD-PSO all the particles affects the other particles. This means that each particle gathers information from all the other particles and gives the trapped particle more chance to get out of the trap.

E. Generic pseudo-code

Table I is the generic pseudo code for the proposed algorithm. The time step Δt in (9) and (10) is considered as 1. Note that There are two *for* loops (*loop #1, #2*) in the main loop. If they are merged into one *for* loop, the particles calculated late are affected by the wrong position by the early-updated particles. To avoid this phenomenon, the velocity vector of each particle is updated in the first *for* loop and the particles' location is updated in the second *for* loop.

IV. EXPERIMENTAL RESULTS

A. Test functions and results

The test functions were the De Jong test functions [12], [13] and Griewangk function [12], [14]. For each test function, the domain space was considered as a *two-dimensional* space. It means that $n = 2$ for all the test functions.

For each test problem, the original PSO algorithm and the proposed algorithm were applied 10 times each. The comparison metrics were best, worst, mean, median, standard deviation of objective function value and standard deviation of solution vector **x**. Each algorithm was executed with 100 particles for 200 generations.

- First De Jong function (sphere)

$$f_1(\mathbf{x}) = \sum_{j=1}^{n} x_j^2$$

where $x_j \in [-5.12, 5.12]$.
The minimum is $f_1(0,\,0) = 0$. It is considered to be a very simple task for every serious minimization method.

- Second De Jong function (Rosenbrock's saddle)

$$f_2(\mathbf{x}) = 100 \cdot (x_0^2 - x_1)^2 + (1 - x_0)^2$$

where $x_j \in [-2.048, 2.048]$.
The minimum is $f_2(1,\,1) = 0$.

- Third De Jong Function (step)

$$f_3(\mathbf{x}) = 6 \cdot n + \sum_{j=1}^{n} [x_j]$$

where $x_j \in [-5.12, 5.12]$.
The minimum is $f_3(-5-\epsilon,\,-5-\epsilon) = 0$ where $\epsilon \in [0, 0.12]$. This function foits a lot of plateaus and is not differentiable.

- Fifth De Jong Function (Shekel's Foxholes)

$$f_4(\mathbf{x}) = \frac{1}{0.002 + \sum_{i=1}^{25} \frac{1}{i + \sum_{j=1}^{2}(x_j - a_{ij})}}$$

$$x_j \in [-65.535, 65.535]$$

with a_{i0} = {-32, -16, 0, 16, 32} for *i* = 0, 1, 2, 3, 4 and $a_{i0} = a_{i\,mod\,5,0}$ as well as a_{i1} = {-32, -16, 0, 16, 32} for *i* = 0, 5, 10, 15, 20 and $a_{i1} = a_{i+k,1}$, *k*=1, 2, 3, 4. The global minimum for this function is $f_4(-32, -32)$ = 0.9980.

- Griewangk function

$$f_5(\mathbf{x}) = \sum_{j=1}^{n} \frac{x_j^2}{4000} - \prod_{j=1}^{n} \cos \frac{x_j}{\sqrt{j+1}} + 1$$

where $x_j \in [-100, 100]$.
The minimum $f_5(0,\,0) = 0$. This function has a lot of local minima so that it is very difficult to find the true optimal point.

The simulation results are shown in Table II. For the original PSO algorithm, three parameters in (13) are selected as follows:

$$\omega = 0.7,\;\; E(C_0 rand) = 0.65,\;\; E(C_1 rand) = 0.65.$$

For the proposed PD-PSO algorithm, parameters in (2), (3) and (11) are selected as follows:

$$M_0 = 0.1,\;\; M_B = 1.0,\;\; C = 2.0,\;\; P_0 = 0.1,\;\; b = 0.3.$$

The maximum velocity of particles is limited to the 25% of the decision space size and the particles were bounded in the decision space.

The two standard deviations of solution vector for the De Jong step function f_3 is not defined. The function has the minimum plateau, not the minimum point. So the standard deviation of solution vector can not be defined.

Authorized licensed use limited to: Korea Advanced Inst of Science & Tech - KAIST. Downloaded on July 15,2021 at 14:24:40 UTC from IEEE Xplore. Restrictions apply.

	Metric	Best f	Worst f	Mean f	Median f	σ_f	$\sigma_{\|\mathbf{x}-\mathbf{x_0}\|}$
1st De Jong	Original	0.0000	0.0000	0.0000	0.0000	0.0000	0.0000
f_1	Proposed	0.0000	0.0000	0.0000	0.0000	0.0000	0.0010
2nd De Jong	Original	0.0000	0.0003	0.0001	0.0000	0.0002	0.0001
f_2	Proposed	0.0000	0.0006	0.0003	0.0003	0.0002	0.0003
3rd De Jong	Original	0.0000	0.0000	0.0000	0.0000	0.0000	N/A
f_3	Proposed	0.0000	0.0000	0.0000	0.0000	0.0000	N/A
5th De Jong	Original	0.9980	5.9288	2.2855	1.9920	1.4743	11.8750
f_4	Proposed	0.9980	1.4558	1.1092	1.0064	0.1727	0.2261
Griewangk	Original	0.0000	0.0271	0.0072	0.0074	0.0078	3.3433
f_5	Proposed	0.0000	0.0011	0.0003	0.0001	0.0004	0.0171

TABLE II

SIMULATION RESULTS FOR THE 5 TEST FUNCTIONS

Original PSO			Proposed PD-PSO		
x_0	x_1	f	x_0	x_1	f
-32.0800	-31.9007	0.9980	-32.1710	-31.8367	0.9980
-0.0132	-31.9651	2.2821	-32.2627	-32.5844	1.0380
-15.9864	-31.9703	1.9920	-32.3232	-31.1499	1.3747
-15.9864	-31.9703	1.9920	-32.3390	-32.2057	0.9996
-31.9783	-31.9783	0.9980	-32.1797	-31.2668	1.4558
-15.9864	-31.9703	1.9920	-32.3761	-31.5784	1.0064
-31.9539	-15.9779	5.9288	-32.7446	-31.4595	1.1925
-15.9864	-31.9703	1.9920	-31.7287	-32.3775	1.0013
-31.9783	-31.9783	0.9980	-32.2148	-31.6756	0.9993
0.0132	-31.9651	2.9821	-32.5494	-32.2457	1.0256

TABLE III

SIMULATION RESULT FOR THE SHEKEL'S FOXHOLES FUNCTION

Original PSO			Proposed PD-PSO		
x_0	x_1	f	x_0	x_1	f
3.1400	-4.4384	0.0074	0.0065	0.0077	0.0000
3.1400	4.4384	0.0074	0.0082	-0.0365	0.0004
0.0000	0.0000	0.0000	-0.0159	-0.0130	0.0002
-3.1400	4.4384	0.0074	-0.0055	0.0034	0.0000
3.1400	4.4384	0.0074	-0.0138	-0.0263	0.0003
0.0000	0.0000	0.0000	0.0069	-0.0122	0.0001
-9.4201	-4.4384	0.0271	-0.0272	0.0527	0.0011
-3.1400	4.4384	0.0074	0.0330	-0.0137	0.0006
0.0000	0.0000	0.0000	0.0063	-0.0075	0.0000
3.1400	4.4384	0.0074	-0.0085	0.0012	0.0000

TABLE IV

SIMULATION RESULT FOR THE GRIEWANGK FUNCTION

B. *Result analysis*

The first two test functions, De Jong Sphere function and Rosenbrock's saddle, have only one minimum point. Both of the algorithms showed good results (Table II). For the Rosenbrock's saddle problem, the original PSO showed just a little bit better result in terms of convergence.

The third test function, De Jong Step function, both of the algorithms found the *minimum plateau*, as Table II shows. All of the results from 20 test cases are placed in the lowest plateau, where the objective function value is minimized.

There were some noticeable results for the forth and the fifth test functions, as shown in Table III and Table IV, respectively. The Griewangk function and the Shekel's Foxholes function have a lot of local minima and the true minimum is surrounded by the false optimum points. The Shekel's Foxholes function has local minima at (-16, -32), (-32, -16), (-16, -16) and so on. The proposed PD-PSO algorithm always found the true minimum point. The solutions produced by the proposed algorithm were located near the global minimum point at (-32, -32). The proposed algorithm successfully avoided the local minimum points, while the original PSO algorithm trapped in the local optimal points in some cases. For the Griewangk function, same phenomenon happened. The proposed algorithm converged to the global optimum placed at (0, 0) during the ten test executions. The local optimum points of the test function are located at (±3.1400, ±4.4484) and so on. The original algorithm trapped in the local minima in some test executions, while the proposed one was not. Experimental results showed that proposed PD-PSO was robust against the local minima, compared to the original PSO. This comes from the exploration capability of the PD-PSO.

V. CONCLUSION

This paper proposed a novel PSO-based algorithm, PD-PSO by introducing potential field and motion dynamics to the concept of original PSO. To derive the motion dynamics, the mass of a particle was calculated from its fitness and the force to the particle is obtained from the potential field formed by other particles based on their fitness value.

To demonstrate the performance of the PD-PSO, five test functions were employed. From the experimental results, it was shown that the proposed PD-PSO algorithm could be used as a global optimizer over both differentiable and indifferentiable search space. Also by the test functions with many local optimal points, it was confirmed that the proposed algorithm had better exploration feature.

As a further work, the fortified exploration feature weakens the exploitation capability and this should be studied more.

REFERENCES

[1] Kennedy, J., Eberhart, R. C., "Particle Swarm Optimization," Proc. of the IEEE International Conf. on Neural Nerworks, vol. IV, pp. 1942-1948. IEEE Press, Piscataway, ND (1995)

Authorized licensed use limited to: Korea Advanced Inst of Science & Tech - KAIST. Downloaded on July 15,2021 at 14:24:40 UTC from IEEE Xplore. Restrictions apply.

[2] Robinson, J., Rahmat-Samii, Y., “Particle Swarm Optimization in Electromagnetics,” IEEE Transactions on Antennas and Propagation, vol. 52, pp. 397-407, IEEE Press, 2004.

[3] Eberhart, R.C., Xiaohui Hu, “Human Tremor Analysis using particle swarm optimization,” Proc. of the Congress on Evolutionary Computation, Washington DC., July 1999, pp. 1945-1949.

[4] Coelllo Coello, C. A., Lechuga, M. S., “MOPSO: A Proposal for Multiple Objective Particle Swarm Optimization,” Proc. of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA, vol. 2, pp. 1051-1056, 2002.

[5] Kim, J.-H., Myung, H., “Evolutionary Programming Techniques for Constrained Optimization Problems,” IEEE Trans. on Evolutionary Computation, vol. 1, no. 1, pp. 119-128, May. 1997

[6] Myung, H., Kim, J-. H., “Constrained Optimization Using Two-phase Evolutionary Programming,” Proc. if the IEEE International Conference of Evolutionary Computation, Nagoya, Japan, pp. 262-267, 1996.

[7] Myung, H., Kim, J-. H., “Evolian: Evolutionary Optimization Based on Lagrangian with Constraint Scaling,” Lecture Notes in Computer Science, Springer Berlin, Vol. 1213, pp. 177-187, 1997.

[8] Xiaohui Hu, Eberhart, R. C., “Adaptive Particle Swarm Optimization: Detection and Response to Dynamic System,” Proc. of the IEEE Congress on Evolutionary Computation, Honolulu, Hawaii, USA, pp. 1666-1670. IEEE Computer Society Press, Los Alamitos (2002)

[9] Carlisle, A., Dozier, G., “Adapting Particle Swarm Optimization to Dynamic Environments,” Proc. of the ICAI, Las Vegas, NV, vol. 1, pp. 429-434 (2000)

[10] Hu, U., Zeng, J., Tan, Y., “A Diversity-Guided Particle Swarm Optimizer for Dynamics Environments,” Lecture Notes in Computer Sciences, vol. 4668, pp. 239-247, Springer Berlin, Heidelberg (2007)

[11] Blackwell, T. M., Bentley, P. J., “Dynamics Search with Charged Swarms,” Proc. of the Genetic and Evolutionary Computation Conference, W. B. Langdon *et al.*, Eds., 2002, pp. 19-26.

[12] Storn, R., Price, K., “Differential Evolution - A Simple and efficient adaptive scheme for global optimization over continuous spaces,” Journal of Global Optimization, Springer Netherlands, Vol. 11, No. 4, December 1997, pp. 341-359.

[13] Ingber, L. and Rosen, B., “Genetic Algorithms and Very Fast Simulated Reannealing: A Comparison,” J. Mathl. Comput. Modeling, Vol. 16, No. 11, 1992, pp. 87 - 100.

[14] Corona, A., Marchesi, M., C. and Ridella, S., “Minimizing Multimodal Functions of Continuous Variables with the Simulated Annealing Algorithm,” ACM Trans. Mathl. Software, march 1987, pp. 272-280.

Authorized licensed use limited to: Korea Advanced Inst of Science & Tech - KAIST. Downloaded on July 15,2021 at 14:24:40 UTC from IEEE Xplore. Restrictions apply.