International Journal of Control and Automation
Vol.7, No.7 (2014), pp.15-32
http://dx.doi.org/10.14257/ijca.2014.7.7.02

Robot Path Planning based on Swarm Intelligence

Xuesong Yan[1], Qinghua Wu[2,3*], Chengyu Hu[1], Hong Yao[1], Yuanyuan Fan[1], Qingzhong Liang[1] and Chao Liu[1]

[1]School of Computer Science, China University of Geosciences, Wuhan, P. R China
[2]Hubei Provincial Key Laboratory of Intelligent Robot, Wuhan Institute of Technology, Wuhan, P. R China
[3]School of Computer Science and Engineering, Wuhan Institute of Technology, Wuhan, P. R China
**Corresponding author E-mail: yanxs1999@126.com, wuqinghua@sina.com*

Abstract

Robot path planning is a NP problem, traditional optimization methods are not solve it very well just like genetic algorithm, which are easy to trap into local optimal. Particle Swarm Optimization (PSO) algorithm was developed under the inspiration of behavior laws of bird flocks, fish schools and human communities, compared with genetic algorithm the PSO algorithm has high convergence speed. In this paper, aim at the disadvantages of standard PSO algorithm like being trapped easily into a local optimal, we improves the standard PSO and proposes a new algorithm to solve the overcomes of the standard PSO. The new algorithm keeps not only the fast convergence speed characteristic of PSO, but effectively improves the capability of global searching as well. Compared with genetic algorithm on the robot path planning problem, the results show that the new algorithm can get more accuracy path and the calculation time is faster.

Keywords: *robot path planning, particle swarm optimization, genetic algorithm, NP problem, function*

1. Introduction

Autonomous robots are developed to perform high level task without further human operation. To accomplish these tasks the robots need to move in the real world. In consequence, one of the more important problems to be solved in the design of autonomous robots is the path planning. These planning problems can involve geometric workspace, constraints and additional complex features such as incomplete knowledge, moving obstacles, sensing and model uncertainties, unpredictability, kinematics, multiple robots and goals. In the last decade, path planning has received considerable attention from robotic communities since this fundamental operation requires the solution of a variety of theoretical and practical problems.

Robot path planning is a NP problem [1], traditional optimization methods are not very effective to it, which are easy to plunge into local minimum. A lot of algorithms have been proposed to solve NP problem. Some of them (based on dynamic programming or branch and bound methods) provide the global optimum solution. Other algorithms are heuristic ones, which are much faster, but they do not guarantee the optimal solutions.

Until now, the research in robot path planning has focused on finding optimal routes from start to goal. The optimality is usually measured in terms of traveled distances [2]. Other measures are also used, e.g. confidence value [3]. For planetary rovers the efficiency of path is often expressed in terms of slope or roughness of the surface [4, 5].

Particle Swarm Optimization (PSO) algorithm was an intelligent technology first presented in 1995 by Eberhart and Kennedy, and it was developed under the inspiration of behavior laws of bird flocks, fish schools and human communities [6]. If we compare PSO with Genetic Algorithms (GA), we may find that they are all maneuvered on the basis of population operated. But PSO doesn't rely on genetic operators like selection operators, crossover operators and mutation operators to operate individual, it optimizes the population through information exchange among individuals. PSO achieves its optimum solution by starting from a group of random solution and then searching repeatedly. Once PSO was presented, it invited widespread concerns among scholars in the optimization fields and shortly afterwards it had become a studying focus within only several years. A number of scientific achievements had emerged in these fields [7-9]. PSO was proved to be a sort of high efficient optimization algorithm by numerous research and experiments [10]. PSO is a meta-heuristic as it makes few or no assumptions about the problem being optimized and can search very large spaces of candidate solutions. However, meta-heuristics such as PSO do not guarantee an optimal solution is ever found. More specifically, PSO does not use the gradient of the problem being optimized, which means PSO does not require that the optimization problem be differentiable as is required by classic optimization methods such as gradient descent and quasi-Newton methods. PSO can therefore also be used on optimization problems that are partially irregular, noisy, change over time, *etc*. This paper improves the disadvantages of standard PSO being easily trapped into a local optimum and proposed an improved PSO algorithm (IPSO) which proves to be more simply conducted and with more efficient global searching capability, then use the new algorithm for robot path planning problem.

2. Particle Swarm Optimization Algorithm

2.1. PSO Algorithm

A basic variant of the PSO algorithm works by having a population (called a swarm) of candidate solutions (called particles). These particles are moved around in the search-space according to a few simple formulae. The movements of the particles are guided by their own best known position in the search-space as well as the entire swarm's best known position. When improved positions are being discovered these will then come to guide the movements of the swarm. The process is repeated and by doing so it is hoped, but not guaranteed, that a satisfactory solution will eventually be discovered. Formally, let $f : R^n \to R$ be the cost function which must be minimized. The function takes a candidate solution as argument in the form of a vector of real numbers and produces a real number as output which indicates the objective function value of the given candidate solution. The gradient of is not known. The goal is to find a solution a for which $f(a) \le f(b)$ for all b in the search-space, which would mean a is the global minimum. Maximization can be performed by considering the function $h = -f$ instead.

PSO was presented under the inspiration of bird flock immigration during the course of finding food and then be used in the optimization problems. In PSO, each optimization problem solution is taken as a bird in the searching space and it is called “particle”. Every particle has a fitness value which is determined by target functions and it has also a velocity which determines its destination and distance. All particles search in the solution space for their best positions and the positions of the best particles in the swarm. PSO is initially a group of random particles (random solutions), and then the optimum solutions are found by repeated searching. In every iteration, a particle will follow two bests to renew itself: the best position found for a particle called pbest; the best position found for the whole swarm called gbest. All particles will determine following steps through the best experiences of individuals themselves and their companions.

For particle id, its velocity and its position renewal formula are as follows:

$$V_{id}^{'} = \omega V_{id} + \eta_1 rand()(P_{idb} - X_{id}) + \eta_2 rand()(P_{gdb} - X_{id}) \tag{1}$$

$$X_{id}^{'} = X_{id} + V_{id}^{'} \tag{2}$$

In here: ω is called inertia weight, it is a proportion factor that is concerned with former velocity, $0 < \omega < 1$, η_1 and η_2 are constants and are called accelerating factors, normally $\eta_1 = \eta_2 = 2$; $rand()$ are random numbers, X_{id} represents the position of particle id ; V_{id} represents the velocity of particle id ; P_{id} , P_{gd} represent separately the best position particle id has found and the position of the best particles in the whole swarm.

In formula(1), the first part represents the former velocity of the particle, it enables the particle to possess expanding tendency in the searching space and thus makes the algorithm be more capable in global searching; the second part is called cognition part, it represents the process of absorbing individual experience knowledge on the part of the particle; the third part is called social part, it represents the process of learning from the experiences of other particles on the part of certain particle, and it also shows the information sharing and social cooperation among particles.

The flow of PSO can briefly describe as following: First, to initialize a group of particles, *e.g.*, to give randomly each particle an initial position Xi and an initial velocity Vi, and then to calculate its fitness value f. In every iteration, evaluated a particle's fitness value by analyzing the velocity and positions of renewed particles in formula (1) and (2). When a particle finds a better position than previously, it will mark this coordinate into vector P1, the vector difference between P1 and the present position of the particle will randomly be added to next velocity vector, so that the following renewed particles will search around this point, it's also called in formula (1) cognition component. The weight difference of the present position of the particle swarm and the best position of the swarm Pgd will also be added to velocity vector for adjusting the next population velocity. This is also called in formula (1) social component. These two adjustments will enable particles to search around two bests.

The most obvious advantage of PSO is that the convergence speed of the swarm is very high, scholars like Clerc [7] has presented proof on its convergence. Here a fatal weakness may result from this characteristic. With constant increase of iterations, the velocity of particles will gradually diminish and reach zero in the end. At this time, the whole swarm will be converged at one point in the solution space, if gbest particles haven't found gbest, the whole swarm will be trapped into a local optimum; and the capacity of swarm jump out of a local optimum is rather weak.

2.2. Experiment Comparison

In order to verify the convergence speed of the PSO algorithm, we selected four benchmarks function and compared the results with traditional genetic algorithm (GA).

F1: Schaffer function

$$\min f_1(x_i) = 0.5 - \frac{(\sin^2 \sqrt{x_1^2 + x_2^2} - 0.5)}{[1 + 0.001(x_1^2 + x_2^2)]^2}, -10 \le x_i \le 10$$

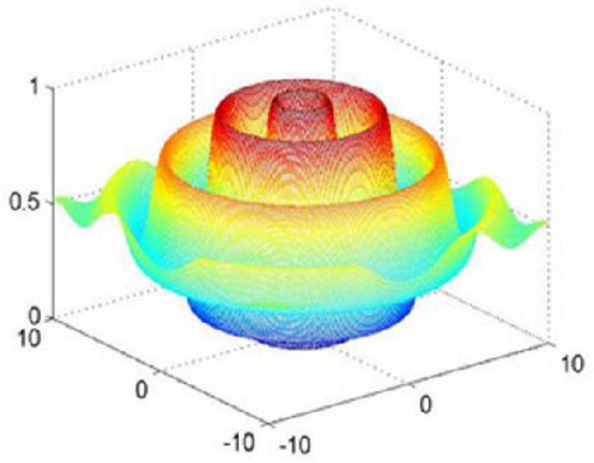

Figure 1. Schaffer Function

In this function the biggest point is in the situation where xi= (0, 0) and the global optimal value is 1.0, the largest in the overall points for the center, and 3.14 for the radius of a circle on the overall situation from numerous major points of the uplift. This function has a strong shock; therefore, it is difficult to find a general method of its global optimal solution.

F2: Shubert function

$$\min f_2(x,y) = \left\{ \sum_{i=1}^{5} i\cos\left[(i+1)x+i\right] \right\} \times \left\{ \sum_{i=1}^{5} i\cos\left[(i+1)y+i\right] \right\}, x, y \in [-10,10]$$

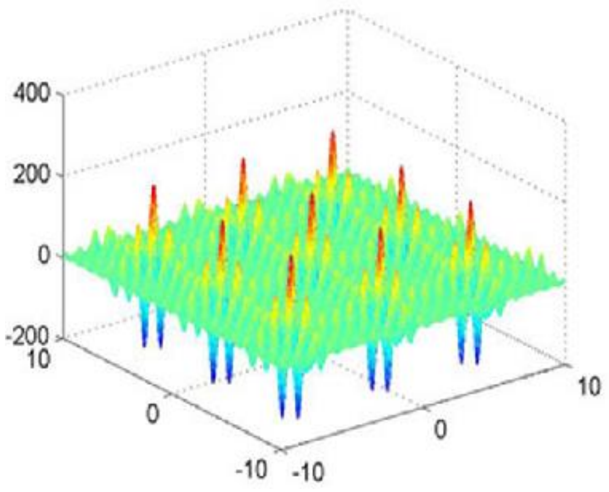

Figure 2. Shubert Function

This function has 760 local minimum and 18 global minimum, the global minimum value is -186.7309.

F3: Hansen function

$$\min f_3(x,y) = \sum_{i=1}^{5} i\cos((i-1)x+i)\sum_{j=1}^{5} j\cos((j+1)y+j),\ x,y\in[-10,10]$$

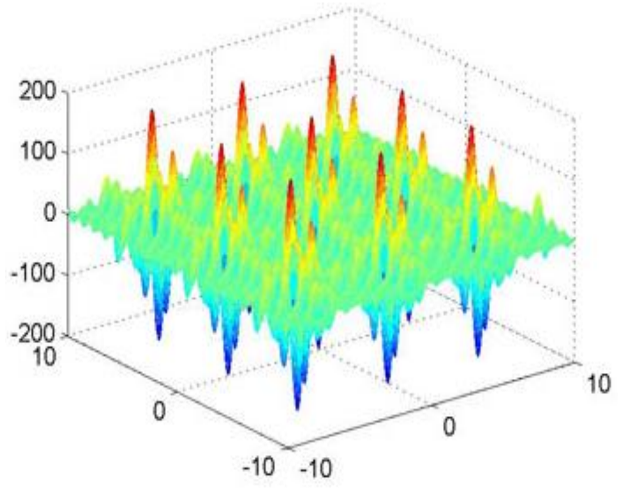

Figure 3. Hansen Function

This function has a global minimum value -176.541793, in the following nine point (-7.589893, -7.708314), (-7.589893, -1.425128), (-7.589893, 4.858057), (-1.306708, -7.708314), (-1.306708, -1.425128), (-1.306708, 4.858057), (4.976478, -7.708314), (4.976478, -7.708314), (4.976478, 4.858057) can get this global minimum value, the function has 760 local minimum.

F4: Camel function

$$\min f_4(x,y) = \left(4 - 2.1x^2 + \frac{x^4}{3}\right)x^2 + xy + \left(-4+4y^2\right)y^2,$$

$$x,y\in[-100,100]$$

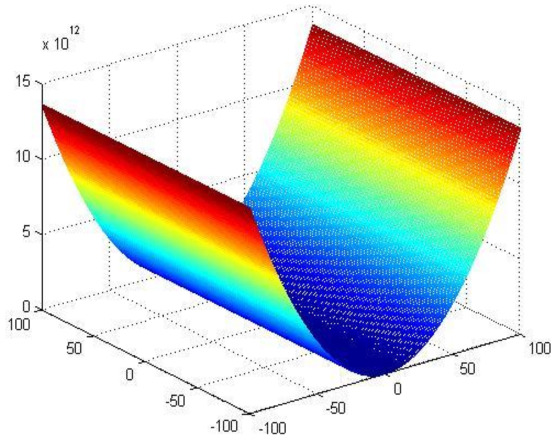

Figure 4. Camel Function

Camel function has 6 local minimum (1.607105, 0.568651), (-1.607105, -0.568651), (1.703607, -0.796084), (-1.703607, 0.796084), (-0.0898, 0.7126) and (0.0898,-0.7126), the (-0.0898,0.7126) and (0.0898,-0.7126) are the two global minimums, the value is -1.031628.

The two algorithms of the same experimental parameters set. In the experiment, each case is run for 100 times. Table 1 shows the statistics of our experimental results. GA found the known optimal solution to F1 with the mean convergence generations is 36042 and the mean convergence time is 1024 seconds, to F2 with the mean convergence generations is 8344 and the mean convergence time is 148 seconds, to F3 with the mean convergence generations is 81110 and the mean convergence time is 2048 seconds, and to F4 with the mean convergence generations is 240188 and the mean convergence time is 6098 seconds,; PSO algorithm is efficiency for the four cases: found the known optimal solution to F1 with the mean convergence generations is 6534 and the mean convergence time is 123 seconds,, found the known optimal solution to F2 with the mean convergence generations is 2187 and the mean convergence time is 56 seconds, found the known optimal solution to F3 with the mean convergence generations is 10587 and the mean convergence time is 356 seconds, and found the known optimal solution to F4 with the mean convergence generations is 29534 and the mean convergence time is 890 seconds. In sum, we can see that in solving the four functions, the PSO algorithm's convergence speed is more efficient than genetic algorithm.

Table 1. Experiment Results Comparison

Function	Algorithm	Mean convergence generations	Mean convergence time(s)
F1	GA	36042	1024
	PSO	6534	123
F2	GA	8344	148
	PSO	2187	56
F3	GA	81110	2048
	PSO	10587	356
F4	GA	240188	6098
	PSO	29534	890

3. Improved PSO Algorithm(IPSO)

3.1. Improvement of PSO

In the standard PSO algorithm, the convergence speed of particles is fast, but the adjustments of cognition component and social component make particles search around Pgd and Pid. According to velocity and position renewal formula, once the best individual in the swarm is trapped into a local optimum, the information sharing mechanism in PSO will attract other particles to approach this local optimum gradually, and in the end the whole swarm will be converged at this position. But according to velocity and position renewal formula (1), once the whole swarm is trapped into a local optimum, its cognition component and social component will become zero in the end; still, because $0 < \omega < 1$ and with the number of iteration increase, the velocity of particles will become zero in the end, thus the whole swarm is hard to jump out of the local optimum and has no way to achieve the global optimum. Here a fatal weakness may result from this characteristic. With constant increase of iterations, the velocity of particles will gradually diminish and reach zero in the end. At this time, the whole swarm will be converged at one point in the solution space, if gbest particles haven't found gbest, the whole swarm will be trapped into a local optimum; and the capacity of swarm jump out of a local optimum is rather weak. In order to get through this disadvantage, in this paper we presents a new algorithm based on PSO. In order to avoid being trapped into a local optimum, the new PSO adopts a new information sharing mechanism. We all know that when a particle is searching in the solution space, it doesn't know the exact position of the optimum solution. But we can not only record the best positions an individual particle and the whole swarm have experienced, we can also record the worst positions an individual particle and the whole swarm have experienced, thus we may make individual particles move in the direction of evading the worst positions an individual particle and the whole flock have experienced, this will surely enlarge the global searching space of particles and enable them to avoid being trapped into a local optimum too early, in the same time, it will improve the possibility of finding gbest in the searching space. In the new strategy, the particle velocity and position renewal formula are as follows:

$$V_{id}^{'} = \omega V_{id} + \eta_1 rand()(X_{id} - P_{idw}) + \eta_2 rand()(X_{id} - P_{gdw}) \tag{3}$$

$$X_{id}^{'} = X_{id} + V_{id}^{'} \tag{4}$$

In here: P_{idw} , P_{gdw} represent the worst position particle id has found and the worst positions of the whole swarm has found.

In standard PSO algorithm, the next flying direction of each particle is nearly definite; it can fly to the best individual and the best individuals for the whole swarm. From the above conclusion we may easily to know it will be the danger for being trapped into a local optimum. In order to decrease the possibility of being trapped into the local optimum, the improved PSO introduces elite selection strategy. Traditional genetic algorithm is usually complete the selection operation based on the individual's fitness value, in the mechanism of elite selection, the population of the front generation mixed with the new population which generate through genetic operations, in the mixed population select the optimum individuals according to a certain probability. The characteristic of this strategy is mainly in the following aspects. First is robust, because of using this selection strategy, even when the genetic operations to produce more inferior individuals, as the results of the majority of individual residues of the original population, does not cause lower the fitness value of the individual. The second is in genetic diversity maintaining, the operation of large populations, you can better maintain the genetic diversity of the population evolution process. Third is in the sorting method, it is good to overcome proportional to adapt to the calculation of scale. This process of this strategy in improve PSO like this: To set particle number in the swarm as m, father population and son population add up to 2m. To select randomly q pairs from m; as to each individual particle i, if the fitness value of i is smaller than its opponents, we will win out and then add one to its mark, and finally select those particles which have the maximum mark value into the next generation.

The flow of the improved PSO is as follows:

Step 1: to initialize randomly the velocity and position of particles;

Step 2: to evaluate the fitness value of each particle;

Step 3: as to each particle, if its fitness value is smaller than the best fitness value P_{idb} , renew the best position P_{idb} of particle id ; or else if its fitness value is bigger than the worst fitness value P_{idw} , renew P_{idw} ;

Step 4: as to each particle, if its fitness value is smaller than the best whole swarm fitness value P_{gdb}, renew the best fitness value P_{gdb} of particle id; or else if bigger than the worst whole swarm fitness value P_{gdw}, renew P_{gdw};

Step 5: as to each particle,

1) To produce new particle t by applying formula (3), (4),

2) To produce new particle t' by applying formula (5), (6),

3) To make a comparison between t and t' and then select the better one into the next generation;

Step 6: to produce next generation particles according to the above genetic selection strategy;

Step 7: if all the above steps satisfy suspension needs, suspend it; or turn to Step 3.

3.2. Experiment Verify

In order to verify the improvement of the new algorithm based on PSO, we select the seven benchmark functions to test the performance of the new algorithm.

$$f_5(x) = \sum_{i=1}^{n} x_i^2,\ x_i \in (-100,100),\ f_{\min} = 0$$

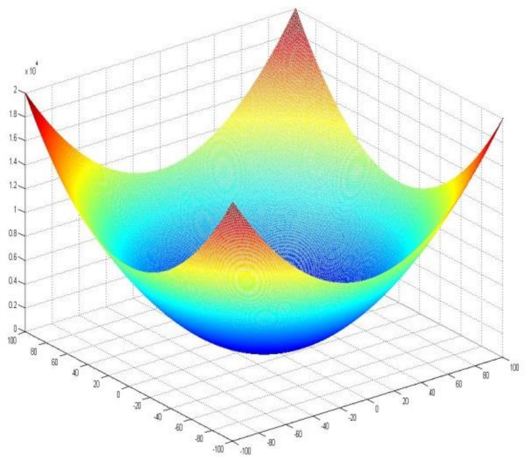

Figure 5. Function F5

$$f_6(x) = 6 \cdot \sum_{i=1}^{5} \lfloor x_i \rfloor,\ x_i \in (-5.12,5.12),\ f_{\min} = 0$$

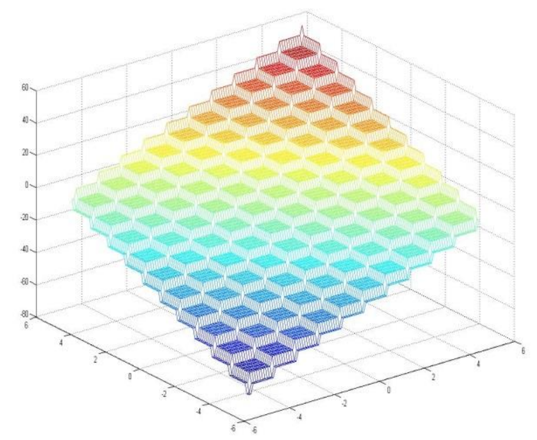

Figure 6. Function F6

$$f_7(x) = \sum_{i=1}^{n} i \cdot x_i^4 + U\left(0,1\right), x_i \in (-1.28,1.28),\ f_{\min} = 0$$

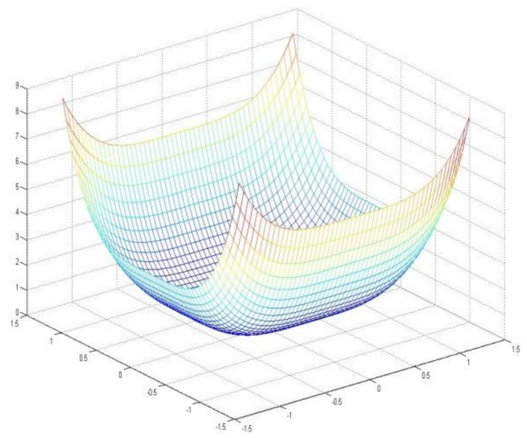

Figure 7. Function F7

$$f_8(x) = \frac{1}{4000}\sum_{i=1}^{n}(x_i - 100)^2 - \prod_{i=1}^{n}\cos(\frac{x_i - 100}{\sqrt{i}}) + 1, x_i \in (-300,300),\ f_{\min} = 0$$

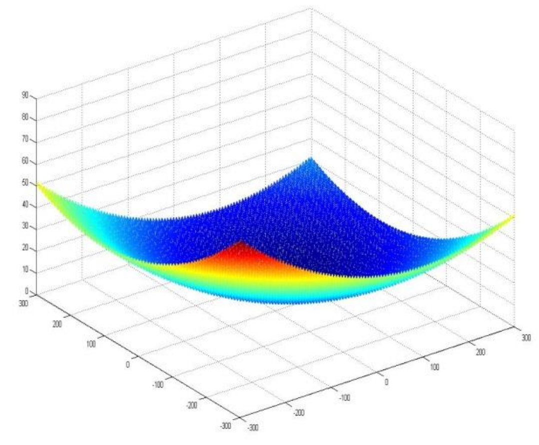

Figure 8. Function F8

$$f_9(x) = -20 \cdot \exp(-0.2\sqrt{\frac{1}{n} \cdot \sum_{i=1}^{n} x_i^2}) - \exp(\frac{1}{n} \cdot \sum_{i=1}^{n} \cos(2\pi \cdot x_i)) + 20 + e, x_i \in (-32,32),\ f_{\min} = 0$$

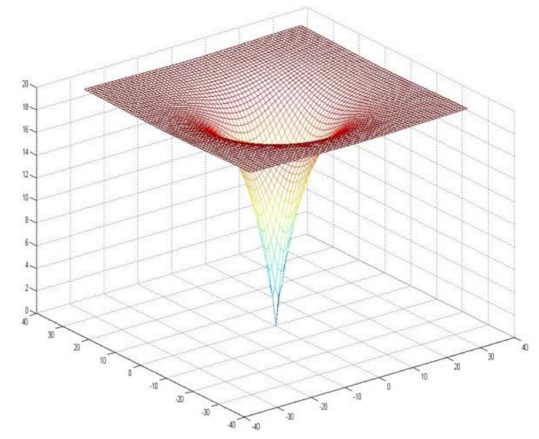

Figure 9. Function F9

$$f_{10}(x) = \sum_{i=1}^{n} 100((x_{i+1} - x_i^2)^2 + (x_i - 1)^2), x_i \in (-2.048, 2.048),\ f_{\min} = 0$$

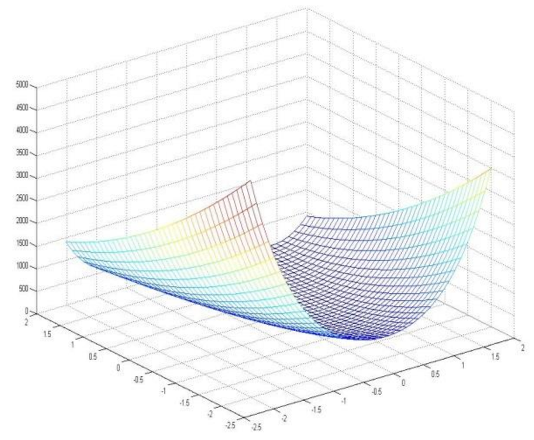

Figure 10. Function F10

$$f_{11}(x) = \sum_{i=1}^{n} -x_i \sin(\sqrt{|x_i|}),\ x_i \in (-500,500),\ f_{\min} = -12569.5$$

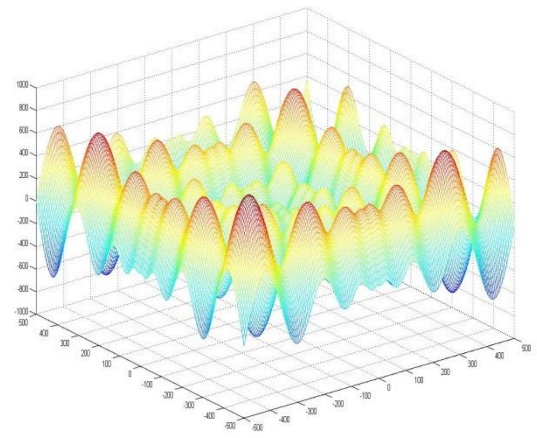

Figure 11. Function F11

The two algorithms of the same experimental parameters set. Each function in Figure 5 to Figure 11 is run 50 times with the two algorithms, their experimental results such as Table 2. By analyzing the experimental results we know, in solving function f1, f4 and f7, use the PSO algorithm is easily into local optimum, but use the new algorithm, convergence soon, and find a better solution, the average fitness and the best fitness was both superior to PSO algorithm. For the function f2, the new algorithm and PSO all can find the global optimal, these two algorithm for this test function is very effective. For function f5, the two algorithms can find the best solutions are the same (see Table 2), and the new algorithm to get the best value of the average is better than PSO algorithm. In sum, we can see that in solving function f1, f4, f5 and f7, the new algorithm more efficient than PSO algorithm, in solving other function, the performance almost same of the two algorithms. In short, this new algorithm has the following value: better global search capability.

Table 2. Experiment Results Comparison

Function	Algorithm	Best value	Worst Value	Standard deviation
F5	PSO	1495.71	7032.89	201.038
	IPSO	4.13731E-29	1.0882E-24	2.28015E-26
F6	PSO	0	0	0
	IPSO	0	0	0
F7	PSO	0.00177094	0.00833963	0.000210055
	IPSO	0.00193565	0.0103595	0.000259903
F8	PSO	72.5069	123.954	1.52289
	IPSO	2.18559E-12	8.63194E-25	0.00177512
F9	PSO	-3.19744E-14	4.4229	0.148418
	IPSO	-3.19744E-14	1.50229	0.0749509
F10	PSO	1.84889E-28	8.63194E-25	1.83991E-26
	IPSO	2.55147E-28	1.20401E-23	2.41678E-25
F11	PSO	-5038.62	-3233.13	54.0123
	IPSO	-9535.19	-8203.56	45.1661

4. Robot Path Planning Algorithm based on IPSO

4.1. Problem Definition

Robot path planning can generally be considered as a search in a configuration space defined in what is known as the configuration space formulation [11-13]: Let A be a single rigid object, moving in a Euclidean space $W = \Re^N$, $N = 2 or 3$. Let $O_1,...,O_n$ be fixed rigid objects distributed in W. The $O_i's$ are called obstacles. If A is described as a compact subset of W, and the obstacles $O_1,...,O_n$ are closed subsets of W , a configuration of A is a specification of the position of every point in A with respect to FW , where FW is a Cartesian coordinate system. The configuration space of A is the space denoted by C , with all possible configurations of A . The subset of W occupied by A at configuration q is denoted by $A(q)$. A path from an initial configuration q_{init} to a goal configuration q_{goal} is a continuous map $\tau : [0,1] \rightarrow C$ with $\tau(0) = q_{init}$ and $\tau(1) = q_{goal}$.

The workspace contains a finite number of obstacles denoted by o_i with i=1,…,n . Each obstacle o_i , maps in C to a region $C(B_i) = \{q \in C \mid A(q) \cap O_i \neq 0\}$ which is called C obstacle The union of all the C obstacle is the C obstacle region, $\overset{n}{\underset{i=1}{U}} C(O_i)$ and the set $C_{free} = c \setminus \overset{n}{\underset{i=1}{U}} C(O_i) = \{q \in C \setminus A(q) \cap (\overset{n}{\underset{i=1}{U}} O_i = 0\}$ is called the free space. A free collision path between two configurations is any continuous path $\tau : [0,1] \rightarrow C_{\text{free}}$.

The configuration space is a powerful conceptual tool because it seems to be the natural space where the path planning problem "lives". This is mainly because any transformation of a rigid or articulated body becomes a point in the configuration space.

From mathematical point of view [14], robot path planning problem can be expressed as a find the optimal value of the goal function, the goal function is the cost of the path planning, constraints condition is to avoid collision with the obstacles, so the mathematic model of the robot path planning can expression like the following formula:

$$\min f(X), X \in R^n; s.t. g_i(X) \leq b_i, i = 1, 2, \cdots p \tag{5}$$

Where, the $f(\mathrm{X})$ is the goal function, $g_i(\mathrm{X})$ is the constraints condition and p denotes the numbers of the constraints in equation.

4.2. Path Evaluation

Find fitness function is an important part of evolutionary algorithms. For the robot path planning problem, if the fitness function only consider the length of the individual path is not enough, taking into account the situation of the obstacles cross with individual paths. For this reason, in our algorithm fitness function measures the optimality of each path considering two factors: the distance between the final robot position and the goal point and the cross situation which the individual have no cross with the obstacles. Fitness function formula like following:

$$f(S_i) = \frac{1}{\sum_{i=d_1}^{d_2 - 1} d(v_i, v_{i+1})} \tag{6}$$

Where v_{d1}, v_{d2} is the location of the initial point s and the goal point g in the chromosome Si .

4.3. Simulation Result

We simulated the robot path planning algorithm with computer, and compared the planned path with the genetic algorithm. In the experiment, we generated the obstacles randomly and run algorithm 50 times. Figure 12 is the genetic algorithm result, left corner is start point and right corner is intention point, the optimum path is shown with green line. Figure 13 is our algorithm result. From the result figures, the simulation results indicated that our algorithm is efficiency for solving the robot path planning problems and it can found the right path. We also compared the calculation time with the two methods in Table 3, the results show our algorithm is faster than the genetic algorithm.

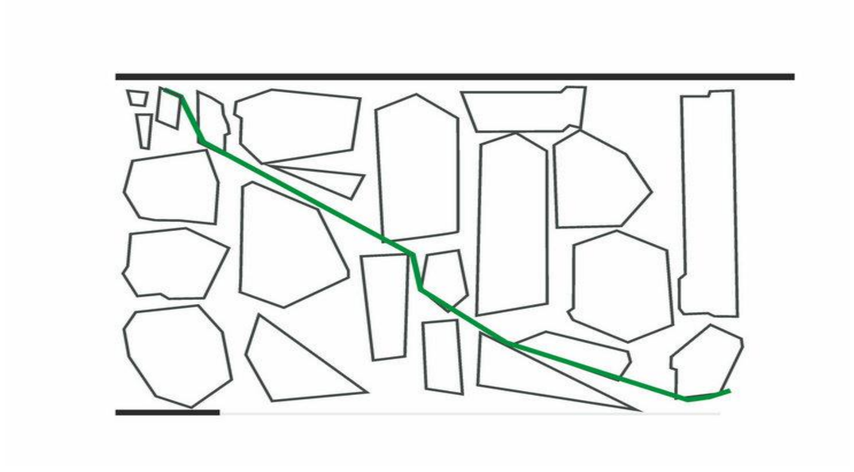

Figure 12. The Genetic Algorithm Result

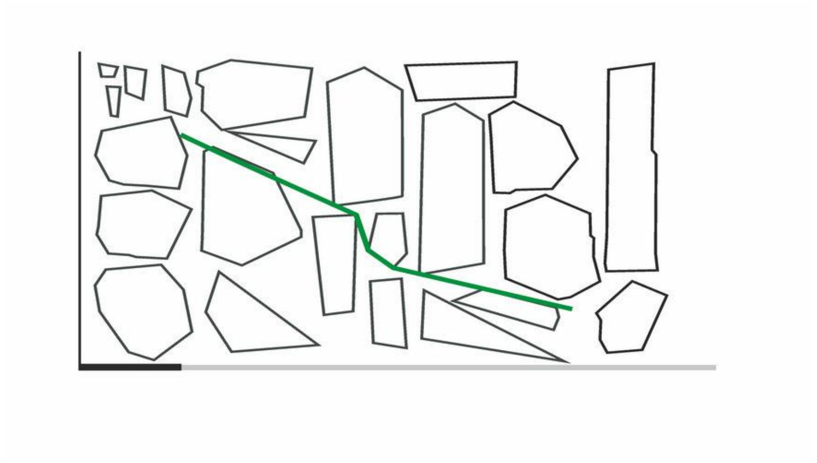

Figure 13. Our Algorithm Result

Table 3. Time Comparison of Two Algorithms

Algorithm	Max time(s)	Min time(s)	Ave time(s)
GA	89	43	56
IPSO	68	36	42

5. Conclusion

This paper introduce a new algorithm based on the standard PSO algorithm, for the standard PSO algorithm the new algorithm has done two improvements: 1. By introducing a new information sharing mechanism, make particles moved on the contrary direction of the worst individual positions and the worst whole swarm positions, thus enlarge global searching space and reduce the possibility of particles to be trapped into a local optimum; 2. By introducing elite selection strategy, decreased the possibility of being trapped into a local optimum. Compared with the standard PSO algorithm, the new algorithm enlarges the searching space and the complexity is not high. By analyzing the testing results of Benchmarks optimization, we reach the conclusion: in the optimization precision and the optimization speed, the new algorithm is efficiency than the standard PSO algorithm and the new algorithm is more efficient than PSO in coping with function optimization problems. We also use this new algorithm to solve robot path planning problem and compare the simulation result with genetic algorithm, the result shows the new algorithm is efficient than genetic algorithm.

Acknowledgements

This paper is supported by Natural Science Foundation of China. (No.61272470, No. 61203307 and No. 61305087), the Provincial Natural Science Foundation of Hubei (No. 2013CFB320), supported by National Civil Aerospace Pre-research Project of China and the Fundamental Research Founds for National University, China University of Geosciences(Wuhan).

References

[1] G Liu, T Li, Y Peng amd X Hou, "The Ant Algorithm for Solving Robot Path Planning Problem", Proceedings of 3rd International Conf on Information Technology and Applications, vol. 2, **(2005)**, pp. 25-27.
[2] A. Yahja, S. Singh and A. Stentz, "An Efficient On-line Path Planner for Outdoor Mobile Robots", Robotics and Autonomous Systems, vol. 32, **(2000)**, pp. 129-143**.**
[3] U. Nehmzow and C.Owen, "Robot Navigation in the Real World: Experiments with Manchester's Forty Two in Unmodified Large Environments", Robotics and Autonomous Systems, vol. 33, **(2000)**, pp. 223-242.
[4] A. Howard and H.Seraji. "Vision-Based Terrain Characterization and Traversability Assessment", Journal of Robotic Systems, vol. 18, no.10, **(2001)**, pp. 577-587.
[5] D. B. Gennery, "Traversability Analysis and Path Planning for Planetary Rovers", Autonomous Robots, vol. 6, **(1999)**, pp. 131-146**.**
[6] J. Kennedy and R. C.Eberhart, "Particle Swarm Optimization", IEEE International Conference on Neural Networks, **(1995)**.
[7] M. Clare and J Kennedy, "The Particle Swarm - Explosion, Stability, and Convergence in a Multidimensional Complex Space", IEEE Trans. on Evolution2ary Computation, vol.6, no. 1, **(2002)**.
[8] C.A. Coello and M.S. Lechuga, Mopso, "A Proposal for Multiple Objective Particle Swarm Optimization, IEEE Proceedings World Congress on Computational Intelligence, **(2002).**
[9] J.Kennedy, "The Particle Swarm: Social Adaptation of Knowledge", Proc. IEEE int.Conf.on Evolutionary Computation, **(1997)**.
[10] E. Oscan and C. K.Mohan, "Analysis of A Simple Particle Swarm Optimization System", Intelligence Engineering Systems Through Artificial Neural Networks, **(1998)**, pp. 253-258.
[11] J. C. Latombe, "Robot Motion Planning", Kluwer Academic Publishers, Boston **(1998)**.
[12] M Bracho de Rodrýguez and J Ali Moreno, "Heuristic Algorithm for Robot Path Planning Based on Real Space Renormalization", Proceedings of the International Joint Conference, 7th Ibero-American Conference on AI: Advances in Artificial Intelligence Table of Contents, **(2000)**.
[13] J. Barraquand, B. Langlois and J. C. Latombe, "Numerical Potential Field Techniques for Robot Path Planning. IEEE Transactions on System", Man and Cybernetic, **(1992)**, pp. 224-241.
[14] R. Durbin and D. Willshaw, "An Anlaogue Approach to the Traveling Salesman Problem Using an Elastic Net Approach. Nature", vol 326, no. 6114, **(1987)**, pp. 689-691.
[15] X.S.Yan and Q. H. Wu, "A Robot Path Planning Algorithm Based on Evolutionary Algorithm", The Supplement Issue of Dynamics of Continuous, Discrete & Impulsive Systems, Series B: Applications & Algorithms,vol.14 (S4), **(2007)**, pp. 66-70.
[16] X.S.Yan and Q. H. Wu, "A Fast Evolutionary Algorithm for Robot Path Planning", Proceedings of the 2007 IEEE International Conference on Control and Automation, **(2007)**.
[17] X. S. Yan, Q. H. Wu, C. Y. Hu and Q. Z. Liang, "Circuit Design Based on Particle Swarm Optimization Algorithms", Key Engineering Materials, vols. 474-476, **(2011)**.
[18] X. Yan, Q. Wu and H. Liu, "An Improved Robot Path Planning Algorithm", TELKOMNIKA Indonesian Journal of Electrical Engineering, vol. 10, no. 8, **(2012)**, pp. 1948-1955.

Authors

Xuesong Yan received the MS degree in Computer Application Technology from China University of Geosciences in 2003, and the PhD degree in Computer Software and Theory from the School of Computer Science, Wuhan University in 2006. He is currently an associate professor in China University of Geosciences. His research interests are in the areas of evolutionary computation, evolvable hardware and machine learning.

Qinghua Wu received the MS degree in Computer Application Technology from China University of Geosciences in 2003, and the PhD degree in Earth Explore and Information Technology from China University of Geosciences, in 2011. She is currently a lecturer in Wuhan Institute of Technology. Her research interests are in the areas of evolutionary computation and image processing.

Chengyu Hu received the MS degree in Automation from Wuhan University of Technology in 2003, and the PhD degree in Automation from School of Mechanical Science and Engineering, Huazhong University of Science and Technology in 2010. He is currently an associate professor in China University of Geosciences. His research interests are in the areas of intelligence computation and automation.

Hong Yao received the MS degree in Computer application technolgoy from China University of Geosciences in 2004, and the PhD degree in Computer Architecture from School of Computer Science, Huazhong University of Science and Technology in 2009. He is currently an associate professor in China University of Geosciences. His research interests are in the areas of network architecture, computer applications and information systems.

Yuanyuan Fan received the MS degree in Computer software and theory from Wuhan University in 2004, and the PhD degree in Computer software and theory from School of Computer Science, Wuhan University in 2011. Her is currently an lecture in China University of Geosciences. Her research interests are in the areas of evolutionary computation and evolvable hardware.

Qingzhong Lian received the MS degree in Computer application technolgoy from China University of Geosciences in 2004, and now the PhD Candidate in China University of Geosciences. He is currently an lecture in China University of Geosciences. His research interests are in the areas of network architecture, computer applications and information systems.

Chao Liu received the MS degree in Computer application technolgoy from China University of Geosciences in 2004, and now the PhD Candidate in Computer Architecture from School of Computer Science, Huazhong University of Science and Technology. He is currently an lecture in China University of Geosciences. His research interests are in the areas of network architecture, computer applications and information systems.