International Conference on Mechanical, Industrial and Materials Engineering 2015 (ICMIME2015)
11-13 December, 2015, RUET, Rajshahi, Bangladesh.

Paper ID: AM-10

Robotic Path Planning Algorithm as a Tool of Visualization of Characteristic Differences amongParticle Swarm Optimization,Artificial Bee Colony Algorithm and Firefly Algorithm

Md. Tajmiruzzaman[1], Md. Al-Mamun[2]
[1,2]Department of Industrial & Production Engineering, Rajshahi University of Engineering & Technology, Rajshahi-6204, Bangladesh
E-mail:taju_ipe09@yahoo.com
Cell: 01717718097

Abstract

Particle Swarm Optimization (PSO), Artificial Bee Colony (ABC) Algorithm and Firefly Algorithm are meta-heuristic optimization algorithms based on the intelligent behaviors of bird, honeybee and firefly swarms respectively. Although these three algorithms can deal with any number of dimensions (variables), Robotic Path Planning uses only two or three variables. The usage of these algorithms in Robotic Path Planning has been demonstrated in numerous research works. But in this paper, contrarily, a simple Robotic Path Planning algorithm is developed to demonstrate and visualize the characteristic differences among the three optimization algorithms. The shape, length, consumed time and straightness of the paths created separately by these algorithms are visualized and compared. Finally, it is shown that beside the conventional usage of standard benchmark functions to test the accuracy and convergence speed of an optimization algorithm, the proposed Robotic Path Planning technique can be successfully used either as an alternative or at least as a supplementary tool.

Keywords: Robotic Path Planning, Particle Swarm Optimization, Artificial Bee Colony Algorithm, Firefly Algorithm.

1. Introduction

Whenever a new Optimization algorithm is developed, researchers have to test its performance for different situations. In most cases, standard unconstrained benchmark functions such asRosenbrock, Rastrigin, Griewank, Schwefel, Sphere, Ackley functions are used [1]. Thereafter the algorithm needs to be tested in constrained as well as other dynamic environments to ensure its robustness for use.Robotic Path Planning algorithms can provide these constrained and dynamic environments [2], and beside this, some other inner characteristics of the Optimization algorithm may be visualized in this way.

Among the two broad categories of bio-inspired algorithms (Evolutionary algorithms and Swarm-based algorithms), the latter includes Particle Swarm Optimization, Ant Colony Optimization, Artificial Bee Colony algorithm, Firefly algorithm, Cuckoo Search, Bat algorithm etc. The idea of swarm intelligence was first introduced by G. Beni, in 1989 [3]. In this paper, Particle Swarm Optimization, Artificial Bee Colony Algorithm and Firefly Algorithm are considered to be tested in our proposed way.

The construction of this paper is as follows. In section-2, the three Optimization algorithms are described concisely. In section-3, the proposed Robot path planning algorithm and other details of it are depicted. Section-4 demonstrates experimental settings, computer simulation details and results of the work. Finally, section-5 will conclude the paper.

2.Overview of Optimization algorithms

2.1 Particle Swarm Optimization

PSO is a stochastic optimization technique developed by Eberhart and Kennedy [4]in 1995. It is inspired by social behavior patterns of organism that are live and interact within large groups. It incorporates Swarming behaviors observed in flocks of birds, school of fish or swarm of bees and even in human social behavior. The idea of PSO technique is that particles move through the search space with velocities which are dynamically adjusted according to their historical behavior. Therefore, the particles have the tendency to move towards better search space in the course of their search process. Particle swarm optimization algorithm start with a group of random particles and searches for optima by updating each generation. Each

particle is considered as a volume-less particle in the n-dimensional search space. The ith particle is represented as $x_i = \{x_{i1}, x_{i2},.., x_{iD}\}$. At each generation, each particle is updated by two best values. The first one is the best solution the particle has achieved by itself so far. This is known as p_{best}. The second one is the best solution obtained so far by all particles in the population. This best value is known as g_{best}. At each of the iterations, these two best values are combined to adjust the velocity along each dimension and that velocity is used to compute a new movement for the particle with the help of equations given below:

$$v_i = w \cdot v_i + c_1 \cdot rand_1 \cdot (p_{besti} - x_i) + c_2 \cdot rand_2 \cdot (g_{best} - x_i) \quad (1)$$

$$x_i = x_i + v_i \quad (2)$$

Where, c_1 and c_2 are cognitive parameter and social parameter respectively; $rand_1$ and $rand_2$ are two random numbers generated in range [0,1].The use of w, called inertia weight was proposed by Shi and Eberhart [5], in1998.Applying a high inertia weight at the start of the algorithm and making it decay to a low value through the PSO algorithm execution, makes the algorithm globally search in the start of the search, and search locally at the end of the execution.

2.2 Artificial Bee Colony (ABC) Algorithm

Artificial Bee Colony algorithm is an optimization algorithm first introduced by D. Karaboga [6], [7]. In the ABC algorithm, there are three types of bees: employed bees, onlooker bees, and scout bees. The employed bees search food sources around the positions initially and randomly located by the algorithm, and meanwhile they share the information of these food sources to the onlooker bees. The onlooker bees select good food sources from those found by the employed bees. The food sources with higher nectar amount (fitness) will have a large chance to be selected by the onlooker bees than the one of lower quality. When a food source is abandoned, the employed bee associated with it becomes a scout bee. The whole population of bees is called a colony. The first half of the colony turns into employed bees and the second half into the onlooker bees. The number of employed bees or the onlooker bees is equal to the number of solutions (the number of food sources).

The ABC generates a random initial population of SN solutions (food source positions), where SN denotes population size. Let $X_i = \{x_{i1},x_{i2},\ldots,x_{iD}\}$ represent the ith solution in the population, where D is the dimension size. Each employed bee X_i generates a new candidate solution V_i in the neighborhood of its present position as follows:

$$v_{ij} = x_{ij} + rand \cdot (x_{ij} - x_{kj}) \quad (3)$$

Where, X_k is a randomly selected candidate solution ($i \neq k$),j is a random dimension index selected from the set {1,2,…,D} and 'rand' is a random number within [−1, 1]. Once the new candidate solution V_i is generated, a greedy selection between X_i and V_i is used. If the fitness value of $V_i$is better than that of its parent X_i, then update X_i with V_i; otherwise keep $X_i$unchanged.

After all the employed bees complete the search process, an onlooker bee evaluates the nectar information taken from all employed bees and chooses a food source with a probability related to its nectar amount.

$$p_i = \frac{fit_i}{\sum_{i=1}^{SN} fit_i} \quad (4)$$

Where, fit_i is the fitness value of the ith solution in the swarm. The better the solution i, the higher the probability of the ith food source to be selected. And then onlooker bees find newer food sources around the more probable sources.

Being an iterative process, ABC algorithm repeats the upper mentioned process until it reaches the termination condition. If a food source position (solution) remains unchanged over a predefined number of iterations (called 'limit'), then this position is replaced by a random food source found by a scout bee.

2.3 Firefly Algorithm

The Firefly algorithm was propounded by Dr. Xin She yang [8] at Cambridge University in 2007 which was inspired by the mating behavior of tropical fireflies.

Firefly algorithm is established on three idealized rules [9]:i) artificial fireflies are unisex sothat sex is not an issue for attraction; ii) attractiveness is proportional to their flashing brightness which decreases as the distance from the other firefly increases due to the fact that the air absorbs light. Since the most attractive firefly is the brightest one that convinces its neighbors to move toward itself, so ultimately all other fireflies will coincide

with the brightest firefly (best solution). In case of no brighter one, it freely moves any direction; and iii) the brightness of the flashing light can be considered as objective function to be optimized.
For an optimization problem, the light intensity or brightness I_i of a firefly i is theobjective function or fitness value $f(x_i)$ and the position of it, x_i is a candidate solution.
The main steps of the FA start from initializing a swarm of fireflies;the initial positions of fireflies are generated randomly ($x_i \in [x_{min}, x_{max}]^D$), and the light intensity of each of the swarm is evaluated. And now, a loop of pairwise comparisons of light intensities (fitness values) is executed, where the firefly with lower light intensity will move toward the higher one.
The movement of a firefly i,attracted by another brighter firefly j (which has better fitness value $f(x_j)$), is determined as follows:

$$x_i = x_i + \beta_{i,j} \cdot (x_j - x_i) + \alpha \cdot \left(rand - \frac{1}{2} \right) \cdot |x_{max} - x_{min}| \quad (5)$$

Where, the second term is the attraction (movement of position) of firefly i towards the brighter firefly j. But this attraction is compromised by the third term which adds a random movement to the current position ofi. It is worth pointing out that, this second term performs exploitation whereas the third term performs exploration here.$\beta_{i,j}$is attractiveness parameter; α is the randomization parameter with 'rand' being a random number generator uniformly distributed in [0, 1].$\beta_{i,j}$is determined as follows:

$$\beta_{i,j} = \beta_{max} \cdot e^{-\gamma r_{i,j}^2} \quad (6)$$

Where, β_{max} is the maximum attractiveness when distance r = 0, and γ is absorption coefficient (of the air)which controls the speed of the convergence. Thus, the attractiveness will vary with the distance $r_{i,j}$between firefly i and j.

$$r_{i,j} = \|x_i - x_j\| = \sqrt{\sum_{d=1}^{D} (x_{i,d} - x_{j,d})^2} \quad (7)$$

After moving, the new firefly is evaluated and updated for the light intensity again. During pairwise comparison loop, the best-so-far solution is iteratively updated. Thispairwise comparison process is repeated until termination criteria are satisfied.

3. Robotic path planning

Path planning problem can be stated as: "Given a mobile robot R which is free to move in a two dimensional space V which holds some obstacles (static as well as moving) O whose shapes and trajectories are known to prior, and also given a source position S and target position T, robot path planning is to find an optimal path from S to T without any collision with obstacles, using a proper algorithm."
Now the algorithm which does this task we call it as 'Robot Path Planning algorithm' or RPP algorithm. An RPP algorithm does the task in many ways; one of the best ways to do it is to exploit a befitting optimization algorithm as GA [10], ACO [11], PSO [12] and ABC [13].
What these optimization algorithms do is that, parallel with their converging to optimum solution, they advance the robot towards its goal point. Butthe intrinsic exploration or randomization of an optimization algorithm may also cause the robot to deviate from the course and to wander in unusual trajectories. For this, the mother RPP algorithm makes some adjustments and adds other accessories to the optimization algorithm to make it work in proper ways. It also has to make provisions for dealing with the probable future obstacles. Sonow you see that, a good RPP algorithm takes an optimization algorithm and furnishes it with a whole accouterment; and if we take these trappings out of it, then the RPP algorithm will show us the inherent characteristics of the optimization algorithm it uses, and you can visualize these properties by the two dimensional paths it creates.

3.1 Proposed approach

The position of the robot is represented by Cartesian co-ordinates such as x- and y-coordinate positions.
The main steps of the proposed algorithm are as follows:

Step 1: An optimization algorithm is chosento be visualized.
Step 2: An initial population of particles (bees or fireflies) are generated around the robot's start position and within its sensing range in a circle.
Step 3: Each particle assumes a new position by the execution of an iteration of the optimization algorithm.

Step 4: All the particles are checked if they lies inside any obstacle or if the lines connecting the new positions of the particles to the robot's current position intersect any obstacle; if any particle lies inside or intersects any obstacle, then the particle is relocated to another new position and checked again and again until it finds a feasible position.

Step 5: The best particle (i.e., the one nearest to the goal and most distant from all the sensed obstacles) is found accordingly with the requirement. Move the robot onto it and go to Step 3.

Step 6: Perform Steps 3–5 until the goal is within the robot's sensing range and can be accessed via a straight line.

Step 7: If the robot's position is stuck in a local optimum other than the goal point, re-initialize the particles around the robot's position and go to Step 3.

Step 8: Trace all the successive point positions of the robot and process for visualize.

3.2 Fitness Function

In the step-5 of the proposed approach, we see that the best particle among all others is to be chosen. So every particle in its current position has to be evaluated over its fitness for it to be selected. So something is needed to be used as evaluator of every particle; this evaluator may be called as a fitness function.The fitness value of ith particle is determined as follows:

$$F = w_g \cdot \sqrt{(x_i - x_g)^2 + (y_i - y_g)^2} - w_o \cdot \sum_{o=1}^{M} \sqrt{(x_i - x_o)^2 + (y_i - y_o)^2} \quad (8)$$

Where (x_i, y_i) is current position of ith particle and (x_g, y_g) is co-ordinate of goal point and (x_o, y_o) is co-ordinate of the sensed obstacle, where o = 1, 2, 3,.., M; M being the number of sensed obstacles.

The first term is for measuring the Euclidean distance between the particle and goal point; the second term is for the cumulative distance of the particle from all the sensed obstacles. w_g and w_o are two weighing factor to balance the two terms.

4. Experiments and results

All the experiments are executed in a Fujitsu LH531 computer and the configuration of PC was Intel(R) Pentium(R) CPU B960 @ 2.20 GHz and 2 GB RAM.As RPP uses a two dimensional space, it is needless to say that, the dimension used for all the three algorithms, D=2. All the experiments are performed in a space of 200x200 square units. In the objective function, w_g and w_o are taken as 1 and 0.2 respectively. Maximum generation was kept as 100 for all. Number and shapes of obstacles in all the environments are uniform. To make a dynamic environment, some positions were assignedto the obstacles initially, before starting iterations, and then the obstacles move randomly in a range of (-1,1) for both x and y co-ordinates in each iteration.

Table 1. Length of the paths generated by PSO, ABC and FA

Particle No.	PSO			ABC			FA		
	Best	Worst	Mean	Best	Worst	Mean	Best	Worst	Mean
100	278.24	282.59	280.032	292.71	321.29	299.58	272.30	279.12	276.42
200	272.70	280.63	276.10	279.43	290.12	285.21	269.16	276.32	273.29
300	272.03	278.25	274.64	277.75	287.85	282.55	269.67	274.39	272.06
375	271.34	276.48	273.37	276.47	285.72	280.27	268.82	273.24	271.63
450	271.02	276.20	272.03	275.31	284.92	278.81	268.47	273.22	270.89

From Table 1andFig. 1(a) through Fig. 3(b), the larger are the particle (candidate solutions) numbers, the more the paths generated straighten. As the paths created by ABC algorithm are most curled up, it can be said that the algorithm has a larger exploration tendency and it lacks on convergence speed. The Firefly algorithm based pathis the most straightened all the time, so it can be assumed as the most converging algorithm but it is the most time consuming, as we can see in Table 2. PSO performs well and almost parallel with Firefly algorithm. It

should also be noticed that, all the algorithms succeed in making the robot reach the goal point after all, so they are all reliably usable in practical field.

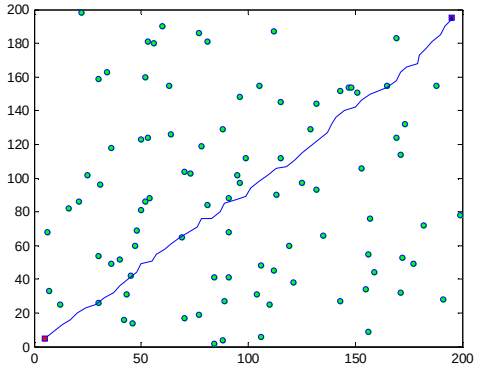

Fig. 1(a). Path by PSO for 100 particle

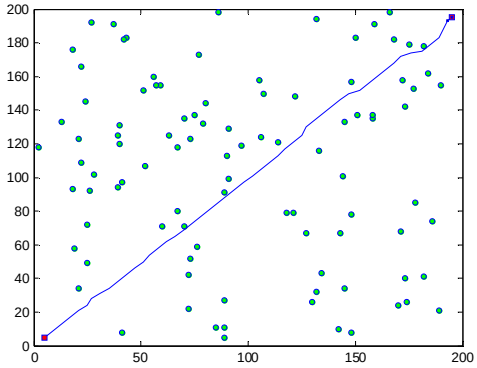

Fig. 1(b). Path by PSO for 300 particle

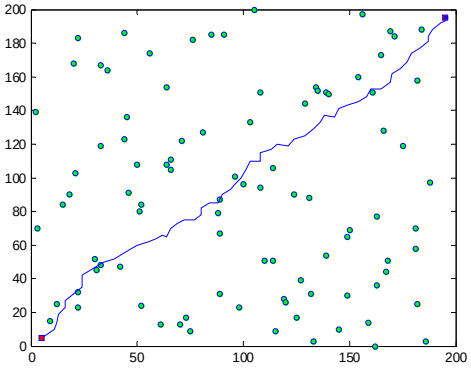

Fig. 2(a). Path by ABC for 100 particle

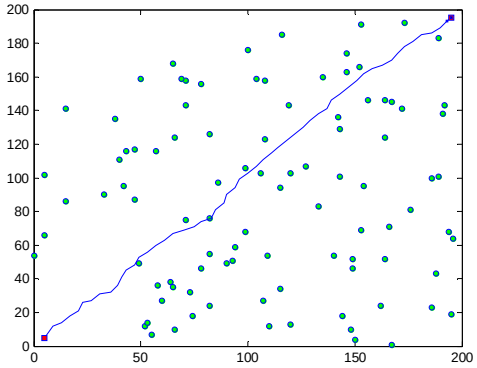

Fig. 2(b). Path by ABC for 300 particle

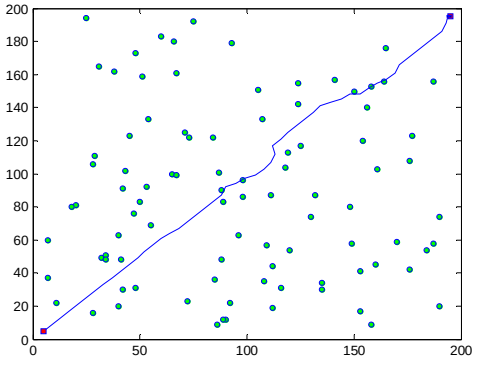

Fig. 3(a). Path by FA for 100 particle

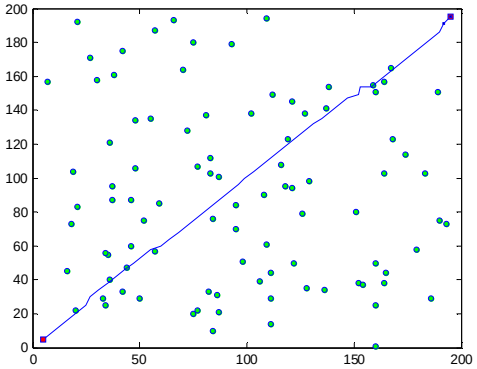

Fig. 3(b). Path by FA for 300 particle

Table 2. Processing time for the paths generated by PSO, ABC and FA

Particle No.	PSO			ABC			FA		
	Best	**Worst**	**Mean**	**Best**	**Worst**	**Mean**	**Best**	**Worst**	**Mean**
100	40.56	49.18	43.64	51.21	62.55	56.41	41.87	61.06	45.90
200	38.55	53.91	41.52	42.17	55.43	46.041	54.69	60.56	56.98
300	38.25	46.84	40.99	40.84	54.98	43.83	77.77	82.86	80.04
375	37.40	47.09	40.56	38.91	54.77	42.34	93.64	102.29	97.85
450	36.82	45.61	40.12	37.73	53.83	41.27	112.75	119.68	116.43

5. Conclusion

This work demonstrates just a new way to use the idea of robotic path planning. As in these cases: when the path is unnecessarily lengthy or roundabout, it can be assumed that the optimization algorithm has an excessive emphasis on exploration and has a lack on convergence capability; if the path is stuck on some point in the space and cannot be further extended despite the particle number being sufficiently large, then the optimization algorithm may be assumed as having a deficiency on exploration: the future researchers can use RPP algorithms in other advantageous ways as this work uses.

6. References

[1] Md. Tajmiruzzaman, Md. Asadujjaman,"Artificial Bee Colony, Firefly and Bat Algorithm in Unconstrained Optimization",*Proc. ofICMIEE,*Paper ID. ICMIEE-PI-140169, pp. 1-6, 2014.

[2] Md. Rakibul Islam, Md. Tajmiruzzaman, Md. MahfizulHaqueMuftee, Md. SanowarHossain, "Autonomous Robot Path Planning Using Particle Swarm Optimization in Dynamic Environment with Mobile Obstacles & Multiple Target", *Proc. of ICMIEE*, Paper ID. ICMIEE-PI-140282, pp. 1-6, 2014.

[3] G. Beni, J. Wang, "Swarm Intelligence in Cellular Robotic Systems", *Proc. of NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy*, June 26–30, 1989.NATO ASI Series, Vol. 102, pp. 703-712, 1993.

[4] R. Eberhart and J. Kennedy, "A New Optimizer using Particle Swarm Theory", *Proc. of the Sixth International Symposium on Micro Machine and Human Science*, pp. 39-43, 1995.

[5] Y. Shi, and R. C. Eberhart, "Parameter Selection in Particle Swarm Optimizer", *Proc. of Seventh Annual Conference on Evolutionary Programming, Berlin: Springer-Verlag*, pp. 591-601, 1998.

[6] D. Karaboga, "An Idea Based on Honey Bee Swarm for Numerical Optimization", *Technical Report-TR06, Erciyes University, Engineering Faculty, Computer Engineering Department*, 2005.

[7] D. Karaboga, B. Basturk, "A Powerful and Efficient Algorithm for Numerical Function Optimization: Artificial Bee Colony (ABC) Algorithm", *Journal of Global Optimization*, Vol.39, pp. 459-471, 2007.

[8] X. S. Yang, "Firefly Algorithm, Stochastic Test Functions and Design Optimization", *International Journal of Bio-Inspired Computation*, Vol. 2, Paper No. 2, pp.78–84, 2010.

[9] X. S. Yang, "Nature-Inspired Metaheuristic Algorithms", *Luniver Press*, Second edition, pp. 81-96,2010.

[10] J. Solano, D.I. Jones, "Generation of Collision-free Paths by a Genetic Approach", *IEEE Colloquium on Genetic Algorithm for Control SystemEngineering*, pp. 5/1-5/6, 1993.

[11] H. Ying Tung, C. Cheng Long, C. Cheng Chih, "Ant Colony Optimization for Best Path Planning", *Proc. of IEEE/ISCIT*, Paper No.04, pp. 109-113, 2004.

[12] G. Z. Tan, G. J. Liu,"Global Optimal Path Planning for Mobile Robots Based on Particle Swarm Optimization", *Proc. ofApplications Research of Computers*, Vol. 24, Paper No.11, pp. 210-212, 2007.

[13] Qianzhi Ma,Xiujuan Lei, "Dynamic Path Planning of Mobile Robots Based on ABC Algorithm",*Lecture Notes in Computer Science,*Vol. 6320, pp. 267-274, 2010.