Al-Khwarizmi Engineering Journal,Vol. 12, No. 4, P.P. 26- 35 (2016)

Al-Khwarizmi Engineering Journal

Path Planning of an Autonomous Mobile Robot using Enhanced Bacterial Foraging Optimization Algorithm

Nizar Hadi Abbas* **Farah Mahdi Ali ****

*,***Department of Electrical Engineering/ College of Engineering/ University of Baghdad*

*Email: drnizaralmsaodi@gmail.com

**Email: farah95m@yahoo.com

(Received 20 September 2015; accepted 31 January 2016)

http://dx.doi.org/10.22153/kej.2016.01.001

Abstract

This paper describes the problem of online autonomous mobile robot path planning, which is consisted of finding optimal paths or trajectories for an autonomous mobile robot from a starting point to a destination across a flat map of a terrain, represented by a 2-D workspace. An enhanced algorithm for solving the problem of path planning using Bacterial Foraging Optimization algorithm is presented. This nature-inspired metaheuristic algorithm, which imitates the foraging behavior of E-coli bacteria, was used to find the optimal path from a starting point to a target point. The proposed algorithm was demonstrated by simulations in both static and dynamic different environments. A comparative study was evaluated between the developed algorithm and other two state-of-the-art algorithms. This study showed that the proposed method is effective and produces trajectories with satisfactory results.

Keywords: *Autonomous Mobile Robot, Path Planning, Static and Dynamic Environments, Bacterial Foraging Optimization Algorithm.*

1. Introduction

The field robot path planning was launched at the middle of the 1960s. Robot path planning is an important problem in navigation of mobile robots. The aim is to find an optimal and collision-free path from a predefined start position to a target point in a given environment. Generally, there are many paths for robot to reach the target, but in fact, the best path is selected according to some guideline. These guidelines are: shortest distance, least energy consuming or shortest time with the shortest distance is the most adopted criteria [1]. Path planning can be seen as an optimization problem since its purpose is to search for a path with *shortest distance* under certain constraints such as the given environment with *collision-free* motion [2]. In the past several decades, research on optimization algorithms has covered a wide area of researchers' attention. Optimization methods and algorithms can be classified in many types, but the simplest way is to look at the nature of the algorithm, and this grouped them into deterministic and stochastic [3, 4] where deterministic techniques depend on the mathematical nature of the problem, while stochastic techniques do not depend on the mathematical properties of a given function and are hence more appropriate for finding the global optimal solutions for any type of objective function. However, the weaknesses of first technique are its dependence on gradient, local optima and inefficient in large-scale search space and cannot solve discrete functions. Stochastic techniques are considered to be more users friendly. As many real-world optimization problems become increasingly complex, using stochastic methods is inevitable. These algorithms have been found to perform better than the classical or gradient-based methods, especially for optimizing the non-differentiable, multimodal, and discrete complex functions. Some effective

stochastic techniques that mimic the behaviors of certain animals or insects (birds, ants, bees, flies and even germs!) and called Nature-Inspired Algorithms have been developed since 1980s. Currently, these nature-inspired paradigms have already come to be widely used in many areas in engineering fields [5, 6]. Some of these algorithms are Particle Swarm Optimization (PSO) [7], Ant Colony Optimization (ACO) [8], Artificial Bee Colony (ABC) [9] and Bacteria Foraging Optimization (BFO) [10].

The Bacteria Foraging Optimization (BFO) optimization is one of the most-recent population based (swarm intelligence based) meta-heuristic algorithms, which simulate the foraging behavior of E- coli colonies. It proposed by K. M. Passino [10] in 2002.BFO is a simple but powerfulbio-inspired optimization technique uses the analogy of swarming principles and social behavior in nature - swarm intelligence- and it have been adopted to solve a variety of engineering and mobile robotics problems, including path planning problem [11].

In this paper, an enhanced version of the BFO algorithm called Adaptive Tumble BFO (ATBFO) is proposed. A method for recording the best positions achieved by the bacteria so far and saving those directions as a guide to better potential candidate solutions is proposed.

The rest of this paper is organized as follows: section 2 describes path planning and problem formulation; Section 3 describes the standard BFO algorithm; Section 4 describes the proposed ATBFO algorithm; Section 5 explains robot path planning using ATBFO, and the simulation results are shown in section 6. Finally, section 7 presents the conclusion of the paper.

2. Path Planning and Problem Formulation

Robot Path planning (RPP) is one of the important aspects in robot navigation research. Depending on the environment where the robot located in; RPP can be classified into two types:

1) RPP in static environment which has fixed obstacles.

2) RPP in dynamic environment which has both fixed and moving obstacles.

Each of these two types could be further subdivided into a sub-group:

1) Global Path Planning (GPP): if the knowledge of the environment is known, the global path can be planned *offline* before the robot starts to moves.

2) Local Path Planning (LPP): is usually constructed *online* when the robot avoids the obstacles in a real time environment [12].

In this paper, local path planning is adopted where the environment is totally unknown. Constructing a model for the environment is important issue; an appropriate representation of the terrain is needed to generate a sufficiently complete map of the given surroundings that the robot will encounter along its route. The mobile robot is defined as a point object in the 2-D space. Since the robot reduced to a point each obstacle must be inflated by the size of the robot's radius to compensate. Each obstacle can be represented by polygon surrounded by a circle. Obstacles are finite in size and do not overlap and have a *safety zone* which is the region around the obstacle that the mobile robot must avoid. As the obstacle is of an irregular shape, the radius must be one-half of the longest side of the obstacle plus the robot's radius.

3. Standard BFO Algorithm

Bacteria Foraging Optimization (BFO) algorithm invented by K. M. Passino [10] is a relatively new population-based algorithm; it is a nature-inspired metaheuristic algorithm, which imitates the foraging behavior of E. coli. An E. coli bacterium can move in two different ways; it can run (swim for a period of time), or it can tumble, and it alternates between these two modes of operation through its entire lifetime. By tumbling and running, the bacteria will search for nutrient area and keep away from the poisonous area, as shown in Fig. 1.

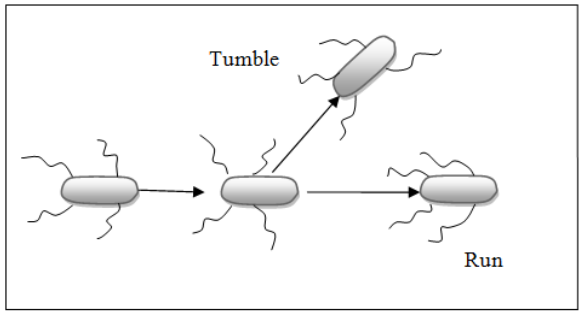

Fig. 1. Chemotactic behavior of E. coli: run and tumble

3.1. Chemotaxis

Chemotaxis is the main motivation of the bacteria's foraging process. It consists of a tumble

with several runs as shown in Fig. 1. In BFO, the position updating, which simulates the chemotaxis procedure is used in Eq. (1). θ_i^jrepresents the position of the ith bacterium in the jth chemotaxis step, C(i)is the step length during the jith chemotaxis and∅(i)is a unit vector which stands for the swimming direction after a tumble. It can be generated by Eq. (2), where Δ_i is a randomly produced vector with the same dimension of the problem:

$$\theta_i^{j+1} = \theta_i^j + C(i).\emptyset(i) \quad \ldots(1)$$

$$\emptyset(i) = \frac{\Delta_i}{\sqrt{\Delta_i^T \Delta_i}} \quad \ldots(2)$$

In each chemotactic step, the bacterium generated a tumble direction firstly. Then the bacterium moves in the direction using Eq. (1). If the nutrient concentration in the new position is higher than the last position, it will run one more step in the same direction. This procedure continues until the nutrient get worse or the maximum run step is reached. The maximum run step is controlled by a parameter calledNs.

3.2. Reproduction

For every Nc time of chemotactic steps, a reproduction step is taken in the bacteria population. The bacteria are sorted in descending order by their nutrient obtained in the previous chemotactic processes. Bacteria in the first half of the population are regarded as having obtained sufficient nutrients so that they will reproduce. Each of them splits into two (duplicate one copy in the same location). Bacteria in the residual half of the population die, and they are removed out from the population. The population size remains the same after this procedure. Reproduction is the simulation of the natural reproduction phenomenon. By this operation, individuals with higher nutrient are survived and duplicated, which guarantees that the potential optimal areas are searched more carefully.

3.3. Elimination and Dispersal

In nature, the changes of environment where population lives may affect the behaviors of the population. For example, the sudden change of temperature, nutrient concentration and the flow of water. All these may cause bacteria in the population to die or move to another place. To simulate this phenomenon, eliminate-dispersal is added in the BFO algorithm. After every Nre time of reproduction steps, an eliminate-dispersal event happens. For each bacterium, a random number is generated between 0 and 1. If the random number is less than a predetermined parameter, known asPe, the bacterium will be eliminated, and a new bacterium is generated in the environment. The operator can be also regarded as moving the bacterium to a randomly produced position. The eliminate-dispersal events may destroy the chemotactic progress. However, they may also promote the solutions since dispersal might place the bacteria in better positions [13]. Over all, contrary to the reproduction, this operator enhances the diversity of the algorithm. In BFO algorithm, the eliminate-dispersal events happen for Nedtimes.

4. Adaptive Tumble BFO (ATBFO) Algorithm

BFO possesses a poor convergence behavior over complex optimization problems. Subsequently, a method for speed up the searching process is needed. The tumble angles (Δi) in the chemotactic phase are generated randomly. As a result, the algorithm is more like a random searching algorithm except it will try in better directions[14]. A method for recording the best positions achieved by the bacteria so far and saving those directions as a guide to better potential candidate areas is proposed. Firstly, tumble angles are generated randomly, then each bacterium counts its health improvement if it is larger than predefine constant parameter (F_{improv}), it will save this tumble angle as an input to the next reproduction in the foraging process. That is, the bacterial join resources uncovered by other bacteria in previous chemotactic step. Afterwards, they start exploiting the neighborhood of these current positions until the needed criterion (i.e., the feedback from the search process) is reached. The saved tumble angles would help each bacterium for evaluate more precise solutions.

The pseudo code of ATBFO algorithm is given in Pseudo code 1. Where S is the colony size, i is the bacterium's ID counter from 1 to S, Xi is the ith bacterium's position of the colony, Ns is the maximum number of steps for a single activity of swim, imp_{count} is a counter to determine the fitness improvement for the bacteria positions, F_{improv} is a user defined threshold of the improvement count required, $\Delta_{initial}$ is the initial directions that generated randomly and Δ_i^j , Δ_i^{j+1}

are current and next saved tumble angles respectively.

Algorithm 1: Adaptive Tumble Bacterial Foraging Optimization Algorithm

Step 1: Initialize parameters
Ned , Nre , Nc, Ns, Xiand $\Delta_{\mathbf{initial}} = \mathbf{rand}(\mathbf{S})$

Step 2: If $j < Nc, j = j + 1$**. For each bacterium, do the following processes:**
Tumble: Generating a random direction using Eq. (2) then make a move by Eq. (1) and calculate the fitness value at current position.
Run: When $m < Ns, m = m + 1$.. Calculating the fitness value. If better than last step, keep running until $m = Ns, imp_{count} = imp_{count} + 1$ otherwise go to Step 3.
Step 3: If $imp_{count} > F_{improv}$, $\Delta_i^{j+1} = \Delta_i^j$ **otherwise**
$\Delta_i^{j+1} = \Delta_{initial}^{j}$.
Step 4: If $k < Nre, k = k + 1$, start reproduction go to Step 2, otherwise go to Step 5.
Step 5: If $l < Ned, l = l + 1$, start elimination-dispersal step go to Step 2, otherwise end.

5. Robot Path Planning Using ATBFO

Despite classical methods' drawback such as trapped into local minima and high time complexity in high dimensional problems, they have very simple structure which makes them easy to implement. Classic artificial potential field (APF) method was originally introduced by Khatib [15], as a real time obstacle avoidance method. APF is particularly attractive and has been widely used for path planning related problems for more than two decades because of its elegant mathematical analysis and simplicity [16]. However it suffers from many drawbacks [17]: trap situations due to local minima, no passage between closely spaced obstacles, oscillations in the presence of obstacles and oscillations in narrow passages. In other hand BFO is a simple and effective searching algorithm. Consequently a method for hybridizing APF with BFO as LPP technique (making use of both methods' advantages), is introduced in this study.

As stated before the environment would be 2D work space with circular obstacles. The mobile robot is represented as a dot by Cartesian coordinates (x, y) in the *xy*-plan. A description for the sensor is needed since the robot is working in totally unknown environment. In this work range based obstacle detection is done using infrared sensor (IR). Five IR sensors are positioned in the mobile robot (R) each with 1.2 meter range. These sensors are located in five angles: 0, 45, 90,135 and 180 Fig. 2.

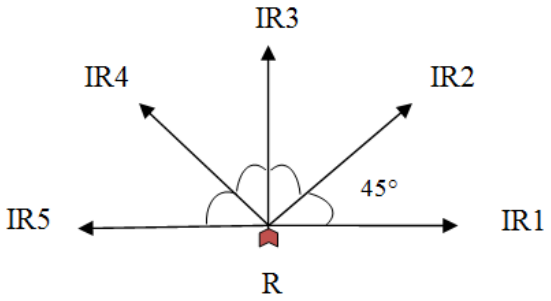

Fig. 2. Robot model.

In local path planning, finding a path is done on line by only the available readings from the sensors. A function for guiding the robot toward the target is needed. In APF method, two cost functions (attractive and repulsive) are used to attract the robot to the target and push it away from any obstacles at the same time. A combination of both forces drives the robot to its final destination. In this work both attractive and repulsive potential functions *(not their gradients forces)* are hybridized with BFO to drive the mobile robot. A goal cost function similar to APF [18] is introduced as in Eq. (3) and Eq. (4).

$$J_{goal} = \frac{1}{2} * \xi * \|P_R - P_T\|^2 \quad \ldots(3)$$

$$\|P_R - P_T\| = \sqrt{(x_R - x_T)^2 + (y_R - y_T)^2} \quad \ldots(4)$$

Where ξ is a positive constant scaling factor, $\|P_R - P_T\|$ is the Euclidean distance between the robot (P_R) and the target (P_T). For the obstacle cost function, a repulsive function is assigned for each detected obstacle. As shown in Eq. (5).

$$J_{obs} = \begin{cases} \frac{1}{2} * \eta * \left(\frac{1}{d(R)} - \frac{1}{do}\right) & \text{if } d(R) \le D_{sense} \\ 0 & \text{if } d(R) > D_{sense} \end{cases} \quad \ldots(5)$$

Where , η is a positive constant scaling factor, d(R) is the distance from robot to the obstacle, do is a positive constant representing the influence distance of the obstacle and D_{sense} is the sensing range. The final function that would guide the robot is the sum of both functions; as shown in Eq. (6).

$$J_{Total} = J_{goal} + J_{obs} \quad \ldots(6)$$

The main target of the BFO algorithm is to find the minimum $J(\theta)$, $\theta \in R^2$ without considering the gradient, $\nabla J(\theta)$, where θ is the position of a

bacterium, and $J(\theta)$ is cost of goal plus obstacle functions from the environment. $J(\theta) < 0$, $J(\theta) = 0$, and $J(\theta) > 0$ represent that the bacterium at location$\theta$is in nutrient rich, neutral, and noxious environments respectively. Each bacterium tries to climb up the nutrient concentration (i.e., lower and lower values of $J(\theta)$) and avoids being at positions θ where $J(\theta) \geq 0$. According to BFO algorithm,ith bacterium at the position θ takes a chemotaxis step j with the step size $C(i)$ in the random direction and calculates the cost function $J(\theta)$ at each step. If the cost function of the new position θ_i^{j+1}that is, (θ_i^{j+1}), is smaller than the (θ_i^{j}), then another step size $C(i)$ in the same previous direction will be taken. This process in the direction of lower cost function will be continued until the maximum number of steps (Ns) is reached or until bacteria enter a poisonous area. Thereafter, in eachNc chemotaxis step, the least healthy bacteria (lowest or second half of the population) as stated by the cost function ($J\theta_i^{j}$) are replaced by the copies of healthy ones (highest or first half of the population). This procedure is called reproduction step, and it is followed by the elimination–dispersal (Ned) event. For each elimination-dispersal event, the bacterium in the population is subjected to elimination dispersal with probabilityPed.

Initially the bacteria are randomly generated and distributed in front of the Mobile Robot (MR). The angles in which the bacteria are distributed equal to $45°$ and $-45°$ with respect to the target. The range of distribution must equal or smaller than sensing range (equal or smaller than 1.2 m). These bacteria are searched for the optimal path toward the target position while avoiding the obstacles by helping the robot's sensors. Then the healthiest one which has found the smallest $J(\theta)$ among other ones is chosen and the mobile robot goes to this position. This process will continue until the target is reached. For choosing the best becterium, cost function $J(\theta)$, $\theta \in R^2$is tested and compared, which is an attractant-repellent function from the environment.

During the algorithm, whenever the MR detects an obstacle within its sensor range, it assigns a value to J_{obs} (Eq. (5)) otherwise its value is zero. In other words, a goal cost function J_{goal} (Eq. (3)) is assigned to target position throughout the mission. The total cost function which is the sum of both functions J_{Total}(Eq. (6)) represents the cost function $J(\theta_i)$ defined previously. In order to make a decision for which bacterium would be chosen as the next position that the MR headed to, a difference cost function is defined by:

$$d_i^j = J(\theta_i(t+1)) - J(\theta_i(t)) \qquad i = 1,2,\dots S \quad \dots (7)$$

After that, the robot position is updated from its current location $u(t)$ to its next location $u(t+1)$ (pick best direction) and move one-step length equal to lambda (λ) as follows:

$$\phi_{best} = \tan^{-1}\frac{{\theta_{best}}^y(t+1) - {\theta_i}^y(t)}{{\theta_{best}}^x(t+1) - {\theta_i}^x(t)} \quad \dots (8)$$

$$u(t^x+1) = u(t^x) + \lambda * \cos\phi_{best} \quad \dots (9)$$

$$u(t^y+1) = u(t^y) + \lambda * \sin\phi_{best} \quad \dots (10)$$

Where ϕ_{best} is the angle between current bacteria coordinate $({\theta_i}^x(t), {\theta_i}^y(t))$ and best next bacteria coordinate $({\theta_{best}}^x(t+1), {\theta_{best}}^y(t+1))$ and $(u(t^x+1), u(t^y+1))$ is MR next step point coordinates that it should headed toward smoothly.

6. Simulation Results

In all simulation arenas, the population size S=100, number of chemotactic steps $Nc = 8$, maximum number of steps that a bacterium can swim $Ns = 6$, number of reproductions $Nre = 2$, number of elimination-dispersals events $Ned = 1$, elimination-dispersal probability $ped = 0.15$ and speed of the movement taken in one step $C(i) = 0.1, i = 1,2,\dots S, \xi = 0.1, \eta = 0.06$, do = 1.5, step length lambda$\lambda = 0.2$, $F_{improv} = 3$,$D_{sense} =$ 1.2m. All experiments are achieved the following solutions after executing the algorithm ten times using MATLAB R2011b programming language. The MATLAB codes are run on a computer system with 2.13 GHz Core i3 CPU, and 2 G RAM.

6.1 Case study 1: Environment with 4 obstacles

In case study 1 four obstacles with equal size are situated in the environment. All obstacles' positions are listed in Table 1. The cost function representing goal plus obstacles are illustrated in Fig. 3 and Fig. 4.

Table 1,
Obstacles definition for case 1.

Obstacle	Radius	Center(X,Y)
1	0.7	(2,2)
2	0.7	(4,4.1)
3	0.7	(7.2,7)
4	0.7	(9,8)

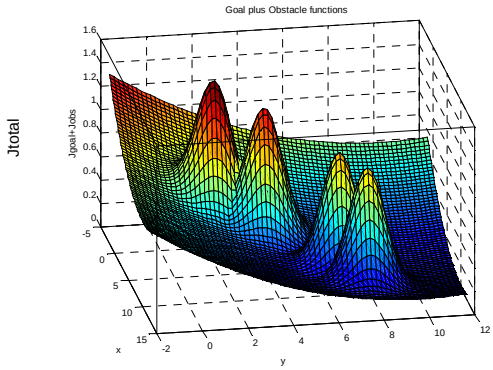

Fig. 3. Cost function used for case 1.

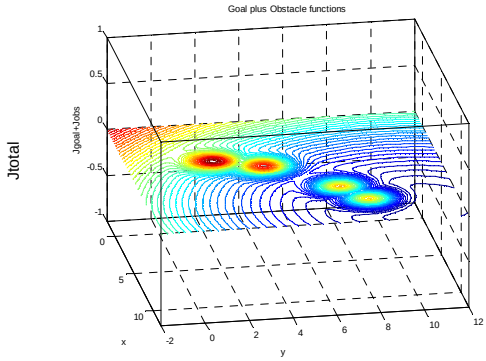

Fig. 4. Contour plot forcost function for case 1.

The best path with shortest distance using ATBFO algorithm is equal to 14.5346 as illustrated in Fig. 5.A second scenario is tested by moving the first obstacle from point (3, 2) to point (2, 2). As stated by [19] the APF fail to reach the target due to local minimum problem. The proposed algorithms were able to overcome this drawback or difficulty as shown in Fig.6 with total distance equal to 14.5412. The proposed RPO algorithm in [19] achieved a path with total run time equal to (11.0321s).

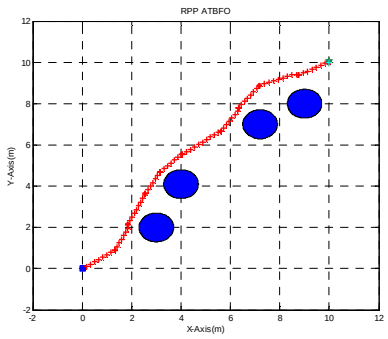

Fig. 5. Best shortest path for case 1.

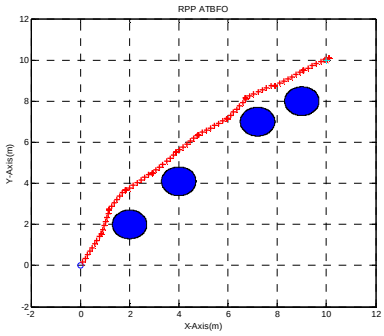

Fig. 6. Best shortest path (2nd scenario) for case 1.

6.2 Case study 2: Environment with 12 obstacles

In case study 2, a complex environment with twelve obstacles in different sizes is presented. All obstacles' positions (center and radius) are listed in Table 2. The cost function representing goal plus obstacles are illustrated in Fig. 7 and Fig. 8. The best path with shortest distance using ATBFO algorithm is equal to 14.4797as illustrated in Fig. 9, while thebest path with shortest execution time is 8.274537s as in Fig. 10. The New Bacteria Colony Approach (NBCA) with variable velocity and GA in [11] as GPP algorithms are compared with the proposed LPP algorithms.It achieved path equal to 14.40351 (after divided by scale = 10) and time 62.14s. Best result achieved by GA with path of 14.40982 (after divided by scale = 10) and time 34.70 s.

In order to investigate the effect of the step size (C(i)) on the overall algorithms, the whole test is repeated with smaller step length $C(i) = 0.05$. The best two paths according to distance and run time are illustrated in Fig. 11 (total distance =14.7852) and Fig. 12 (run time =11.628703s).

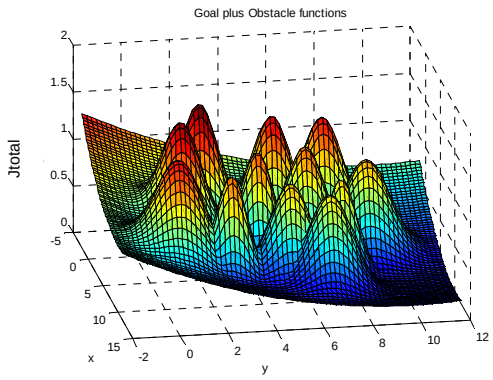

Fig. 7. Cost function for case 2.

Table 2,
Obstacles definitions for case 2.

Obstacle	Radius	Center (X,Y)
1	1	(1.3,2.5)
2	.8	(1,7.6)
3	.5	(7.6,.9)
4	1.3	(4.5,4.5)
5	.9	(1.2,5.5)
6	1.4	(8,3)
7	1.2	(6.6,7.7)
8	.8	(3.2,1.5)
9	.7	(7.5,5.5)
10	.6	(8.7,7)
11	.8	(3.5,6.6)
12	.5	(4.5,9)

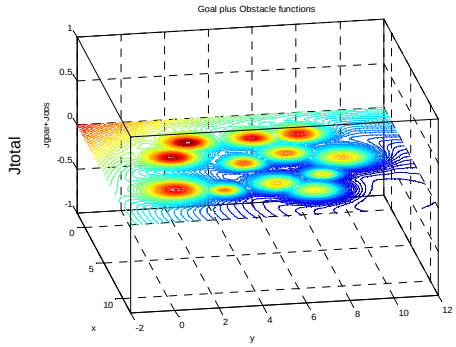

Fig. 8. Contour plot forCost function for case 2.

Although large step size speeds up the algorithm and yields to fast convergence, it leads to less smoother paths.

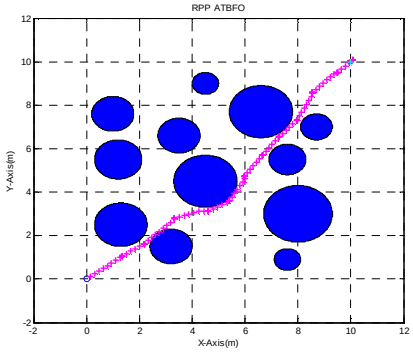

Fig. 9. Best shortest path for case 2.

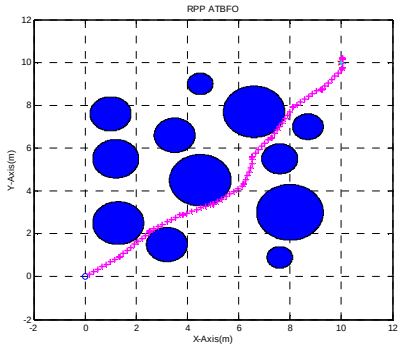

Fig. 9. Best shortest path for case 2.

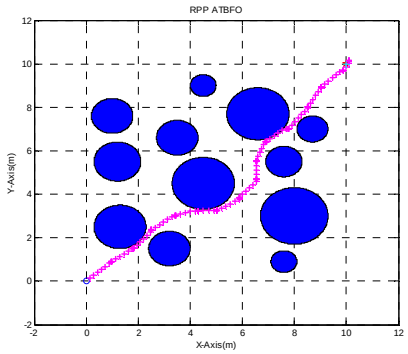

Fig. 11. Best shortest path for case 2(2nd test).

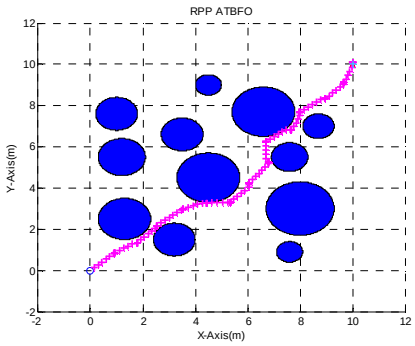

Fig. 12: shortest time path for case 2 (2nd test).

6.3 Case study 3: Environment with 6 moving Obstacles

A dynamic environment with six moving obstacles with equal sizes is presented. All obstacles' *initial* positions (center and radius) are listed in Table 3.

Table 3,
Obstacles definitions for case 3.

Obstacle No.	Radius	Center(X,Y)
1	.4	(1.5,1)
2	.4	(3,3)
3	.4	(5.5,4.1)
4	.4	(6,5)
5	.4	(8,6)
6	.4	(11,10)

Only the original BFO is hybridized with APF. Since the next best location cannot be predicted due to obstacles movement, the calculation for fitness improvement toward best next position is pointless. In all scenarios three obstacles move linearly and the other three circulate in a path with radius equal to 1. The resultant path for the first scenario can be shown in Fig. 13 which takes 10.5203 seconds.

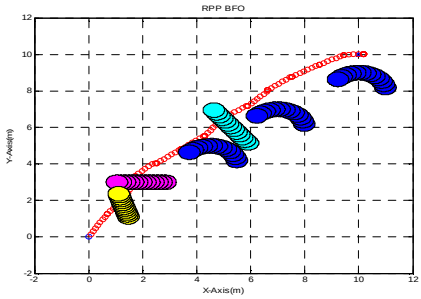

Fig. 13. 1st scenario pathfor case3.

In the second scenario one of the obstacles circulates around the target. The path generated takes 11.5450 seconds and is shown in Fig. 14.

Finally, in this scenario in addition to the six moving obstacles, the target represented by green stars, rotates around center point (9.5, 9.5) with radius equal to 1. The elapsed time was 11.6911 seconds and is shown in Fig. 15.

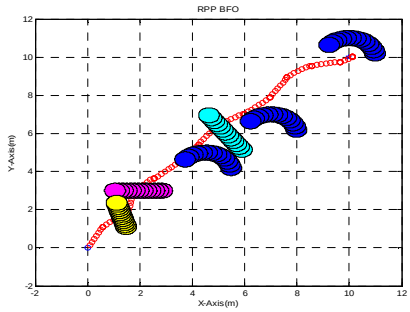

Fig. 14. 2nd scenario pathfor case3.

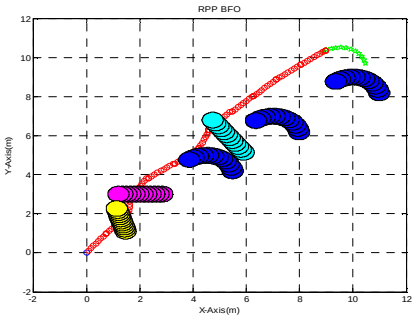

Fig. 15: 3rd scenario pathfor case3.

The proposed Random Particle Optimization (RPO) in [19] is compared with the BFO-APF method. The proposed algorithm was able to reach feasible paths and avoid been trapped in local minima situation in three different scenarios.

7. Conclusion

In this paper, a mobile robot local path planning model based on ATBFO is developed. The proposed algorithm models the environment using APF method through two contradictory forces: attractive force for the goal and repulsive force for the obstacles. The ATBFO algorithm examines negative feedback from the algorithm to choose appropriate direction vectors (Δ_i) that guide the search process to promising area with a better local search.

In order to investigate the influence of different step size on the RPP problem, two-step sizes were tested. The acquired simulation results show that larger step size speed up the search process but lead to an unfavorable path (less smoothness). Conversely, small step size may slow down the algorithm, but it achieves more suitable paths when comes to get smoother ones. Three case studies are adopted to evaluate the performance of the proposed algorithm. Its performance has been compared with some state-of-the-art algorithms. The ATBFO algorithm achieved feasible paths with relatively fast speed range from 2.3m/s for simple environment to 1.85m/s for crowded environment case making it reliable and efficient in practice.

List of Symbols

Symbol	Definition
Δ_i	Random direction
η	Positive scaling factor
λ	Robot step length
$\emptyset_*$	uniformly distributed number between $[-1,1]$
ϕ	Best picked direction
θ	Bacterium position
ξ	Positive scaling factor
C	Step size
d	Difference cost function
d	Positive constant for obstacles
d(R)	distance from the robot to the obstacle
D	Sensing range
F	Fitness improvement threshold
J_g	Health goal function
J_o	Health obstacle function
J_t	Health total function
N	Chemotactic no.
N	Elimination and dispersal no.
N	Reproduction no.
N	Swim counter
P	Elimination parameter
P	Robot position
P	Target position

S Bacteria colony size

u Robot position at time t

8. References

[1] N. H. Abbas and F. M. Ali, "Path Planning of an Autonomous Mobile Robot using Directed Artificial Bee Colony Algorithm,"International Journal of Computer Applications,vol. 96, pp.11-16. June 2014.

[2] P. A.M. Ehlert, "The use of Artificial Intelligence Robots,"Report on research project, Delft University of Technology, Netherlands, October 1999.

[3] C. A. Floudas, "Deterministic Global Optimization: Theory, Methods and Applications,"Nonconvex Optimization and Its Applications, Kluwer Academic Publishers, Dordrecht, The Netherlands, 2000.

[4] J. C. Spall, "Introduction to Stochastic Search and Optimization: Estimation, Simulationand Control,"Wiley-Interscience Series in Discrete Mathematics and Optimization,Wiley-Interscience,Hoboken,NJ, USA, 2003.

[5] X. Yang, " Nature-Inspired Metaheuristic Algorithms," 2nd Edition, by Luniver Press United Kingdom, 2010.

[6] H. Chen, Y. Zhu, and K. Hu, " Adaptive Bacterial Foraging Optimization,"Abstract and Applied Analysis,Hindawi Publishing Corporation, 2011.

[7] R. C. Eberhart, and J. Kennedy, "A New Optimizer using Particle Swarm Theory,"InProceedings of the Sixth International Symposium on Micro Machine and Human Science,*Nagoya*, Japan, vol. 1, pp. 39-43. 1995.

[8] M. Dorigo, G. D. Caro, and L. M. Gambardella, "Ant Algorithms for Discrete Optimization," Artificial Life,vol.5, no. 2,pp.137-172,1999.

[9] D. Karaboga, "An Idea based on Honey Bee Swarm for Numerical Optimization, "Technical Report, Erciyes University, Engineering Faculty, Computer Engineering Department, pp. 1-10, 2005.

[10] K. M. Passino, " Biomimicry of Bacterial Foraging for Distributed Optimization and Control," IEEE Control Systems Magazine, June 2002.

[11] L. S. Coelho and C. A. Sierakowski, " Bacteria Colony Approaches with Variable Velocity Applied to Path Optimization of Mobile Robots," 18th International Congress of Mechanical Engineering, OuroPreto, MG, Brazil, November 6-11, 2005.

[12] H. Miao, " Robot Path Planning in Dynamic Environments using Simulated Annealing Based Approach," Master thesis, Queensland University of Technology, Queensland, Australia, March 2009.

[13] S. Das, A. Biswas, S. Dasgupta1, and A. Abraham," Bacterial Foraging Optimization Algorithm: Theoretical Foundations, Analysis, and Applications", Foundations ofComputational Intelligence, vol. 3, SCI 203, pp.23-55,Springer, 2009.

[14] X. Yan, Y. Zhu, H. Zhang, H. Chen, and B. Niu, "An Adaptive Bacterial Foraging Optimization Algorithm with Lifecycle and Social Learning," Hindawi Publishing Corporation ,Discrete Dynamics in Nature and Society, pp. 1-20, 2012.

[15]] O. Khatib, "Real-Time Obstacle Avoidance for Manipulators and Mobile Robots", Int. J. of Robotics Research, vol. 5, no. 1, pp. 90-99, 1986.

[16] H. Adeli, M.H.N. Tabrizi, A. Mazloomian, E. Hajipourand M. Jahed, "Path Planning for Mobile Robots using Iterative Artificial Potential Field Method",International Journal of Computer Science (IJCSI), vol. 8, Issue 4, no 2, July 2011.

[17] Y. Koren and J. Borenstein,"Potential Field Methods and their Inherent Limitations for Mobile Robot Navigation," Proceedings of the 1991 IEEEInternational Conference on Robotics and Automation,Sacramento, California, vol. 2, pp.1398-1404,09-11 April1991.

[18] L. C.Ling, T. Jing and Y. J.Yu, "Path Planning of Mobile Robot Using New Potential Field Method in Dynamic Environments", Seventh International Conference on Natural Computation (ICNC),Shanghai, China, pp.1011-1014, 26-28 July 2011.

[19] B. Mohajer, K. Kiani, E. Samiei, and M. Sharifi, " A New Online Random Particles Optimization Algorithm for Mobile Robot Path Planning in Dynamic Environments," Mathematical Problems in Engineering, Hindawi Publishing Corporation, vol. 2013, Article ID 491346, 9 pages, 2013.

تخطيط مسار الروبوتات المتنقلة على أساس خوارزمية النهم للبكتريا المحسنة

نزار هادي عباس* فرح مهدي علي**

قسم الهندسة الكهربائية / كلية الهندسة / جامعة بغداد

*البريد الألكتروني :drnizaralmsaodi@gmail.com

**البريد الألكتروني :farah95m@yahoo.com

الخلاصة

يقدم هذا البحث ، وصفا لمشكلة تخطيط مسار الروبوتات المتنقلة في الزمن الحقيقي والمتمثلة بايجاد مسار امثل للروبوت من نقطة بداية والى نقطة نهاية على خريطة مسطحة بشكل مساحة ثنائية الابعاد. قدم خوارزمية محسنة لطريقة النهم للبكتريا لحل مسألت تخطيط المسار. ان هذه الخوارزمية المستوحاة من الطبيعة والتي تقلد عملية بحث عن المؤن لبكتريا مسماة بال (أي كولاي) استخدمت لايجاد مسار من نقطة بداية الى نقطة نهاية. الخوارزمية المقترحة تم عرضها واثباتها عن طريق محاكاة لنوعين من البيئات المختلفة الثابتة والمتحركة. اجريت مقارنة بين الطريقة المطورة و طريقتين اخريين حديثة. هذا البحث اظهر فاعلية هذه الطريقة المقترحة في ايجاد مسارات بشكل مرضٍ.