Rev. Roum. Sci. Techn.– Électrotechn. et Énerg.
Vol. **63**, *1*, pp. 106–111, Bucarest, 2018

A NOVEL GOAL-ORIENTED STRATEGY FOR MOBILE ROBOT NAVIGATION WITHOUT SUB-GOALS CONSTRAINT

MUHAMMAD ZOHAIB[1], JAMSHED IQBAL[2,3,4], SYED MUSTAFA PASHA[4]

Key words: Autonomous mobile robots, Obstacle avoidance, Path planning.

This paper proposes an approach for obstacle avoidance to ensure safe navigation in a mazy environment. The presented bubble bug algorithm (BBA) is an enhancement of already proposed intelligent bug algorithm (IBA), which over performs bug variants. The improved performance of BBA is achieved by fusing IBA with bubble band technique (BBT) and bubble rebound algorithm (BRA). The novelty of the proposed strategy BBA lies in its salient features compared with the existing algorithms, achieved by addressing limitations of IBA and BRA. The proposed solution is goal-oriented and offers collision avoidance from neighboring obstacles by defining bubble around the robot. Simulation results illustrating navigation in various scenarios and comparative plots of path cost witness effectiveness of the proposed approach.

1. INTRODUCTION

History of robots dates back to 17th century with the development of a mechanical doll of a human size. The revolution in the field of mechatronics has made it possible to see the 'fiction' robots in reality in various fields of life [1, 2]. Today, robots are being actively used for rehabilitation, motion assistance [3] and in many other application areas including space and industries [4, 5]. Broadly speaking, robots can be categorized into two fundamental groups: manipulator type arms [6] and wheel based mobile robots.

The increasing applications of mobile robots have brought up several challenges regarding their path planning, safe navigation, control strategy and obstacle avoidance etc. One of the key issues concerning mobile robot locomotion is to simultaneously address the path planning problem and to devise the control strategy for obstacle avoidance [7]. Conventional monitoring methods are limited due to the uncertainty of environment. Real world scenarios necessitate considering dynamic and unknown environments thus highlighting significance of control strategy for obstacle avoidance autonomous navigation.

With a focus on these features, the present paper extends already proposed research [8] and combines path planning and control for autonomous robot navigation by proposing a goal-oriented algorithm. The path planning provides global solution whereas obstacle avoidance feature deals with upcoming obstacles especially faced by the robot during the edge following process. This is achieved by introducing a dynamic bubble around the robot that offers traversal along a safe trajectory.

2. LITERATURE REVIEW

Literature reports variety of the algorithms to navigate mobile robots autonomously. Bug algorithms are fundamental and complete [9] with provable guarantee, since they let the robot to reach the destination if it lies in the given space. Each algorithm in bug family carries its own termination property. A comparison of bug navigation algorithms is reported in [10]. These algorithms may take the robot far away from the goal in some scenarios [7, 11]. Simultaneous localization and mapping (SLAM) based solutions can generate map of the environment [12]. However, most of the reported studies of SLAM are limited to indoor locations. For an outdoor environment, an algorithm based on GPS and smart phone can be used for autonomous navigation. For underwater environments, a terrain map building method has been proposed by E.H. Lee and S. Lee in [12]. Highlighting terrain matching problems, a path planning approach has been presented by Li. *et al.* in [13]. The reported algorithms which are most relevant to the current research are presented below:

2.1. DIST-BUG ALGORITHM: THE MOST EFFICIENT IN BUG FAMILY

The earliest algorithms from bug family bug1 and bug2 offer minimum memory requirements, easy tuning and do not suffer from local minima [11]. However, they do not have capability to make optimal use of sensor's data to generate shorter paths. An improved approach, dist-bug, which is considered as the most advanced variant of bug family, addresses this problem [8]. It allows the robot to converge in comparatively less time by considering the shorter distance from the robot's position to destination. The trajectory of dist-bug algorithm is illustrated in Fig. 1.

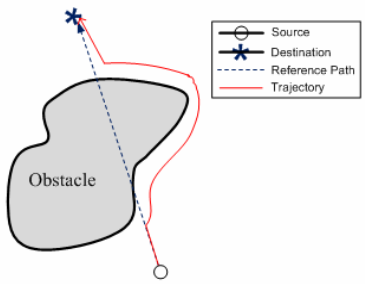

Fig. 1 – Trajectory of dist-bug algorithm.

2.2. INTELLIGENT BUG ALGORITHM (IBA): AN IMPROVED VERSION OF DIST-BUG ALGORITHM

Although dist-bug algorithm considers the path cost throughout the decision making process, however it is not goal oriented strategy. Thus, it can restrict a robot to follow edges of an obstacle even in the presence of an obstacle free

[1] University of Central Punjab, Department of Information Technology, Lahore, Pakistan
[2] FAST National University of Computer and Emerging Sciences, Department of Electrical Engineering, Islamabad, Pakistan
[3] University of Jeddah, Electrical and Computer Engineering Department, Kingdom of Saudi Arabia, Email: jmiqbal@uj.edu.sa
[4] COMSATS Institute of Information Technology, Department of Electrical Engineering, Islamabad, Pakistan

path towards goal. This gives a clue to improve dist-bug algorithm by devising an approach to make it goal oriented and to consider time to destination. Based on this, IBA proposed in [8], offers an intelligent control to navigate the robot in maze environments. By taking goal position into account, leaving point in IBA is selected on the basis of free path toward the destination. Leaving point is the position from where the robot stops following edges of an obstacle and starts moving towards goal. The improved characteristics of IBA make it efficient to prove its convergence with relatively shorter and smoother trajectory in comparatively less time as proved in [8]. The trajectory followed by a robot using IBA is illustrated in Fig. 2.

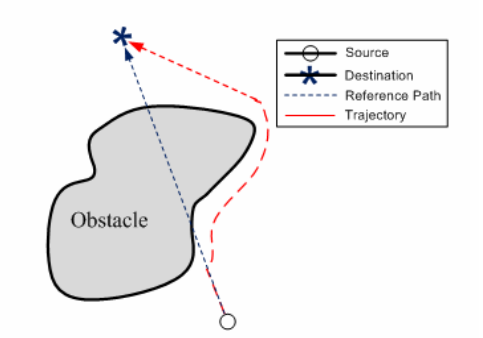

Fig. 2 – Trajectory of IBA.

2.3. RANDOM PARTICLES OPTIMIZATION (RPO) ALGORITHM: IMPROVEMENT IN ARTIFICIAL POTENTIAL FIELD (APF) METHOD

RPO algorithm [14], proposed by Mohajer *et al.*, improves APF approach by steering a robot without suffering from local minima problem. In RPO algorithm, particles are distributed randomly on a circle of radius C(t) around a robot. These particles are used to search optimal path toward the destination to navigate a mobile robot in an unknown environment. The robot is finally steered in the direction corresponding to the searched optimal path. This procedure is iterated until the robot reaches its destination. The corresponding trajectory followed by a robot from source to destination is shown in Fig. 3.

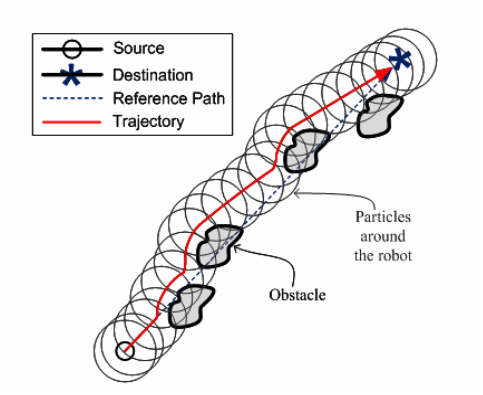

Fig. 3 – Behaviour of the robot executing RPO.

If $\theta_s(t)$ is the position of each particle s in time t, then the next position at time (t+dt) can be calculated by eq. (1).

$$\theta s(t+\mathrm{d}t) = \theta s(t) + C(t)\frac{\Delta(t)}{\|\Delta(t)\|}, \tag{1}$$

where $\Delta(t)$ is a vector of unit length used to define the direction of a particle w.r.t time. The selection of the best particle is based on two different strategies: Gaussian cost function and distance norm to destination. When an obstacle is sensed by the particles, the algorithm generates a repellent Gaussian function J^o for the obstacle and an attractant Gaussian J^D function for destination. Both of these functions are given by eq. (2) and (3) respectively.

$$J^o = \begin{cases} \alpha^o \exp\left(-\mu^o \|\theta_i(t) - P_o(t)\|^2\right) & \|P_o(t) - q_o(t)\|^2 \le \beta \\ 0 & \|P_o(t) - q_o(t)\|^2 > \beta \end{cases} \tag{2}$$

$$J^D = -\alpha^D \exp\left(-\mu^D \|\theta_i(t) - P_D\|^2\right), \tag{3}$$

where constants ($\alpha°$, $\mu°$) and (α^D, μ^D) define the height and width of repellent and attractant respectively. $P_o(t)$, $q_o(t)$ and P_D are the positions of obstacle, robot and destination respectively. β is the range of sensor. The range of i and s vary from 1 to total number of particles n. Eq. (2) demonstrates that J^o is zero if the obstacle is outside the range of sensors. Total cost function is given by eq. (4).

$$J = J^o + J^D. \tag{4}$$

To find the best particle, error in distance to destination and error in cost function are calculated. The distance $d_s(t)$ at time t of a particle s from the goal can be computed as $\|\theta_s(t) - P_D(t)\|^2$. If $d_s(t + \mathrm{d}t)$ is the distance at time $t + \mathrm{d}t$, then errors in distance and cost function are given by eq. (5) and (6) respectively.

$$e_s^d(t+\mathrm{d}t) = d_s(t+\mathrm{d}t) - d_s(t) \tag{5}$$

$$e_s^J(t+\mathrm{d}t) = J\left(\theta_s(t+\mathrm{d}t)\right) - J\left(\theta_s(t)\right). \tag{6}$$

The particle with minimum distance error having negative error cost function is selected as the best particle. Finally, the robot takes next step toward this particle with control $u(t)$ as given in eq. (7).

$$u(t) = C(t)\frac{\theta s_{best}(t+\mathrm{d}t) - q(t)}{\|\theta s_{best}(t+\mathrm{d}t) - q(t)\|}. \tag{7}$$

2.4. BUBBLE REBOUND ALGORITHM (BRA): AN ENHANCEMENT OF BUBBLE BAND TECHNIQUE (BBT)

BBT defines the bubbles in the robot's surrounding by considering the maximum free space around it. Initially, obstacle-free path with sharp turns is determined by the path planner. This path is then regenerated with smooth and short trajectory based on the concept of elastic bands [15]. Virtual forces are applied until the elastic band reaches the equilibrium. These forces consider the kinematics and dynamics of a robot w.r.t. its configuration, obstacle's position and path [16]. The concept of virtual force is similar to that in APF algorithm, however BBT allows the collision avoidance forces unlike APF. The elastic band is represented as a series of bubbles so as to ensure that the path lies within the free space. The features e.g. shape and size of the bubbles can be different and are determined by a robot's geometry and minimum distance between the robot

and the obstacle in an environment. The path from mid of the bubbles allows the robot to traverse safely by avoiding collision. Behaviour of the robot executing BBT is illustrated in Fig. 4.

BRA [17] is an advanced version of BBT in which the bubbles are determined within the range of sensors by considering the possible distance that can be covered by a robot in a specific time interval. BRA assumes that the robot is equipped with an array of equidistant sonar sensors covering field of view (FoV) of 180°. This arrangement allows a half circular bubble as shown in Fig. 5.

A robot following BRA is initially steered toward the destination. In case the robot encounters an obstacle in its path within the bubble, it adjusts itself in the direction of calculated rebound angle having lowest obstacles density.

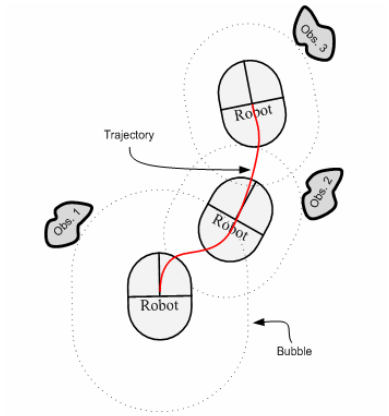

Fig. 4 – Illustration of BBT.

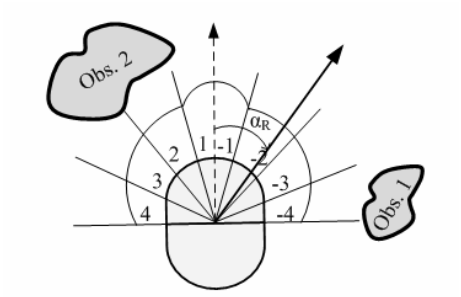

Fig. 5 – Sensor's alignment and the bubble in BRA.

It then starts moving in the new direction until the destination is visible or another obstacle is detected by the bubble. The rebound process is exemplified in Fig. 6, which shows that the robot turns to the computed rebound angle after detecting an obstacle and continues its motion until it reaches the destination.

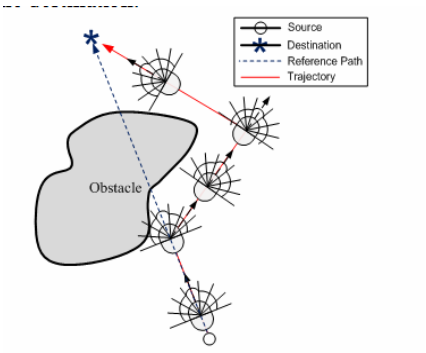

Fig. 6 – Behaviour of the robot executing BRA.

The rebound angle is computed by considering the sensors' arrangement. Because of their uniform distribution in 180°, the angular separation α_o is defined by eq. (8).

$$\alpha_o = \frac{\pi}{N}, \tag{8}$$

where N is the total number of sensors. The position of i^{th} sensor α_i is given by eq. (9)

$$\alpha_i = i\alpha_o, \quad \alpha_i \in \left[-\frac{N}{2}, \quad \frac{N}{2}\right]. \tag{9}$$

Using eq. (8) and (9), the rebound angle α_R can then be calculated by eq. (10).

$$\alpha_R = \frac{\sum_{i=-N/2}^{N/2} \alpha_i D_i}{\sum_{i=-N/2}^{N/2} D_i}, \tag{10}$$

where D_i is the distance detected by i^{th} sensor. In a scenario having two obstacles which are located symmetrically, the rebound angle is simply $\pi/2$.

3. NOVELTY OF THE PROPOSED STRATEGY

The novelty of BBA is justifiable based on the issues exhibited by the reported algorithms detailed below:

3.1. DIST-BUG

- Being a part of bug family, it is not a goal oriented algorithm. The path of a robot is only a function of minimum distance to destination.
- It considers a robot as a point without taking into account its dimensions.
- The collision may be possible in the presence of an obstacle adjacent to a robot especially while avoiding the edges of the obstacle.
- The decisions are based on the current percepts and therefore sensor noise may result in a wrong decision.

3.2. IBA

- Limitations of IBA are similar to dist-bug algorithm, however it is a goal oriented algorithm since it takes the goal information into account throughout its motion.

3.3. RPO

- The algorithm assumes a mobile robot as a point robot.
- The algorithm is designed by considering holonomic omni-wheeled mobile robots without taking into account the nonholonomic constraints.
- The radius of the circle for random particles distribution is constant and is not a function of dynamics or dimensions of a robot.
- It does not constrain a robot to maintain a certain distance from obstacles and thus the obstacles may lie inside the imaginary circle of particles.

- In case of a nonholonomic robot, collision with a nearby obstacle may be possible in a fully dynamic environment.

3.4. BRA

- It is not a goal oriented obstacle avoidance strategy since the robot avoids obstacles by considering the bubble rebound angle without taking into account direction of the goal.
- It may take a robot far away from the destination in case the goal is not visible.
- It does not offer smooth trajectory.
- The robot may stop in front of detected obstacle in order to adjust its heading.
- It may fail to work in some cases even a valid path to destination is available.
- It requires defining sub-goals from source to destination in certain mazy-type scenarios.

Looking onto these limitations, it is evident that there is no single algorithm that encompasses all the following features in one approach i.e.

- Goal orientation.
- Consideration of nonholonomic constraints, robot's dimensions and surrounding.
- Applicability for holonomic robots.
- Execution of relatively safer trajectories by limiting a robot to maintain a minimum distance from obstacles e.g. using dynamic bubble.
- Convergent behavior in case the destination is present in an environment.
- No requirement of sub-goals in a mazy environment.
- Simple and cost-effective control implementation e.g. using microcontrollers and range sensors.

The novelty of BBA lies in proposing a strategy that offers all of the aforementioned salient features.

4. PROPOSED BUBBLE BUG ALGORITHM (BBA)

The proposed BBA offers a robust autonomous control by introducing a dynamic bubble around the robot to navigate it in a mazy-like environment. Similar to IBA, the strategy has two main behaviors; move to goal and obstacle avoidance. In move to goal behavior, the robot plans a reference path from its current position to destination and follows it until the destination is reached or an obstacle is encountered. After detecting an obstacle, the behavior of the robot is switched to obstacle avoidance, which is an advanced version of the corresponding behavior in IBA since it introduces the dynamic bubble to tackle nearby obstacles.

4.1. INTRODUCING BUBBLE

The circular shaped bubble takes into account the robot's parameters and indicates the minimum distance at which the nearest obstacle can exist. It is a function of speed, rotational radius and distance of the nearest obstacle which can be determined by the range sensor. The size of the bubble is dynamically adjustable up to the maximum range of sensors with respect to the change in robot's velocity. So, it will be large in size for speedy robots having long range sensors and short for slow robots. Reference to Fig. 7, if R_{rob} is the robot's rotational radius, V_{rob} is its velocity and D_{max} is the maximum allowable distance for nearest obstacle, the radius of the bubble R_{bub} can be determined by eq. (11)

$$R_{bub} = R_{rob} + \left[\frac{V_{rob} \times D_{max} + R_{rob}}{D_{max}} \right]. \qquad (11)$$

Considering the limitations of a physical robot, two assumptions are taken *i.e.* V_{rob} can vary from 0 to $D_{max}/4$ and $R_{rob} << D_{max}$, where in the first assumption, $D_{max}/4$ is selected to take into account the sensor's limitations i.e. the robot's velocity must be within the range of the sensors so that it can navigate safely without being affected by the sensor's noise. The second assumption implies the sensor's efficiency (range) w.r.t. the robot's rotational radius. The radius of the bubble must satisfy this assumption so that the robot can turn in its vicinity without encountering a collision during edge following process. The sensor's alignment with the bubble is illustrated in Fig. 7.

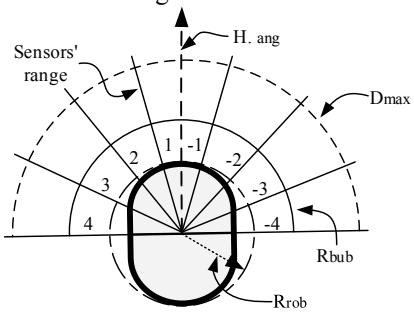

Fig. 7 – Dynamic bubble with sensors' alignment

4.2. OBSTACLE AVOIDANCE IN BBA

In obstacle avoidance behavior, the robot computes the heading angle using eq. (10) and adjusts itself by rotating along its own axis while considering the nonholonomic constrains for safe navigation. After adjusting itself, the robot starts following edges of the obstacle in the same way as in IBA. Simultaneously, it monitors the presence of obstacles in its surrounding. Figure 8 depicts the behavior of the robot executing BBA.

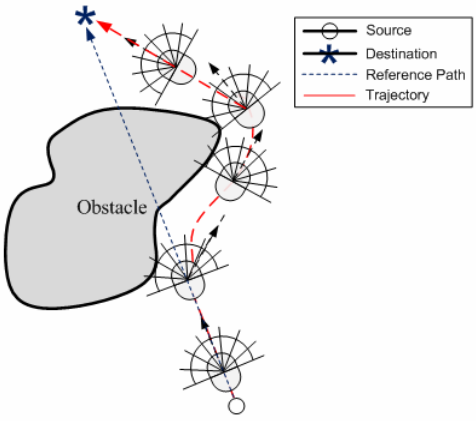

Fig. 8 – Behaviour of the robot executing BBA.

During the edge following process, if the robot encounters an obstacle within the bubble, it starts following the boundary of the new obstacle. In case, two or more obstacles are encountered, the robot follows edges of the nearest obstacle. For symmetric obstacles (i.e. obstacle sensed by sensors 1 and –1), the resultant heading angle is zero and

the robot is attuned toward 90º to follow the boundary as discussed in [17]. The flowchart of BBA is presented in Fig. 9, which clearly defines the obstacles detection in both behaviors of the robot. The robot continuously monitors upcoming obstacles and considers visibility of the goal during edge following process. The switching condition from obstacle avoidance to move to goal is also based on visibility of obstacle-free path toward destination.

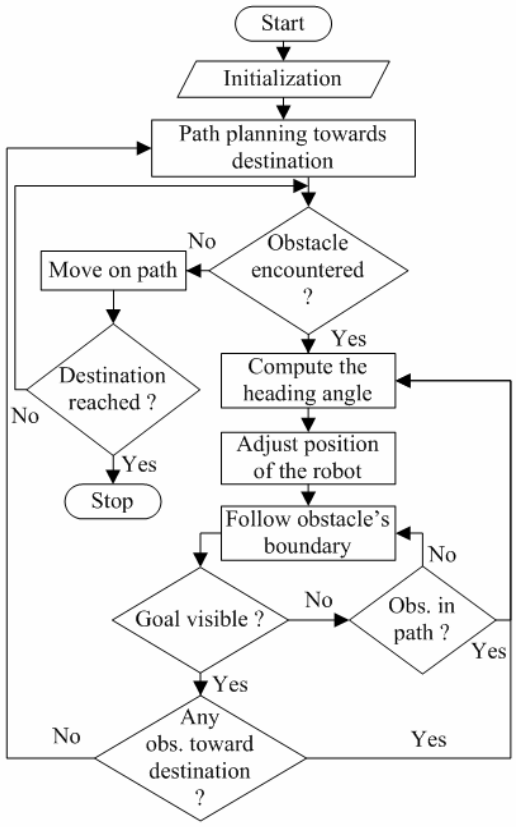

Fig. 9 – Flowchart of BBA.

5. SIMULATION RESULTS

The contribution of the proposed approach has been demonstrated in simulation. For this purpose, a virtual environment is created using MATLAB running on 2014a, running on a computer with the following specifications; HP 1000 Notebook, Intel® Core™ i3-2370M CPU @ 2.40 GHz and 64-bit Operating System (OS). After implementing the algorithms in MATLAB programming environment, the results are achieved to compare performance of IBA and BBA with the reported algorithms. Three main simulated scenarios are presented below:

5.1. SCENARIO-1: IMPROVEMENT OF IBA OVER DIST-BUG ALGORITHM

Comparing the performance of IBA and dist-bug algorithm in the same environment, Fig. 10 presents the corresponding robot trajectories. The behavior of the robot in dist-bug algorithm is illustrated in Fig. 10a, where the robot is following the edge of the obstacle until it reaches to the leaving point having minimum distance to destination. Simulation result of IBA shown in Fig. 10b indicates that the robot follows edges of the obstacle until it finds a clear path toward the destination.

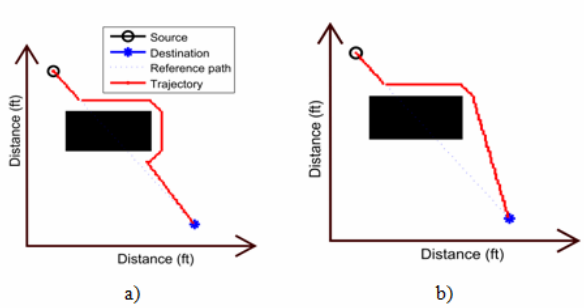

Fig. 10 – Robot trajectories for performance comparison using a) dist-bug algorithm; b) IBA.

A path cost plot given in Fig. 11 presents the time taken by the two algorithms. Comparing the robot trajectories confirms that IBA improves dist-bug algorithm since the trajectory profile of the former approach is comparatively shorter and smoother. For the given scenario, time taken by IBA and dist-bug algorithm is 162 s and 179 s respectively, which highlights 9.5 % over-performance of IBA in terms of path cost.

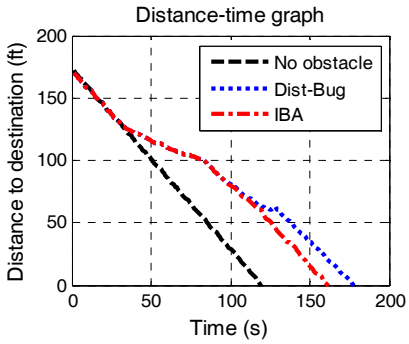

Fig. 11 – Path cost comparing dist-bug and IBA.

5.2 SCENARIO-2: NOVELTY OF BBA IN CONTRAST WITH IBA

Consider a scenario of two obstacles placed closer to each other in a virtual environment. Figure 12a shows the behavior of the robot in IBA, where the robot starts following the reference path. After encountering an obstacle in its path, the robot starts avoiding it by following its edges. Meanwhile, it collides with another nearby obstacle and diverges from its path ultimately losing the way toward destination. Figure 12b illustrates the behavior of BBA in the same scenario. The robot initially tracks the reference path and after detecting an obstacle, it starts following its edges. After finding second obstacle, the robot adjusts itself in the direction of calculated heading angle and starts following the edges of the second obstacle. Finally, it reaches the destination without collision thus highlighting the novelty of BBA in contrast with IBA.

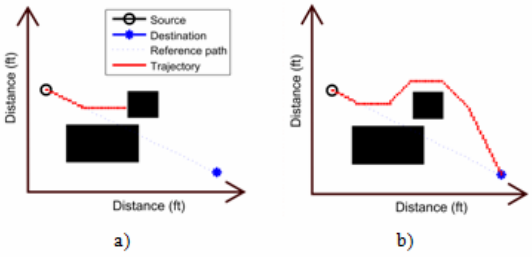

Fig. 12 – Robot trajectories for performance comparison.

using a) IBA; b) BBA.

5.3 SCENARIO-3: CONVERGENCE OF BBA IN MAZY ENVIRONMENT IN COMPARISON WITH BRA

Scenario 3 deals with the limitations of BRA as it requires sub-goals to converge in a mazy environment. A virtual environment for this scenario is created that consists of an obstacle placed in such a way that it hinders the line of sight (LOS) of source and destination. Fig. 13a shows the behavior of the robot under BRA, where the robot starts following the path toward destination. On encountering the obstacle, the robot avoids it by moving toward the defined sub-goal 1. After reaching there, it follows the path toward the second sub-goal and finally reaches the destination. Simulation result of the proposed BBA is illustrated in Fig. 13b. The robot continuously follows the edges of the encountered obstacle until it finds a clear path toward destination without requiring sub-goals.

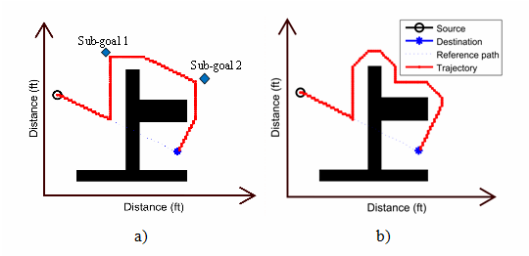

Fig. 13 – Robot trajectories for performance comparison using: a) BRA; b) BBA.

The path cost of BBA is improved by 5.63 % in comparison with BRA which is evident in Fig. 14 since BBA takes 134 s whereas BRA takes 142 s in reaching the destination. The robot executing BBA does not need to stop in front of the obstacle for adjusting itself as the adjustment is made continuous during the edge following process. The proposed approach does not take the robot away from the goal and thus requires less time to converge. The time comparison graph confirms efficiency of BBA in the given scenario without suffering from sub-goals constraint.

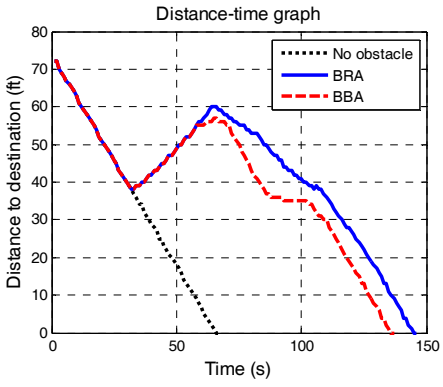

Fig. 14 – Path cost comparing BRA and BBA.

6. CONCLUSIONS

The paper presents a novel approach to improve IBA and BRA for navigating mobile robots autonomously. The proposed BBA offers short trajectory in lesser time as compared to other reported algorithms. Comparison of BBA with existing bubble based approach in terms of time cost function dictates over performance of the presented approach. The algorithm finds potential in applications deploying mobile robots in unknown static as well as dynamic environments. Going beyond mobile robots, result of this research can also be mapped on other real-world scenarios e.g. collision avoidance in a car parking system by combining BBA with intelligent follow the gap method (IFGM). It is envisaged in near future to investigate optimization and robustness of BBA more rigorously in relatively much complicated scenarios by implementing it on a real robotic platforms.

Received on April 1, 2017

REFERENCES

1. W. Alam, A. Mehmood, K. Ali, U. Javaid, S. Alharbi, J. Iqbal, *Nonlinear control of a flexible joint robotic manipulator with experimental validation*, Strojniški vestnik – Journal of Mechanical Engineering, **64**, *1*, pp. 47–55 (2018).
2. J. Iqbal, R.U. Islam, S.Z. Abbas, A.A. Khan, S.A. Ajwad, *Automating industrial tasks through mechatronic systems – A review of robotics in industrial perspective*, Tehnicki Vjesnik-Technical Gazette, **23**, *3*, pp. 917–924 (2016).
3. L. Vlădăreanu, A. Curaj, R.I. Munteanu, *Complex walking robot kinematics analysis and PLC multi-tasking control*, Rev. Roum. Sci. Techn. – Électrotechn. et Énerg., **57**, *1*, p. 90–99 (2011).
4. B. Dumitrescu, A.D. Ionita, H. Gavrilă, *Picking lines modeling*, Rev. Roum. Sci. Techn. – Électrotechn. et Énerg., **61**, *1*, pp. 78–83 (2016).
5. A.I. Gal, L. Vladareanu, R.I. Munteanu, *Sliding motion control with bond graph modeling applied on a robot leg*, Rev. Roum. Sci. Techn. – Électrotechn. et Énerg, **60**, *2*, pp. 215–224 (2015).
6. O. Khan, M. Pervaiz, E. Ahmad, J. Iqbal, *On the derivation of novel model and sophisticated control of flexible joint manipulator*, Rev. Roum. Sci. Techn. – Electrotechn. et Energ., **62**, *1*, pp. 103–108 (2017).
7. M. Zohaib, S. M. Pasha, N. Javaid, J. Iqbal, *An improved algorithm for collision avoidance in environments having U and H shaped obstacles*, Studies in Informatics and Control **23**, *1*, pp. 97–106, (2014).
8. M. Zohaib, S.M. Pasha, N. Javaid, J. Iqbal, *IBA: Intelligent bug algorithm – A novel strategy to navigate mobile robots autonomously*, Emerging Trends and Applications in Information Communication Technologies, Communications in Computer and Information Science (Springer-Verlag Berlin, Heidelberg), **414**, pp. 291–299 (2014).
9. V. Sezer, M. Gokasan, *A novel obstacle avoidance algorithm: Follow the gap method*, Robotics and Autonomous Systems, **60**, *9*, pp. 1123–1134, (2012).
10. N. James, B. Thomas, *Comparison of bug navigation algorithms*, Journal of Intelligent and Robotic Systems, Springer Science, **50**, *1*, pp. 73–84 (2007).
11. A. Yufka, O. Parlaktuna, *Performance comparison of bug algorithms for mobile robots*, 5th International Advanced Technologies Symposium, Karabuk, Turkey (2009).
12. E.H. Lee, S. Lee, *Development of underwater terrain map building method on polar coordinates by using 3D sonar point clouds*, International Journal of Applied Engineering Research, **11**, *14*, pp. 8259–8264 (2016).
13. Y. Li, T. Ma, P. Chen, Y. Jiang, R. Wang, Q. Zhang, *Autonomous underwater vehicle optimal path planning method for seabed terrain matching navigation*, Ocean Engineering, **133**, pp. 107–115 (2017).
14. B. Mohajer, K. Kiani, E. Samiei, M. Sharifi, *A new online random particles optimization algorithm for mobile robot path planning in dynamic environments*, Mathematical Problems in Engineering, Article ID 491346, 2013.
15. V. Kunchev, L. Jain, V. Ivancevic, A. Finn, *Path planning and obstacle avoidance for autonomous mobile robots: A review*, Knowledge-Based Intelligent Information and Engineering Systems, (Springer Berlin, Heidelberg), pp. 537–544 (2006).
16. C. Frese, J. Beyerer, *A comparison of motion planning algorithms for cooperative collision avoidance of multiple cognitive automobiles*, IEEE Intelligent Vehicles Symposium, pp. 1156–1162 (2011).
17. I. Susnea, A. Filipescu, G. Vasiliu, G. Coman, A. Radaschin, *The bubble rebound obstacle avoidance algorithm for mobile robots*, 8th IEEE International Conference on Control and Automation, Xiamen, China, 2010, pp. 540–545.