Journal of Optimization in Industrial Engineering
Vol.14, Issue 1, Winter & Spring 2021, 133-148
DOI: 10.22094/JOIE.2020.1863337.1629

Research Article

A Hybrid Method for Industrial Robot navigation

Somayeh Raiesdana*

Faculty of Electrical, Biomedical and Mechatronics Engineering, Qazvin Branch, Islamic Azad University, Qazvin, Iran

Received 25 February 2019; Revised 14 September 2020; Accepted 24 September 2020

Abstract

Robot navigation in dynamic unknown environments is a challenging issue in the field of autonomous mobile robot control. This paper presents a hybrid robust method for navigating an industrial robot in an environment that contains dynamic obstacles. The objectives are to find the shortest path, to minimize the energy consumption of robot, to make the smoothness of the generated paths and to tackle dynamic obstacles. Robots employed in industrial environments demand considerable autonomy and require high level of accuracy and manoeuvrability at the same time. Besides, no collision is tolerable along the way. A single-objective optimization method based on path criteria fails to satisfy all of the requirements. This paper proposes a hybrid algorithm including the whale optimization algorithm (WOA) for path planning, a learnable function approximation network for making smoothness of the generated paths and a fuzzy logic controller to avoid obstacle collision. In this algorithm, WOA optimizes the best path to be taken from the start to goal position. Once a sequence of points is candidate and segments of path are merged, a radial basis function is trained to provide a smooth movement path in the dynamic environment while trying to maximize the safety margin. To further improve the safety of navigation, a fuzzy-based obstacle avoidance algorithm is executed when the robot is placed in the vicinity of an obstacle. Fuzzy decisions are made based on values of distance information. The proposed hybrid method for path planning and obstacle avoidance issues was implemented and evaluated in dynamic environments including specific shaped obstacles. A GUI-based simulation platform was designed in Matlab environment for testing the proposed algorithm. Implementation results indicate that the proposed algorithm has yielded in smooth non-marginal goal-directed navigation with acceptable performance metrics. Meanwhile, collisions to dynamic obstacles were adaptively and non-rigidly avoided. Such a model-free hybrid algorithm for path planning and obstacle avoidance can improve autonomy in industrial operation and decrease computational complexity.

***Keywords*:** Mobile robot navigation; Path planning; Obstacle avoidance; Whale Optimization Algorithm; RBF network; Fuzzy Control.

1. Introduction

Mobile robots are increasingly being employed in many static and dynamic environments for an enormous range of applications and received increasing attentions from both academy and industry. They have been widely applied in many fields including industry, agriculture, architecture and military because of their working abilities in hazardous environments. They can help to eliminate the people from direct hostile conditions in hazardous environmental situations and also to increase productivity. Such robots are able to accept high-level descriptions of tasks and execute them without human intervention (Zavlangas and Tzafestas, 2000).

What is expected in manufacturing industries is to develop real-time autonomous robots that can sense and interpret information collected from their environment to determine their position and direction toward goal, avoid colliding to obstacles ahead and perform a smooth navigation in both structured and unstructured environments. The ability to perform autonomously a multitude of tasks without complete a priori information, while adapting to continuous changes in the working environment is in general demanded for intelligent robots with high safety (Huji et al., 1998). On the other hand, in order to be cost effective for industry, limited sensor technology is embedded for mobile robots employed in a factory and in the production line and vision is often not considered.

Effective control of a mobile robot’s locomotion, also called as ‘Robot Navigation’ along with obstacle avoidance is one of the challenging tasks in industrial applications. Path generation and obstacle avoidance are integral parts of autonomous mobile robot navigation. Many researchers have developed and used different techniques to solve path planning for mobile robot. path planning can be categorized into two kinds; global path planning where the robot has complete knowledge about the navigational environment and local planning in which no prior information of the environment is required. In general, global path planning

*Corresponding author Email address: raiesdana@qiau.ac.ir

has less robustness whereas local path planning methods show more flexibility in partially known/unknown environments (Zafar and Mohanta, 2018). Furthermore, hybrid navigation refers to the union of local and global navigation algorithms trying to extend them with additional functionality such as the local minima detection, or the use of obstacle memory. On the other hand, path planning and obstacle avoidance methods are classified into static or dynamic based on the environment where it has fixed or moving objects respectively.

To navigate a robot, an effective motion planner should be designed. Several approaches have already been addressed for robot navigation which mainly aimed to perform simultaneous path planning and obstacles avoidance within static environments. Robot path planning methods generally consist of non-heuristic (classic) and heuristic approaches. Classic algorithms such as Bug family (Ng and Bräunl, 2007), wall following method (Tsui et al., 2008), Artificial Potential Field (APF) methods (Ge and Cui, 2000), cell decomposition (Dugarjav et al., 2015), edge detection, graph- based method (Wooden, 2006), vector field histogram (VFH) method (Borenstein and Koren, 1991), virtual target approach (Groen, 2000) and landmark learning (Deng and Milios, 1996) are computationally more expensive. The classic path planning algorithms suffer from some drawbacks. Due to the complexity and uncertainty of the path planning problem, classical path planning techniques are not suitable for online implementations. They do not produce optimal paths and are often trapped into local minima before reaching the target. Local minima are recognized, when the agent get stucked at a certain state or position. It can be raised by a closed door which was previously open when the path was designed by the global navigation system. Hence, solution may not be optimal and a new solution has to be generated again when the environment changes and therefore the original path may become infeasible. Furthermore, for a large scale problem and when the degree of freedom is high, the traditional methods have to pay expensively to deal with high computational costs and the high extent of complexities (Dao et al., 2016).

On the other hand, nature-inspired meta-heuristic algorithms such as genetic algorithm (GA), particle swarm optimization (PSO), A* algorithm, simulated annealing, firefly algorithm, ant colony optimization and many others are frequently applied for path planning of mobile robot. Most of these meta-heuristic algorithms are inspired from and mimicking the successful features of the underlying biological, physical or sociological systems. Meta-heuristic algorithms including learning, expert-based or nature inspired are particularly useful for optimization problems increasing in dimension and in complexity.

Normally, there are various feasible paths for a robot to reach a target from a start location. However, the best feasible path is the one which satisfies a number of objectives. Planning must satisfy some criteria related to the path; e.g. the length of the planned path should be the shortest and the energy consumption of robot should be the lowest. The next objective is to keep the smoothness of the generated paths as much as possible. Those paths that are generated by geometry-based methods or paths that are planned partially are generally non-smooth and their practicality is questionable in most of the cases since mobile robots lose considerable energy and time when turning their course of moving abruptly. Ultimately, the obstacle avoidance objective is very vital for dynamic environments wherein unpredictable moving obstacles exist. In many real-life situations, the robot needs to keep a certain safe interspace from obstacles to avoid collisions. Furthermore, for industrial applications, avoiding collision to moving obstacles in dynamic environments should be strictly considered. Unlike the static problem, the moving obstacle problem cannot be solved by merely constructing a geometric path even if the best or shortest path between the start and the goal position can be found. Some other objectives are cost minimization in reaching from source to destination, reducing the reaching time as well as reducing the computational, time and space (memory) complexity.

Although numerous researches and experimental works have been carried out for robot navigation, there still exists challenges for mobile robots used in industrial environments (e.g. in a noisy and crowded factory). Such robots demand considerable autonomy and also require high degree of accuracy and manoeuvrability at the same time. A mobile robot in a factory has to cope with moving obstacles that may be human or other robots. Furthermore, in a constantly changing and partially unpredictable environment, robot motion planning must be on-line. On-line planning is an on-going activity that requires continuous flow of information about events occurring in the robot environment (Li and Latombe, 1997). The robot is expected to react against its changing environment by reacting immediately to new information received from sensors.

The motion planning method in this paper focuses on challenges related to dynamic environments. A hybrid global/local planning is proposed to find a suboptimal collision free path within an environment using a global path planner. Real time obstacle avoidance is then implemented using a simple and cost effective local planner. In this research, the main objective is to find an optimal collision free path for a navigating robot to be able to get to the desired goal without colliding to obstacles in a dynamic environment. The aim is to provide a compromise between the accuracy and autonomy of a goal-directed navigation for an industrial robot while the time and cost of navigation is also optimized randomly. A hybrid algorithm is considered safe for unknown environment where dynamic obstacle(s) is/are positioned. It is emphasized that no collision is tolerable along the way during a navigation. Meanwhile, providing a smooth path and avoiding a jerk or sharp turns are also requested; i.e. path smoothing is further necessitated for factory working robots. Smooth paths must satisfy certain constraints like continuity and safety. The continuity problem mainly refers to the geometric continuity in terms of tangential or curvature continuity. The safety check ensures that the smooth path is sufficiently far from the obstacles.

Given an environment surrounded by walls and a few obstacles inside, a robot has to decide on the next direction and position based on sensed distance information. An adaptable path planning method which can adapt to the environmental conditions without any supervision is required. The main approach proposed in this field is Simultaneous Localization and Map (SLAM) algorithm. This algorithm uses features of a map to estimate the location of robot and obstacles. Nonetheless, as the SLMA based maps are probability based and not absolute, the estimation error is accumulated over time which may even be intolerable for a complex environment where lots of dynamical entities exist. Hence, the uncertainty and ambiguity related to estimates of moving objects' position, direction and velocity is increasing which may ultimately lead to unsafe paths that are not practically feasible. This paper employs a newly introduced optimization algorithm named Whale Optimization Algorithm (WOA) for path planning. WOA mimics the social behavior of humpback whales and is inspired from the bubble-net hunting strategy of the foraging behavior of whales (Mirjalili and Lewis, 2016). This algorithm is based on the fact that the whale encircles a prey during chasing and entraps it into a net of bubbles. This method has shown successful performance in solving various optimization problems in comparison to other known optimization algorithms such as particle swarm, gravitational search, differential evolution and genetic (Mirjalili and Lewis, 2016). Some other aspects are further considered in the proposed path planning algorithm for making more improvements including safety margin, avoiding moving obstacles and improving the convergence capability. Make use of an RBF network and a fuzzy inference system will improve the robot navigation performance. Fuzzy logic and neural networks are universal problem-solving methods that can operate optimally in uncertain and unknown conditions.

The structure of this article is as follows. A brief review on related works is brought in section 2. Materials and methods including descriptions on computational tools are included in section 3. Section 4 gives an overview of the proposed method and the software implementation. The results and discussion are then reported in section 5 which is followed with a conclusion in section 6.

2. Literature Review

Metaheuristic methods have been applied successfully to solve robot path planning. Masehian and Sedighizadeh presented two novel PSO-based algorithms for robot path planning and evaluated the generated paths based on two criteria. Despite of being effective revealed by empirical experiments, this method only works for continuous optimization problem (Masehian and Sedighizadeh, 2013). Genetic algorithm has also been used to control mobile robot navigation in an environment included a number of static as well as to control motion when dynamic obstacle were existed (Kumar et al., 2018). Dijkstra's algorithm (Barbehenn, 1998) with uniform cost search has widely been used for global planning. However, it is computationally expensive with a time complexity of $O(N2)$, where N is the number of nodes. A-star algorithm and D-star (or Dynamic A*) algorithms were further used to find an optimal path in real-time by incrementally updating paths to the robot's state as new information is discovered. Rapidly exploring random tree (RRT) algorithm efficiently searches high dimensional spaces by building space-filling trees randomly. This method was used to plan a path with a biased growth towards the goal with high probability in (Ravankar et al., 2018)

Authors used fuzzy logic and neural network to handle the path planning problem respectively in (Singh et al., 2009) and (Vukosavjev et al., 2011). Besides of being employed to solve the path planning paradigms, these methods have occasionally been employed to solve various mobile robot navigation challenges (such as increasing the accuracy, optimizing the cost and time and also dealing with the environment uncertainty as well as the lack of information) in cooperation. In (Mohanty and Parhi, 2013) and (Kumar et al., 2014), the authors have used a new hybrid intelligent method to solve the navigation and path planning problem which was called MANFIS (Multiple Adaptive Neuro-Fuzzy Inference System). In this method, a fuzzy controller in the form of a Takagi-Sugeno decision-making system has been designed within an unsupervised learning network. The inputs of this system were distances obtained from front, left and right sensors as well as heading angle to compute left and right wheel velocity. The objective of this research was to reach the destination through the shortest path. In (Ganapathy et al., 2009), four combinations of neural networks and fuzzy system (two fuzzy controller, one for obstacle avoidance and one for wall following, and alternatively two neural networks) were implemented for moving robot in an operational environment attempting to learn the structure of environment and reach to the target. Results of this study showed that combination of fuzzy logic and neural network controller in a way that the neural network was used for obstacles avoidance and the fuzzy controller for wall following has worked better than other combinational structures. An ANFIS (Adaptive Neuro-Fuzzy Inference System) method has also been used in (Singh et al., 2009) for robot navigation. In this method, sensor values were inputted into a fuzzifier and then a learning process took place in the network. In reference (Vukosavjev et al., 2011), two navigation systems were combined: 1) a self-learning neural network used for path planning and generating and 2) a heuristic Neuro-fuzzy module to control collision avoidance. Results of this research showed a smooth collision-free navigation in the presence of stationary obstacles. However, not the shortest path was always followed. In further attempts, interactive methods were developed in (Joshi and Zaveri, 2010) in order to behaviorally control a mobile robot. The main objective of the proposed method in (Joshi and Zaveri, 2010) was to calculate the distance of robot to obstacles around and to obtain training dataset for neural network in order to generate the movement direction. Afterward, a fuzzy system was employed to adjust velocities of wheels for movement based on the neural network's outputs. In another study (AbuBaker, 2015), Abu Baker has comprehensively used 625 fuzzy rules in order to provide conditions of guaranteed collision avoidance. He has further utilized a neural network to reduce the number of these numerous rules and to select optimal rules when the algorithm was going to be implemented in a real environment.

The dynamic obstacles have been considered via various approaches in the literatures like the "dynamic window approach" proposed in (Fox et al., 1997). Fujii et al. (1998) presented a multilayered neural network model through reinforcement learning for collision avoidance of a mobile robot. Silva et al. proposed the MONODA (Modular Network for Obstacle Detection and Avoidance) architecture for obstacle avoidance and directing a mobile robot in unknown environments (Silva et al., 2000). This model consists of four three-layered feed forward neural network modules where each module detects the probability of obstacles in one direction of the robot. Gaudiano and Chang proposed an approach for obstacle avoidance by employing a neural network model of classical and operant conditioning based on Grossberg's conditioning circuit (Dezfoulian, 2011). Singh and Parhi (2011) introduced a real-time obstacle avoidance approach to solve each of the target seeking, obstacle-avoidance, and wall-following tasks using separate neural networks. In their approach, based on certain criteria one of the networks is selected at each time step to control the mobile robot allowing it to move safely in a crowded real world and unknown environment to reach a specified target while avoiding static as well as dynamic obstacles. A fuzzy (Sugeno) based navigation was further presented by Zavlangas and Tzafestas (2003) for an omnidirectional mobile robot. An automatic fuzzy rule generation system for obstacle avoidance was developed by Castellano et al. (1997) for effective navigation. The navigation system in an unstructured static and dynamic environment was presented using fuzzy logic which avoids the problems of navigation such as the continuous making of loops, backtracking, dead-end traps (U-shaped, maze, snails), steering from narrow passages, curved trajectory (Patle et al., 2019).

3. Materials and Methods

The focus of the study is mainly on the quality of the proposed path and on a robust and error-free navigation. The optimal path is first designed by the WOA optimization technique. WOA is a kind of global optimization technique employing both exploration and exploitation processes. This method has been already tested in solving multiple optimization problems (Mirjalili and Lewis, 2016) and in its binary format it has been applied to feature extraction and classification of many well-known data. WOA is good at local optimum avoidance and it has suitable convergence behavior. Despite of adaptability of this method to variabilities, it is needed to plan paths repeatedly for a robot navigating inside a large scale factory. The proposed method first tries to find the overall path from the initiation to the destination using a successful cost efficient optimization metaheuristic algorithm. The problem is that the generated path is partially linear and is not traced continually all over the environment. To follow such paths, a robot may make sharp turns suddenly or ends up stopping suddenly or considerably slow down while trying to select the best possible alternate trajectory among the potential candidates. Depending on the kinematics of the robot, it might be difficult or even impossible to execute such paths, besides that, marginal non-safe paths are also not suitable for robots delivering delicate, precious, or dangerous items. Hence a path optimization algorithm is further needed in order to make the generated path smoother. Once the WOA candidates a sequence of points and delineates segments of path, an RBF network is trained by a number of points candidate by the WOA. Subsequently, in order to scape form occlusions occurred by obstacles, a local path planner based on fuzzy inference system is included. The flowchart of the proposed Path Planning Algorithm is shown in Figure1. The proposed hybrid optimization planner has several advantages: elimination of instability in control, smooth performance in narrow corridors or ragged pathway and being able to generalize the ability in new situations.

3.1 Kinematics model of wheeled mobile robot

An agent is simply defined as a unicycle type mobile robot for which the kinematic model is:

$$\begin{cases} \dot{x} = v\cos(\theta) \\ \dot{y} = v\sin(\theta) \\ \dot{\theta} = \omega \end{cases} \qquad (1)$$

$$v_R = v + \frac{L.\omega}{2}, v_L = v - \frac{L.\omega}{2}$$

where x and y are the coordinates of the position of the mobile robot. θ is the orientation of the mobile robot with respect the positive direction X-axis. v is the linear velocity and ω is the angular velocity. Dead-reckoning is used to estimate the position and orientation of a robot during motion. The dead- reckoning method is based on the previous position and orientation at time $t = t_k$ to calculate the next position and orientation at $t = t_k + 1$. Using the Euler approximation, equation (1) becomes:

$$\begin{cases} x(k+1) = x(k) + v(k)T_s \cos(\theta(k)) \\ y(k+1) = y(k) + v(k)T_s \sin(\theta(k)) \\ \theta(k+1) = \theta(k) + \omega(k)T_s \end{cases} \quad (2)$$

3.2 Whale optimization algorithm

The social behaviour of whales, specifically the way they hunt, are their unique characteristics. Humpback whales have special behaviours in feeding and hunting. Whales create distinctive circular bubbles around the prey (bubbles can be upward spiral or double loops) to entrap the prey and hunt it. This method is called bubble net hunting. In the whale optimization algorithm, whales' spiral bubble manoeuvre has been modelled mathematically (Mirjalili and Lewis, 2016).

In WOA, an initial population of artificial whales is randomly distributed in the searching space. Each artificial whale (which is, in fact, a solution for the problem being solved) includes a position vector and is defined by a vector of length *n* representing the number of parameters that are going to be optimized. At each iteration of the optimization process, the first stage is to select the best solution. The closer the whale to a prey, the higher the probability of its selection. The distance metric plays a decisive role and the best solution is a whale that has the least distance to the prey. Having selected the best agent, other whales update their positions towards the best search agent. In this algorithm, the position of all whales is updated using three mechanisms: encircling the prey, exploitation and exploration as formulated below.

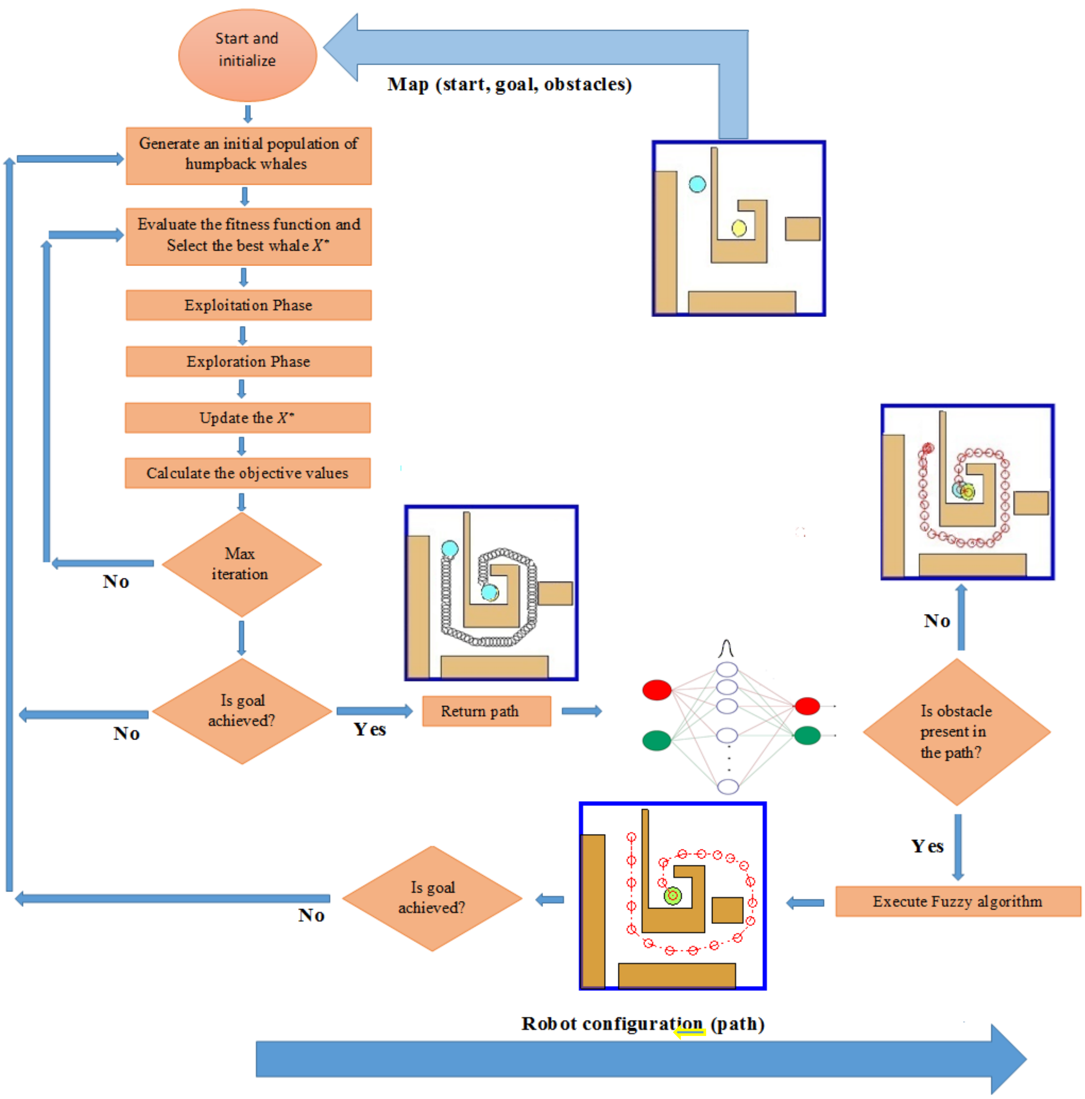

Fig. 1. Flowchart of the proposed hybrid algorithm

In the exploration phase, whales search for the prey (i.e. the best response) randomly and update their positions according to other whales. To consider this behaviour in the WOA model, the whales search for a new prey by moving randomly in the search space according to their distance from each other. Position of each search agent is updated in the exploration phase according to a randomly selected agent. This behaviour can be formulated as

$$\vec{X}(t+1) = \overrightarrow{X_{rand}} - \vec{A}\cdot\vec{D}\vec{D} = \left|\vec{C}\cdot\overrightarrow{X_{rand}} - \vec{X}\right| \quad (3)$$

In these equations, X is the position vector stands for each whale and $Xrand$ is a random position vector. Symbol | | represents the absolute value. A and C are coefficient vectors calculated as follow:

$$\vec{A} = 2\vec{a}\cdot\vec{r} - \vec{a} \qquad \vec{c} = 2\cdot\vec{r} \quad (4)$$

Where a is a parameter that decreases linearly over a course of iterations and r is a random vector in the interval [0;1]. It was suggested in (Mirjalili and Lewis, 2016) to set elements of A randomly larger than 1 or smaller than -1 (A>|1|) to make the search agent move far away from a reference whale.

Now let $X*$ to be the position vector of the best solution found so far. Each agent can update its position in adjacent to the best existing response (near to the prey) simulating the mechanism of recognizing and encircling the prey. In the exploitation phase, whales try to hunt the best prey through the bubble-net attacking process. A prey is hunted at a two-step process including shrinking encircling mechanism and spiral updating of the position. Shrinking encircling mechanism happens following encircling the prey by decreasing the value of a in equation 4. This means making smaller circles around the prey. Afterwards, spiral updating of position is accomplished and then the whales attack the prey by the end of the bubble-net strategy. To do the spiral updating, a spiral equation which mimic the helix-shaped movement of humpback is constructed based on the distance between the whale and the prey. This equation can be formulated as below simulating the position of whale over time.

$$\vec{X}(t+1) = D'.e^{bl}.cos(2\pi l) + \overrightarrow{X^*}(t) \quad (5)$$

In this equation, $D' = \left|\overrightarrow{X^*}(t) - \vec{X}(t)\right|$ is the distance between the prey and the i^th^ whale, b is a constant value, and *l* is a random number in the interval [-1 1]. In summary, when humpback whales reach around a prey, they swim simultaneously inside a small circle and along a spiral path (i.e. both mechanisms of encircling and making spiral bubbles are simultaneously in progress). To update the whales' position with this algorithm, probability of choosing one of the mechanisms of shrinking encircling and the spiral movement is 50%. This process is modelled as below:

$$\vec{X}(t+1) = \begin{cases} \overrightarrow{X^*}(t) - \vec{A}*\vec{D} & if \quad p < 0.5 \\ \vec{D}'\cdot e^{bl}\cdot\cos(2\pi l) + \overrightarrow{X^*}(t) & if \quad p \geq 0.5 \end{cases} \quad (6)$$

p is a random number in the interval [0 1]. Subsequently, the position of each artificial whale is updated during both exploitation and exploration phases.

The following steps should be taken to implement the WOA.

1. Making an initial population of artificial whales (in the optimization problem, each whale represents a solution for the problem being solved)
2. Selecting the best whale according to its distance from the prey (evaluating each solution and determining the fitted one)
3. Updating the position of the whales by three processes including
   - Searching for prey (exploration)
   - Encircling the prey
   - Spiral bubble net attaching (exploitation)
4. Go to step 2 if the termination condition is not satisfied.

3.3 Radial basis function model

Artificial neural networks are efficient model-free learnable structures that are able to handle large-scale problems such as those occur in complicated environments including continuous states. Neural networks with their parallel and distributed nature of processing seem to provide a natural solution for robot navigation. When NNs are used in combination with other optimization algorithms, it seems to work better owing to their parallel and distributed nature in one hand and the quality of convergence and speed of implementation on the other hand. They have unique capabilities such as self-education and self-regulation as well as generalization which help them not requiring any information of the relations within the input data. In this work, a well-known network structure, i.e. radial basis function, is employed for path planning improvement. This network is suitable for interpolation, function approximation, time series modeling, system identification and control. The structure of network is described briefly here. An incoming vector $X = [X_1, X_2, \ldots, X_n]^T$ is mapped over the radial basis function at each hidden node. The output layer yields a vector y by linearly combining the outputs of the hidden nodes and produces the final output as

$$y_n = \sum_{i=1}^{N} w_i \emptyset_i(X_n) \quad (7)$$

where, w_i is the weight of *i*th center, $\emptyset$ is the activity of the *i*th radial basis function, and *k* is the total number of hidden nodes. In this work, normalized Gaussian function has been selected owing to its radial symmetry and better smoothness.

$$\emptyset_i(X_n) = \exp\left(-\frac{\|X_n - C_i\|^2}{2\sigma_i^2}\right) \quad (8)$$

In this equation, $\| \ \|$ denotes a norm that is taken to be Euclidean and C_i and σ_i denote the center and spread width of the *i*th node respectively. To estimate these parameters, a two-step learning strategy is often adopted; (1) a nonlinear iterative process (a gradient descent procedure) to determine basis functions' position and width and (2) a phase of subset selection and regularization procedure for weights' adjustment (Raiesdana and Hashemi Goplayegani, 2013).

3.4 Fuzzy obstacle avoidance

Fuzzy logic is based on Linguistic rules trying to simulate the human decision making and provide concepts that cannot be expressed explicitly. Fuzzy logic provides a systematic framework to treat uncertainties and imprecise knowledge. An advantage of this approach is that it does not require a precise analytical model of the environment and uses linguistic terms and rules from a common natural language. However, the need for determination of all possible scenarios in terms of rules and knowledge base is a disadvantage of this method because of the lack of self-organization capability. Provided of detecting any obstacles in front of the robot, the planner algorithm switches to the fuzzy- based obstacle avoidance algorithm. In general, an obstacle avoidance system should satisfy two conditions. The system must firstly be able to guarantee that output values identified as likely to cause a collision with an obstacle do not occur. Secondly the procedure should not affect the navigation decisions taken by the navigation controller if this is not necessary to avoid a collision.

Fuzzy logic inference system comprises four main Parts: fuzzifier, inference engine, rule base and deffuzifier. Fuzzifier implements measurements of input variables (input signals), scale mapping and fuzzification. All sensed signals are scaled, and fuzzification means that measured signals (crisp input quantities) are converted into fuzzy quantities (linguistic variables). This conversion is achieved using membership functions. Membership function has a value between 0 and 1, and it specifies degree of belongingness of a magnitude to a fuzzy set. If it is unconditionally certain that the quantity fits to fuzzy set, then its value is 1, but if it is absolutely certain that it is no to this set then its value is 0. The outputs of activated rules are combined by fuzzy logic operations to control velocities and steering angle of a robot.

4. Results

All algorithms have been implemented with MATLAB 2018b software within a system with corei7 processor, 8GB of RAM and 2GB of graphical memory. The planner starts based on the information related to the position of goal and robot as well as the existing obstacles. A GUI based simulation platform was created in order to test different environments and scenarios. A graphical view of the GUI has been shown in figure 2.

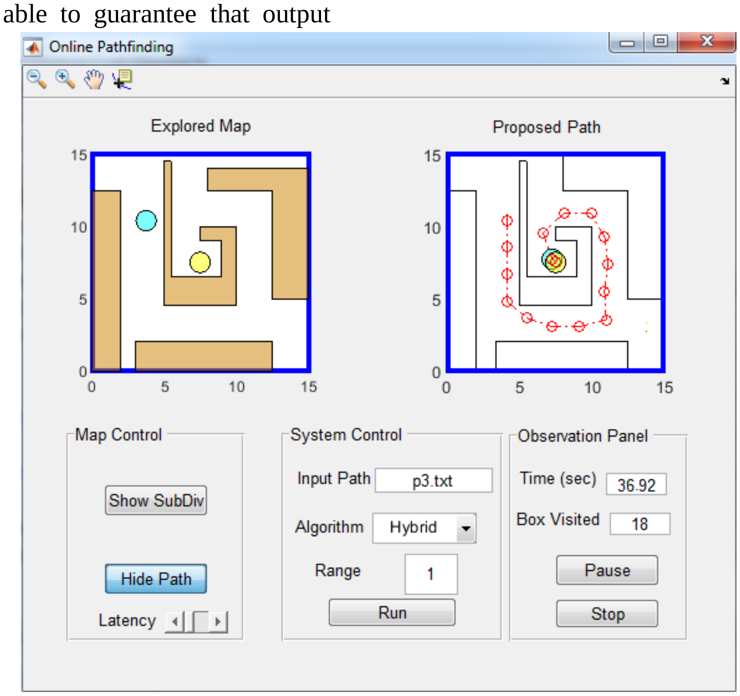

Fig. 2.The GUI platform for online path planning

The distance to the target should be reduced and the angle should be minimized over the time so that an optimum navigation task can be achieved. For simplicity, the robot workplace is described as a path in the global coordinates

O_{XY}. A coordinate transformation is initially accomplished to locate the new X'-axis to coincide with the line Start–Target when Start–Target intersects the X-axis. The corresponding transformation formula is as follows.

$$\begin{bmatrix} x' \\ y' \end{bmatrix} = \begin{bmatrix} cos\psi & -sin\psi \\ sin\psi & cos\psi \end{bmatrix} \times \begin{bmatrix} x \\ y \end{bmatrix} + \begin{bmatrix} x_{Start} \\ y_{Targ} \end{bmatrix} \quad (9)$$

Where ψ is the angle of anti-clockwise rotation from the horizontal X-axis to the line Start–Target (the connecting line which crosses from center of the robot to the center of the target), (x_{Start}, y_{Targ}) is the Start point in the coordinates O_{XY}, and (x', y') is the transformation of (x, y) in the new coordinates. For planning a movement path, the Start–Target line is divided into *n+1* equal segments by *n* points. After drawing *n* vertical lines through these points in turn, a set of parallel lines were denoted $\{l_1, l_2, \dots l_n\}$. A complete path $\{p_1, p_2, \dots p_n\}$ can be constructed by sampling vertical lines of $l_1, l_2, \dots l_n$ randomly. Hence, the robot path planning problem is transformed into optimizing the following set of points $\{Start, p_1, p_2, \dots p_n, Target\}$. Notably this path is collision free; meaning that each point in this path is not covered by obstacles and each line among the set does not intersect with obstacles (Dao et al., 2016).

The objective function is applied to find the shortest path that can be defined as the Euclidean distance between the agent and the goal point in each iteration. However, the angle between the two hypothetical lines connecting the goal point to the robot's two successive positions in each iteration (x_i and x_{i+1}) should also be minimized. Hence, the shortest path criteria and the smoothest movement criteria can both be used in an objective function as below:

$$F(p) = \sum_{i=0}^{n-1} d_i \, e^{\alpha \Psi_i} \quad (10)$$

Where d_i is the distance between p_i and p_{i+1} and Ψ_i is the angle between the line segment starting at point p_i (l_i) and the line from p_i to target. In this way, a mapping from the search space to a model of plausible path planning is constructed during the optimization process. The steps for the Whale-based optimization technique implemented in this paper to find the optimal path for the robot are as below:

1. Modeling the robot workspace including shape and position of obstacles and the robot's start and target position. The environment map has 100*100 cells. Obstacles can be rectangle, triangle or circle distributed randomly.

2. Mapping search agents to a model of robot planning in each iteration during optimization. The solution vector is $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \dots, \boldsymbol{x}_n)^T$ where n (number of points) changes in between 10 to 50 depending on the complexity of path.

3. Generate 100 humpback whales (each specified by a position vector X) distributed randomly in the search space

4. WHILE the maximum number of iterations has not been reached (i.e., i <=I_{max}) DO

If (A<1) Update the global best position Eq. (3)

Else Update the positions of agent Eq. (6)

Take the best position of agent

Calculate the objective values of each agent by Eq. (10)

Store all feasible particles and Update the feasible archive;

Increase the loop counter, i = i + 1

Output optimal results

4. Send information of the generated path (in terms of positions for p_i) to guide the robot motions toward the target position.

Parameters of WOA method were selected as below for simulations. The size of whales' population was equal to 100. Maximum number of optimization iteration varies between 70 and 150 owing to the simplicity/complexity of the environment. Some other parameters for the WOA implementation were seen in table 1. In order to show the convergence behaviour of WOA in the path planning issue, a simple environment with fixed concave shaped obstacles was generated in the Matlab environment. The required movement information (x and y coordinates for points p_i) were extracted with optimization process over iterations to indicate the movement path. The convergence behaviour of the optimization algorithm was plotted as a measure of mean square error (MSE) in figure 3. As it can be seen the convergence rate of WOA has being accelerated as iteration increases. To analyse the convergence behaviour, it can be stated that although the WOA might not found the global optimum soon (i.e. at initial iterations), but it could converge rapidly towards the optimum when enough iterations were elapsed. This probably relates to the adaptive mechanism of WOA that makes it look for the promising regions of the search space at the initial steps of iteration and subsequently helps it to converge to the optimum performance very fast and sharp.

Table 1
Setting for the whale optimization algorithm

Parameter setting for WOA	
Population size	100
a in equation 2	[0,2]
b in equation 3	1
l in equation 3	[-1,1]
maximum number of iterations 100	

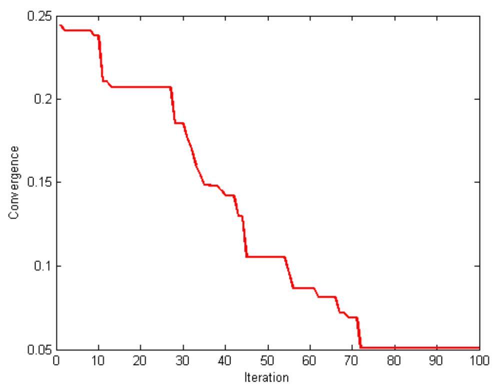

Fig. 3. Performance evaluation of WOA optimization in terms of error

An example of path planning has been shown in figure 4. A number of simple-shape obstacles were distributed randomly in the environment of the robot working space. As it can be seen, the optimization algorithm could find a suitable path among the existing obstacles after ~70 iterations. Subsequent to generation of path segments, a matrix has been constructed by sampling the paths generated by the WOA algorithm in order to access enough data for training the RBF network. The network was trained with each row of this matrix as a possible robot position in the environment. The corresponding output for each input element ([x y]) was the vector of the next robot position. The number of Gaussian basis functions was chosen to be equal to one fifth of the whole data. Normally, designing and training of an RBF network includes three sections: computing the widths σ_i, adjusting the centers μi and adjusting the weights ωi. Width is fixed according to the spread of the centers as $\sigma = d/\sqrt{2h}$ where h is the number of centers and d is the maximum distance between the chosen centers. They have been set to 0.5 and 50 respectively. The smaller the value of d results in smaller width of RBF; it makes the base function more selective in the simulation. The learning rate of network was set to 0.001 and the target error was set to 0.05

Simulation results were shown in Figure 5. The generated path proposed by the WOA was plotted in figure 5a. Figure 5b shows distributed Gaussian functions utilized for training of the RBF function and the smoothed path using the RBF model has been plotted for two extreme values of sigma (0.5 and 2) in figure 5c.

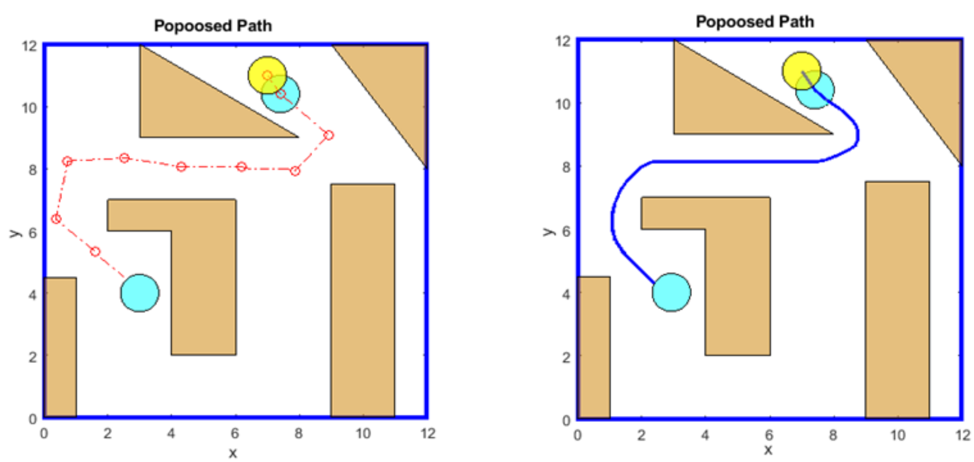

Fig. .4 A sample path planning task with WOA (left) and WOA+RBF (right)

It can be observed in figure 5c that the large value of σ can make smoothness of curve however at the cost of precision. Furthermore, the approximation using RBF with large value of σ will have a few peaks at dense regions resulting in a non-smooth observation.

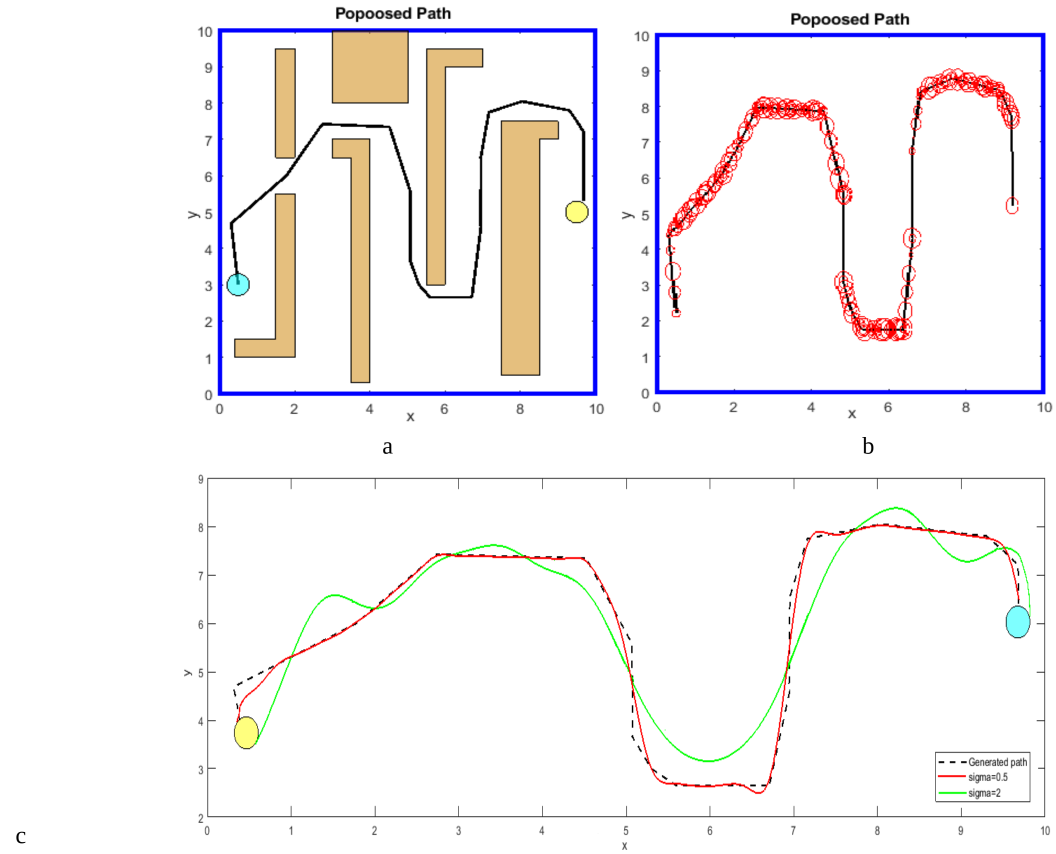

Fig. 5. (a) An optimal path generated WOA for a static environment was shown in black, (b) the RBF basis functions centered on data points and used for training are shown by red circles, (c) the smoothed paths proposed by the trained RBF are shown for two values of sigma (0.5 and 2) indicating the RBF approximation capability.

In order to evaluate the role of the RBF network in the proposed hybrid architecture, three metrics including distance to the generated path, smoothness and security were computed. Distance between trajectories was calculated as the mean square of point to point distances averaged over the whole path from start to the end. On the other hand, the smoothness of the robot movement is measured as a function of the curvature. For curves in the x-y plane, the curvature, k, at any point $(x_i, f(x_i))$ across a trajectory is given by:

$$k\big(x_i, f(x_i)\big) = \frac{f''(x_i)}{(1+(f'(x_i))^2)^{3/2}} \qquad (11)$$

Smoothness of curvature is defined by the square of the change in the curvature (k) of the trajectory with respect to the time, integrating along the length of the trajectory and normalized by the total time t (Ceballos et al., 2010)

$$Smooth = \frac{\int_0^l \left(\frac{dk}{dt}\right)^2 ds}{t} \qquad (12)$$

Finally, the security metric expresses the safety of robot when travels along a path, taking into account the distance between the robot and the obstacles in its path. Mean distance to obstacles is measured giving an idea of the risk taken through the entire mission in terms of the proximity to an obstacle

Table 2

Assessing the role of sigma for RBF functions in path smoothing

RBF function parameter	Distance to WOA-based path	smoothness	security
$\sigma = 2$	0.82	0.73	0.19
$\sigma = 1$	0.68	0.52	0.34
$\sigma = 0.5$	0.42	0.46	0.38

At the next phase, the performance of the local obstacle avoidance planner was evaluated. The characteristics of this fuzzy inference system (FIS) is that it is based on minimum local information. Besides that, the minimal fuzzy rules are defined for this system. Implementation details for the fuzzy-based planner are as follows. There are three inputs to the fuzzy logic system and one output. The inputs are distance to obstacle d_j, the angle to the nearest obstacle ε_j and the angle to the goal γ_j. The variable d was partitioned in the universe of discourse [0 10m] which was defined by five triangular membership functions: very small (VS), small (S), medium (M), far (B), very far (VB). The number of fuzzy sets to fuzzify the angle to goal [1] and the angle to nearest obstacle [2] were chosen to be seven and nine, respectively. Here, triangular and trapezoidal functions were utilized to describe each fuzzy set, which allow fast computation essential under real-time conditions. On the other hand, the output was not partitioned into fuzzy sets, but consisted of crisp values. The output variable was the changing of the heading angle (Δθ) set to be nine singleton membership functions: left very big (LVB), left big (LB), left small (LS), left very small (LVS), N, right very small (RVS), right small (RS), Right big (RB), Right very big (RVB). The membership functions for each input and the output variable are illustrated in figure 6.

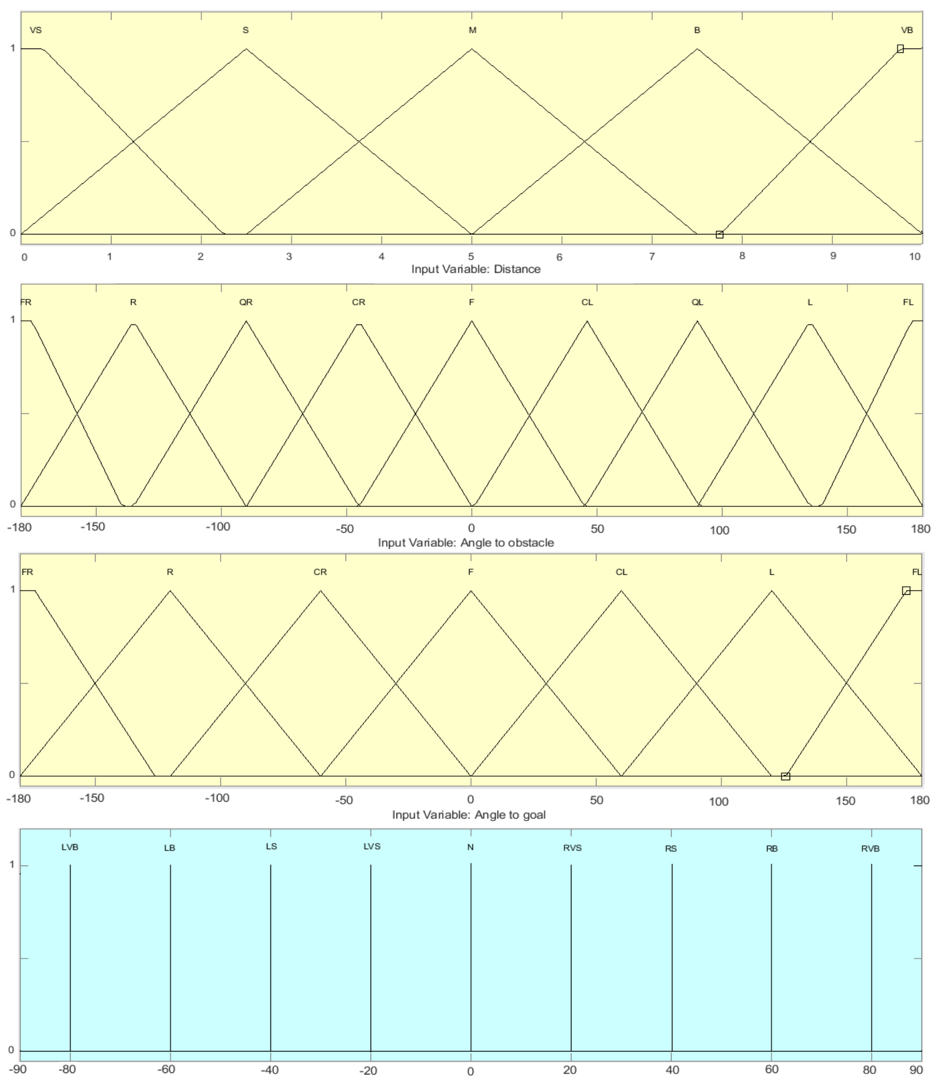

Fig. 6. membership functions for input and output variables including (a) distance to obstacle; (b) angle between robot and goal position; (c) angle between robot and obstacle and (d) steering command

1.The universe of discourse[−180∘ 180∘], membership functions are far right, right, close right, forward, close left, left, far left

2.The universe of discourse[−180∘ 180∘], membership functions are far right, right, quite right, close right, forward, close left, quiet left, left, far left

When the input and the output variables and membership function were defined, the fuzzy rules were presented using linguistic variables. Generally, rules of an FIS system is in the form of: IF ⟨antecedents⟩ THEN ⟨conclusions⟩. The rules for obstacle-avoiding can be written as sentences with three antecedents and one conclusion. An example is:

If (d is VS) and (ε is FR) and (γ is FL) then (Δϑ is LVB)

The rule base of this fuzzy inference system included 286 rules and the Sugeno-type fuzzy model was chosen to implement the FIS in Matlab. To calculate the fuzzy intersection the product operator was employed and the final output of the unit was given by a weighted average over all rules.

The final test in this work was conducted to evaluate the system's obstacle avoidance behavior. Different kind of uncertainties may exist for a mobile robot in an unknown environment. The main clue to solve the real-time moving obstacle avoidance problem is how to deal with the mobility of the obstacle. There are techniques to describe the dynamical behavior of obstacles such as employing probabilistic models to estimate the speed and direction of moving obstacles or space-time coordinate construction in which an extra dimension (time) is added to the space and converts the dynamic obstacle avoidance problem to stationary problem. Owing to complexity of such methods which induces additional computational load to system, the strategy of switching to local fuzzy-based planner in the vicinity of obstacles sounds logical.

Consequently, a dynamic path planning test was designed to illustrate the performance of the fuzzy-based planner. Results with different seniors of this test are illustrated in figure 7. To solve the navigational task designed in this test, the hybrid path planner was initially recruited the global metaheuristic iterative procedure (WOA+RBF) to find an optimal path. As it can be seen in figure 7a, a smooth path has been found and executed with marginal safety owning to conservative settings made for the planner and optimizer. Afterwards, obstacles started to move slowly in one direction with velocity of 0.3 m/s. As it can be seen in figure 7b, the WOA+RBF planer is robust and well-sufficed to avoid collision to obstacles. In this case, the robot path has been trivially changed due to the movement of obstacles. At the next step, a scenario was designed to show that the robot can still rely on its global planner when moving obstacles with predictable uncertainties exist. In this scenario, the obstacles moved faster albeit with known patterns during the navigation of robot. The robot motion speed was 0.5 m/s and obstacles moved with approximately the same speed. Since, the obstacles' speed and direction of movement were known a priori in this test, it was possible to introduce the information into the optimization algorithm; i.e. during the WOA optimization iterations. The position of robot and obstacles was estimated at each iteration and the points that have been candidate for path planning were not chosen from those occupied by obstacles and also by their neighbors. As it can be seen in figure 7c, a smooth non marginal path has been found among the moving obstacles. Noteworthy, the fuzzy planner has not been activated and no computational load has been added when it was not possible. Ultimately, an obstacle in the environment started to alternatively move back and forth which necessitated the activation of fuzzy based planner (in figure 7d). Using the fuzzy rule-base system and making soft online decisions, the robot can subsequently cross the obstacle (green circles in figure 7d) and follow the path toward the chosen goal.

Eventually, the metrics introduced throughout the text were computed to evaluate the performance of each part of the hybrid algorithm. Comparison results were summarized in figure 8. It can be seen the smoothness of the WOA optimization algorithm, as it was implemented in this research, is less and some post processing algorithms are needed to improve the performance path planning. The RBF could improve the generated path smoothness considerably. Furthermore, although it seems that bringing together iterative and inference methods may cause complexity and require specific time and memory requirements, the algorithm run time has increased 23%.

As a final evaluation on the performance of the proposed method, a comparison has been made between the proposed method and some state of the art methods. At is evident, a quantitative comparison was not possible since the implementation platforms as well as hardware/software requirements were not identical. Besides that, the designed environment and scenarios were almost subjective and dependent to the software chosen for simulation. Hence, a qualitative comparison was preferred for evaluation. The results of comparison have been tabulated in table 3 indicating the acceptable performance of the proposed method.

Table 3
Comparison of methods for path planning; the hybrid methods are highlighted

properties	(Masehian and Sedighizadeh, 2013)	(Oscar et al., 2015)	(Kumar et al., 2018)	(Patle et al., 2016)	(Mohanty and Parhi, 2015)	Proposed method
Iteration	-	11 (generation)	100	-	100	[70 150]
Dynamic obstacle avoidance	no	yes	no	no	no	yes
Mean run-time	31.47s	14.5s	-	14s	42.09s	47.5s
Practically feasible for industrial robot applications	no non-smooth, marginal	Yes BPF is for complex environment	yes smooth-marginal	maybe non-smooth	no non smooth, simple scenarios	yes smooth, non-marginal
Kinematic analysis	no	no	no	yes	no	no
Real experiments	no	no	yes	yes	yes	no

BPF: Bacterial Potential Field

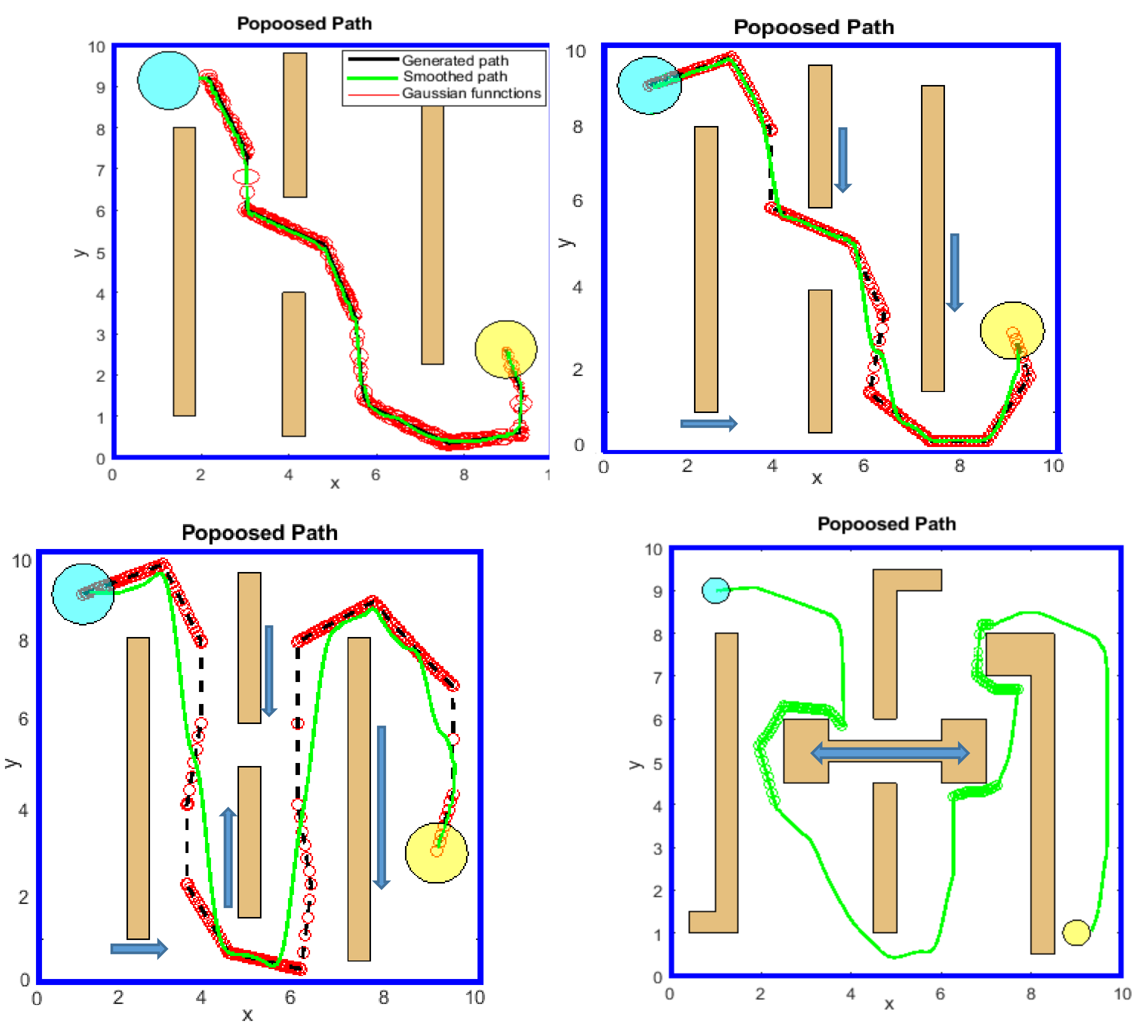

Fig. 7. Path planning in a dynamic environment. a) Obstacles were fixed and the path was generated with WOA+RBF (σ=0.5). b) Obstacles' moved slowly such that the fuzzy based planner has not been activated. The generated path has been partially affected compared to static task. Blue arrows shows the moving direction. In this simulation σ=1. c) Obstacles' moving speed is higher but the global path planner could still generate the optimal path (σ=2). d) The fuzzy planner has been activated and tackled the horizontal obstacle which was moving back and forth. Run time for this test was 42.34s.

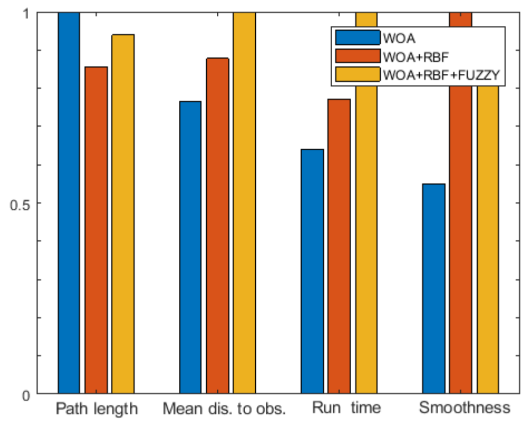

Fig. 8. Comparison of performance metrics (path length, mean distance to obstacle, run time and smoothness) for WOA, RBF and Fuzzy algorithms. Path length and mean dis. to obs. were measured in meter and run time in second. All the metrics were scaled and normalized to be plotted in one figure.

5. Conclusion

The field of robotics may be more practically defined as the study, design and use of robot systems for manufacturing. Industrial robotics play a key role in automation which require specific considerations for designing and planning. The key challenge in autonomous robot navigation is robust planning of paths and avoid the existing static and dynamic obstacles. To be safe in industrial environments, a smooth and non-marginal path should be planned and followed.

In this paper, the path planning issue for an industrial mobile robot (which works in a partially unknown environment) was addressed via a hybrid multi-objective algorithm. The advantages of the proposed hybrid method include solving of local minimum problems with minimum risk as well as planning global goal-directed navigation with acceptable accuracy, smoothness and run time. In addition, being assisted by fuzzy rule-base inference system, the robot would not get stuck in local minima. Overall, by an embedded iterative-inductive algorithm, the robot can make real-time autonomous decisions to move towards the goal and avoid any static or dynamic obstacles in its path. On the other hand, this method is computationally optimum as its complexity (which depends on the WOA searching mechanism, the structure of the RBF network as well as the number of fuzzy rules) was considerably minimized due to the specialized consideration made in its design and implementation.

The hybrid algorithm could be employed for all unstructured environments including diverse shaped obstacles ranging from simple to critical. Sudden environmental changes such as addition/remove of obstacles as well as sudden changes in the position of robot and/or goal are also tolerable. In overall, the algorithm is robust to environmental changes and can tackle obstacles of different kinds. This causes the robot not to waist its energy and time when turning its course of moving abruptly and making sharp turns. For future works, we are pursuing to develop bio-inspired path planning strategies based on human spatial learning and memory. Furthermore, working on algorithms to estimate and predict the future motion of obstacles and/or goal will improve the planning performance. Finally, real test using standard hardware/software is required for validating the proposed algorithms.

Acknowledgements

The author gratefully thanks to Dr. Esmaeilzadehha for his collaborations.

References

AbuBaker, A. (2012). A novel mobile robot navigation system using neuro-fuzzyrule-based optimization technique. *Research Journal of Applied Sciences, Engineering and Technology*, 4, 2577-2583.

Barbehenn, M. (1998). A note on the complexity of Dijkstra’s algorithm for graphs with weighted vertices. *IEEE Transactions on Commuters*, 47(2), 263.

Borenstein, J., Koren, Y. (1991). The vector field histogram-fast obstacle avoidance for mobile robots. *IEEE Trans Robot Autom*, 7(3), 278–288.

Castellano, G., Attolico, G., Distante A. (1997). Automatic generation of fuzzy rules for reactive robot controllers. *Robot Autonomous System*, 22(2),133-149.

Ceballos, N.D.M., Valencia, J.A., and Ospina, N.L. (2010). Quantitative performance metrics for mobile robots navigation. *Mobile Robots Navigation* 15, 486:500.

Dao, T.K., Pan, T.S. and Pan, J.S. (2016). A multi-objective optimal mobile robot path planning based on whale optimization algorithm. *IEEE 13th International Conference on Signal Processing.*ICSP2016, 337- 342.

Deng, X., Milios, E. (1996). Landmark selection strategies for path execution. *Robotics and Autonomous Systems*, 17, 171-185.

Dezfoulian, S.H. (2011). A generalized neural network approach to mobile robot navigation and obstacle avoidance. MSc. Thesis, University of Windsor.

Dugarjav, B., Kim, H., Lee, S-G (2015). Online cell decomposition with a laser range finding for coverage path in an unknown workspace. *International Journal of Mechanical and Production Engineering*, 3(5), 18-24.

Fox, D., Burgard, W., Thrun, S. (1997). The dynamic window approach to collision avoidance. *IEEE Robotics & Automation Magazine*, 4, 23–33.

Fujii, T., Arai, Y., Asama, H., Endo, I. (1998), Multilayered reinforcement learning for complicated collision avoidance problems. *IEEE International Conference on Robotics and Automation*, 2186–2191.

Ganapathy, V., Yun, S. C., and Ng, J. (2009). Fuzzy and neural controllers for acute obstacle avoidance in mobile robot navigation. *IEEE/ASME International Conference on Advanced Intelligent Mechatronics*, 1236-1241.

Ge, S.S., Cui, Y. J. (2000). New potential functions for mobile robot path planning. *IEEE Trans. Robot. Autom*. 16(5), 615–620.

Groen, F.C.A. (2000). A virtual target approach for resolving the limit cycle problem in navigation of a fuzzy behavior-based mobile robot. *Robotics and Autonomous System*, 30(4), 315–324.

Huji, D., Croft, E.A., Zak, G., Fenton R.G., Mills, J.K., Benhabi, B. (1998), The robotic interception of moving objects inindustrial settings: strategy developmentand experiment. *IEEE/ASME Transactions On Mechatronics*, 3, 225 – 239.

Joshi, M. M., Zaveri, M. (2010). Neuro-fuzzy based autonomous mobile robot navigation system. *11th International Conference on Control Automation Robotics & Vision (ICARCV)*, 384-389.

Kumar, A., Kumar, Priyadarshi Biplab, K., Parhi Dayal, R. (2018). Intelligent navigation of humanoids in cluttered environments using regression analysis and genetic algorithm. *Arabian J Sci Eng.* 1-24.

Kumar, M. P., Rishna K. P., Dayal Manfis, R. P. (2014). Approach for Path Planning and Obstacle Avoidance for Mobile Robot Navigation. *Advances in Intelligent Systems and Computing*, 248(1), 361-370.

Li, T. and Latombe, J. (1997). Online manipulation planning for two robot arms in a dynamic environment. *The International Journal of Robotics Research*, 16 (2),144-167.

Masehian, E. and Sedighizadeh, D. (2013). An improved particle swarm optimization method for motion planning of multiple robots springer. *Tracts in Advanced Robotics*, 83, 175-188.

Mirjalili, S., Lewis, A. (2016). The whale optimization algorithm. *Advances in Engineering Software*, 95, 51–67.

Mohanty, P. K., Parhi, D. R. (2013). Path planning strategy for mobile robot navigation using MANFIS controller. *Advances in Intelligent Systems and Computing*, 247, 353-361.

Mohanty, P. K. and Parhi, D. R. (2015). A new hybrid intelligent path planner for mobile robot navigation based on adaptive neuro-fuzzy inference system. Australian Journal of Mechanical Engineering, 13(3), 195–207.

Ng, J., Bräunl, T. (2007). Performance comparison of bug navigation algorithms. *Journal of Intelligent and Robotic Systems*, 50, 73-84.

Oscar, M., Ulises, O-R., Roberto, S. (2015). Path planning for mobile robots using Bacterial Potential Field for avoiding static and dynamic obstacles. *Expert Syst Appl*: 42(12),5177-91.

Patle, B.K., Babu, G., Pandey, A., Parhi, D.R.K., Jagadeesh, A. (2019). A review: On path planning strategies for navigation of mobile robot. *Defense Technology* 15,582-606.

Patle, BK, Parhi, DRK, Jagadeesh, A, Kashyap, S. K. (2016). Probabilistic fuzzy controller based robotics path decision theory. *World Journal of Engineering;* 13(2),181e92.

Raiesdana, S., HashemiGoplayegani, S.M. (2013). Study on chaos anti-control for hippocampal models of epilepsy. *Neurocomputing*, 111, 54-69

Ravankar, A., Ravankar, A., Kobayashi, Y., Hoshino, Y. and Peng, C. (2018). Path smoothing techniques in robot navigation: state-of-the-art, Current and Future Challenges. *Sensors* 18, 3170-3200.

Silva, C., Crisostomo, M., Ribeiro, B. (2000), MONODA: a neural modular architecture for obstacle avoidance without knowledge of the environment. *Proceedings of the IEEE-INNS-ENNS International Joint Conference on Neural Networks*, 6,334-339.

Singh, M.K., Parhi, D.R. (2011). Path optimization of a mobile robot using an artificial neural network controller. *International Journal of Systems Science*, 42,107-120.

Singh, M. K., Parhi, D. R., Pothal, J. K. (2009). ANFIS approach for navigation of mobile robots. *International Conference on Advances in Recent Technologies in Communication and Computing*, 727-731.

Tsui, W., Masmoudi, M. S., Karray, F. (2008). Soft-computing-based embedded design of an intelligent wall/lane-following vehicle. *IEEE/ASME Trans. Mechatronics*, 13(1), 125-135.

Vukosavjev, S., Kukolj, D., Papp, I., Markoski, B. (2011). Mobile robot control using combined neural-fuzzy and neural network. *12th IEEE International Symposium* *on Computational Intelligence and Informatics*, 351-356.

Wooden, D. T. (2006). Graph-based path planning for mobile robots, PhD. Thesis, School of Electrical and Computer Engineering Georgia Institute of Technology.

Zafar, M. N., Mohanta, J. C. (2018). Methodology for path planning and optimization of mobile robots: a review. *procedia computer science* 133,141–152.

Zavlangas, P. G., Tzafestas, S. G. (2000). Industrial robot navigation and obstacle avoidance employing fuzzy logic. *Journal of Intelligent and Robotic Systems*, 27,85–97.

Zavlangas, P.G., Tzafestas, S.G. (2003). Motion control for mobile robot obstacle avoidance and navigation: a fuzzy logic-based approach. *Systems Analysis Modelling Simulation*, 43, 1625–1637.

Raiesdana, S. (2021). A Hybrid Method for Industrial Robot navigation.
Journal of Optimization in Industrial Engineering, 14(1), 133-148.

http://www.qjie.ir/article_676301.html
DOI: 10.22094/JOIE.2020.1863337.1629