ISSN 1 746-7233, England, UK
World Journal of Modelling and Simulation
Vol. 15 (2019) No. 3, pp. 302-320

Two Stage Smooth Path Planning for Mobile Robot for Optimal Obstacle Avoidance Problem in Unknown Environment

Md. Abdul Alim Sheikh[1]*, Alok Kole[2], Tanmoy Maity[3]

[1] Dept. of Electronics & Communication Engineering, Aliah University, Kolkata, India
[2] Dept. of Electrical Engineering, RCC Institute of Information Technology, Kolkata, India
[3] Dept. of Mining Machinery Engineering, Indian Institute of Technology (ISM), Dhanbad, India.

(*Received December 13 2016, Accepted April 04 2019*)

Abstract. One of the most fundamental tasks to be performed by a mobile robot is to find a collision-free and smooth path to follow satisfying certain optimization criteria. However, collision free path generated by existing techniques are very spiky which the robot would get it very hard to follow practically. These are known as the non-holonomic constraints in the robot. Smooth paths are important in robotics because smooth paths are more suitable for developing continuous control algorithms to follow the paths by such non-holonomic mobile robots. The main aim of this paper is to solve mobile robot path planning problem in unknown static environment by determining collision free path that satisfies the chosen criteria for shortest distance and path smoothness. The proposed path planning algorithm consist two stages. In the first stage, an optimized collision free path from starting to goal point is generated using a Genetic Algorithm (GA) added with a penalty function. In the second stage, the collision-free path obtained from first stage is interpolated using a Radial Basis Function Neural Network (RBFNN) to generate smoother paths providing to the non-holonomic constraints. The results of GA algorithm serve as a guide for RBFNN planner. The proposed algorithm is tested on various simple and complex environments. All the paths generated were optimal in terms of path length and path smoothness. The proposed algorithm also reduces the number of steps to be taken between the starting point and the target point. We also study the behavior of the best fitness value in all paths and configurations. Comparisons with previous examples in the literature are also included in the results. Detailed simulation results show the functionality and effectiveness of the proposed algorithm in different scenarios.

Keywords: Mobile Robot, Path Planning, Static Obstacles, Genetic Algorithms, Radial Basis Function Neural Network (RBFNN), Robotic.

1 Introduction

Path-planning is an important primitive for autonomous mobile robots that lets robots find a collision-free path amidst obstacles from its starting position to its destination. Mobile robots are widely used in higher, deeper and riskier environment where there may be risk for people, such as aerospace research, the nuclear industry, and the mining industry. The primary concern for the success of any mobile robotic systems is to find a collision-free path through the robot's environment with obstacles, from a specified start position to the goal to be reached while satisfying certain optimization criteria such as distance, time, cost, and energy [38, 56]. Distance and time being the most commonly accepted criteria.

All path planning problem are classified according to two factors, 1) the environment type (i.e., static or dynamic [68]), 2) the path planning algorithms (i.e., global or local [6, 35]). The static environment is defined as the environment which doesn't contain any moving objects other than a navigating robot; while the dynamic is the environment which has dynamic moving objects (i.e., human beings, moving machines and moving robots).

* Corresponding author. *E-mail address*: alim.sheikh16@gmail.com

The global path planning algorithms requires a complete knowledge about the search environment and that all terrain should be static. Again, in global planning, the map of the environment is totally known, where, in local planning, the information of the environment is not known in advance. A path planning algorithm must ensure that if a solution is possible, it is found and returned. This is called the completeness of the algorithm [60]. The primary difficulty with the existing techniques is that, although powerful for standard path planning, they do not naturally extend to general nonholonomic planning problems. The research activity in this area has been steadily increasing over the last two decades.

This paper presents a proposal for path planning of an autonomous mobile robot in static environments. The problem of path planning is solved separately in two hierarchies.

• The first is generation of a collision free path from starting to destination point, which is solved using a GA optimization algorithm. Collision free path generated by GA is not smooth and it is represented as an array of points in 2-D space. However, GA also generates very sharp paths which the real robot would find it very difficult to follow. These are called the non-holonomic constraints in the robot. Results are drastically improved by running an additional algorithm that smoothes the path.

• The collision-free path obtained from the first stage is interpolated using a Radial Basis Function Neural Network (RBFNN) to generate smoother paths, more direct route to the goal. The results of GA algorithm serve as a guide for RBFNN planner.

We consider the case of constrained environments where the robot is represented as a point. This enables a fast implementation and understanding, without looking into the libraries for collision detection and concepts of multi-dimensional configuration spaces. The algorithm tries to identify good points in the map, which are later judges by the RBFNN algorithm by carrying out planning with the given robot and constraints. We tested the algorithm on various complex and simple paths. We also study the behaviour of the best fitness value in all the discussed paths and configurations. Through some results, we give a comparison between this strategy and methods based on road maps, potential fields, Rapidly-exploring random trees (RRT), GA and A* search. A comparative result is tabulated which shows that proposed method is practical and reliable.

The main contribution of this paper is the development of a new path planning algorithm which consists of Evolutionary Algorithm GA added with a penalty function for optimized collision-free path generation from starting to goal position and later another evolutionary algorithm RBFNN is used to generate smoother paths, more direct route to the goal, providing to the non-holonomic constraints. Two objectives were simultaneously optimized which included path length and path smoothness. We also propose a mobile robot controlling algorithm which allows eight-neighbour movements, so that path-planning can handle complicated search spaces with low complexities. The proposed algorithm also reduces the number of steps to be taken between the starting point and the target point. It meets all the major concerns, e.g. it can find the destination in a relatively short time which provides efficiency. Finally, it accurately and always finds and follows the determined path.

The rest of the paper is organized as follows. First, Section 2 highlights several techniques exist in literature, then section 3 presents the description of the problem formulation, environment, inputs and general assumptions and robotic design in this work. Section 4 provides the detailed description of the proposed approach for generating collision free smooth path using GA and RBFNN. In Section 5, a set of simulation results are presented to demonstrate the effectiveness of the proposed approach as compared with previous work. Finally, this paper is concluded in Section 6.

2 Related works

Significant number of research papers exists in the field of robot path planning in literature. There are two different approaches for mobile robot motion planning: classic and heuristic [56] as shown in Fig. 1.

Most autonomous mobile robot path planning problems are solvable using classic algorithms. These approaches are not necessarily mutually exclusive, but their combination is often used in developing more reliable paths. The classic methods are developed with variations of a few general approaches such as roadmap [30, 60], cell decomposition [10], Artificial Potential Field (APFs) [3, 42, 80], and mathematical programming [47, 56]. Well-known roadmaps include the visibility graph [75], Voronoi diagram [4, 27, 73], and sub-goal networks [71]. Improved APFs of other kinds have been studied [8, 24, 64, 69, 82]. During last few years,

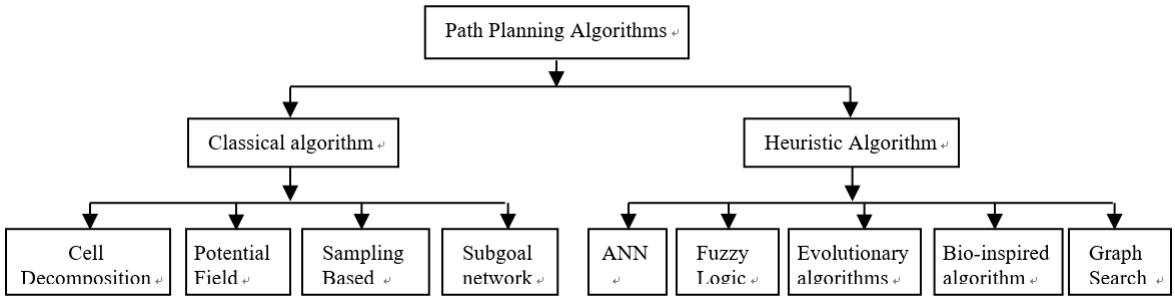

Fig. 1: Classification of Robot path planning algorithms

Rapidly-exploring random trees (RRT) [48] are widely used for developing randomized robot path planning. A comprehensive review on Classic path planning methods can be found in [35].

In the road map approach [60], feasible paths are mapped onto a network of one-dimensional lines; then a search for a desired path is conducted in such a network. However, the searched path is limited to the network, and path planning becomes a graph-searching problem. Well-known roadmaps include the visibility graph, Voronoi diagram, and sub-goal networks. The visibility graph algorithm [75] can compute the shortest distance or optimal path. This approach does not consider the mobile robot size or that a lead robot is too close to the vertex of an obstacle, even colliding with obstacles, and the computational time for path planning is too long. Voronoi diagram [73] and sub-goal network [71] algorithms are improved methods of the visibility graph. Additionally, several researchers have demonstrated that cell decomposition [10] is the simplest method for mobile robot path planning, but they are inefficient for computational memory and planning time according to the cell size. An APF [3, 42] is an important classic method for autonomous mobile robots. An APF has often represented a good quality method to achieve a fast and reactive response to a dynamic environment. However, this method has been widely demonstrated as suffering from unavoidable drawbacks which make it very likely that a robot will become trapped in a local minimum and oscillations. Another improved APF has been proposed [33] using quantum particle swarm optimization for rapid global searching and realizing optimal path planning. They employ quantum particle swarm optimization to modify the parameters of the APF to adapt to a different environment and dynamic obstacles. To address the local minima problem in the traditional APF, a method including robot regression and a potential field filling has been proposed [70, 82]. Similar methods have been proposed in other papers [83, 84]. Before calculating the resultant force that is put on an object in the potential field, they build links among closed obstacles to optimize the planned solution. Improved APFs of other kinds have been investigated [5]. They introduce the relative distance between a robot and target into a repulsive force function and modify the repulsion direction to ensure that the global minimum is at the target position.

Probabilistic Road Maps (PRM) are extensively used methods of planning [15] which may be used for planning of multiple robots. This algorithm may consist of two phases: offline phase and online phase. The offline phase deals with the learning of the provided map. The learning algorithm builds a roadmap that summarizes the various points around the map and the distances between them. The online planner uses these points and the information in the roadmap for planning. An extension of the approach is the lazy PRM proposed by Bohlin and Kavraki [7].

Rapidly-exploring random trees (RRT) [45] are widely used for mobile robot path planning purpose. They start the search process by using the source as the root of a tree-like data structure. Search proceeds by expansions of the tree structure until a goal is found. Recent works includes use of multiple RRTs and combining them to produce optimal results [65]. It is difficult to pass through narrow passages using these sampling based approaches for which a better sampling technique was proposed in [85].

However, most classic approaches, roadmaps, and cell decomposition are based on the free configuration space(C-space) concept [53]. The mentioned classic approaches suffer from many drawbacks, such as high time complexity in high dimensions and trapping in local minima, which makes them inefficient in practice. Moreover, the greater the dimensions of free C-space, the more complex the path planning problem will be.

In order to improve the efficiency of classic methods, many heuristic and meta-heuristic algorithms [55] are developed in robot path planning [56]: Simulated Annealing (SA), Artificial Neural Networks (ANN) [25, 39, 44], Genetic Algorithms (GA) [31, 34, 46, 86], Reinforcement learning (RL) [43], Directed Acyclic Graph [50], Evolutionary Algorithms (EA) [7, 11, 21, 26, 72], Particle Swarm Optimization (PSO) [18, 28, 51], Surrounding Point Set (SPS) And Former And Latter Points (PI FLP) Algorithm [32], Ant Colony (ACO) [57], Artificial Bee Colony (ABC) [17, 52], Dynamic Programming [41], Fuzzy Logic [2], Bacterial Foraging Optimization (BFO) technique [35], Fuzzy-Wind Driven Optimization algorithm [63], Stigmergy [13], Sampling-based [23], graph based methods [56], Wavelet Theory [19], Tabu Search (TS) [29], Cuckoo search algorithm [58], DJIKSTRA'S Algorithm (DA) [80], A* algorithm [29, 87], D* algorithm [22, 44]. Some others algorithms are presented [68],[78],[20],[66], [16],[43],[81],[14], [36],[62][54][59]for path planning problem. In addition, some scholars have investigated robot navigation algorithms based on ant colony optimization algorithms [26] and improved ant colony optimization [21] algorithms. Heuristic algorithms do not guarantee to find a solution, but if they do, are likely to do so much faster than deterministic methods.

Each of these above methods has its own strengths and weaknesses. However, the time complexity of all heuristic algorithms will increase greatly when a larger and more complex environment is considered. Unfortunately, even though algorithms like A* [29] and D* [44] are complete, the paths made by this algorithms lack of smoothness; the robots will often have to stop and readjust their trajectory to continue following the path with every drastic change of direction. On the other hand, for discretizing the environment, lots of solutions may be excluded, and also it could cause non smooth paths in algorithms like A*, D* and potential fields. Furthermore, in the case of potential fields, the algorithm could be trapped in a local minimum formed by concave obstacles [3]. One of drawbacks in the studies is that these algorithms are oriented toward finding the shortest paths without considering the smoothness of the paths. In essence, smooth paths are vital in robotics because non-holonomic [39] mobile robots are commonly used in practice. So a smooth path will be more suitable for such robots because for designing continuous control algorithms to follow the paths this kind of paths are more preferable.

3 Formulation of the problem

To find an optimal robot path successfully, the model of an environment should be first constructed. Here, the mobile robot path planning problem is formulated as follows: given a mobile robot and a description of an environment, we need to plan a collision free path between two specified locations, a start and goal point satisfying certain optimization criteria [6]. According to this definition, path planning problem is categorized as an optimization problem.

3.1 Environment

Path-planning requires a map of the environment. A robotic map is a representation of the world of the robot. The map represents the traversable path, obstacles. A grid-based map is used here to represent the robotic world. Each grid is marked with a colour with white and black. White represents presence of no obstacle and black represents a confirmed presence of obstacle. The grids are converted into a graph (i.e., node) first for the GA algorithm. In the graph each vertex represents grids. All grids that are accessible by other grids (irrespective of presence or absence of obstacles) are marked as edges. One such representation of the map is given in Fig. 2.

To model the problem as an optimization problem, an objective function and specification of variables of that objective function (along with their limits) have been considered. Here, a path is characterized by a fixed number of points in the robotic map. In order to create some path from this set of points, we start from the source and connect it to the first point by a straight line. The first point is connected to the second point by a straight line, and so on. At the end the last point is connected to the goal. The objective function is the length of this path in our case. A heavy penalty is added if any part of the path lies within the obstacle. The penalty is proportional to the length of the path inside the obstacle. The locations of each of these fixed number of points (both X and Y axis positions) are the optimization variables. The variable limits are such that the point lies

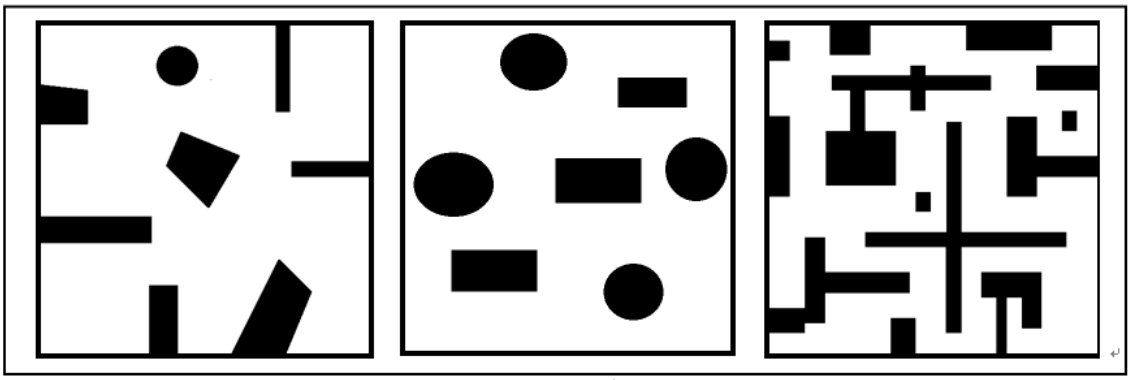

Fig. 2: Sample Maps used in this study

inside the map (lower bound 1 and upper bound as the length/width of the map for the X/Y axis). All points (both X and Y axis values) put one by one make the genetic individual used for optimization. The concept is shown in Fig. 3.

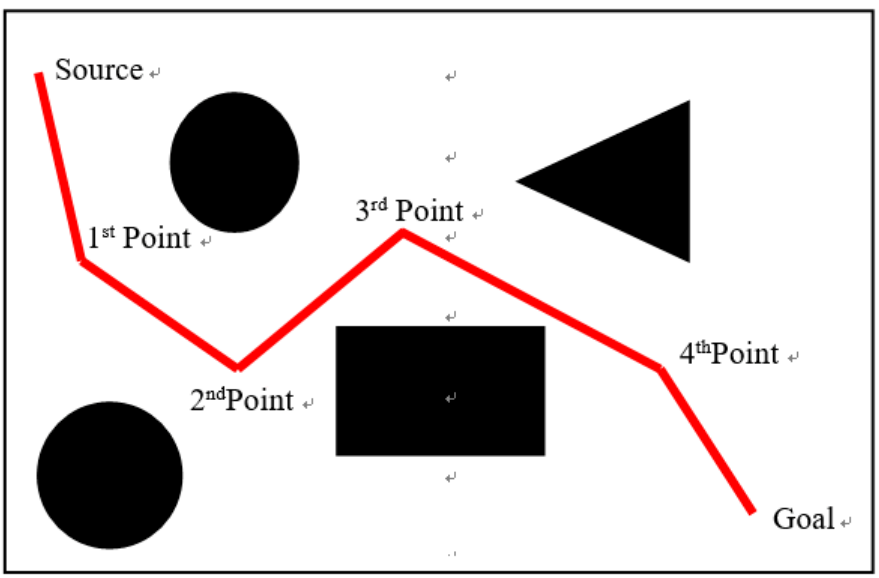

Fig. 3: Genetic Algorithm Individual Representation

Each point in the path denotes a point of turn. The total number of points is an algorithm parameter and must be equal to the maximum number of turns a robot is expected to make in the robotic map. If this number is set to too high would result in very large computational requirements.

3.2 Input

- Initial location of robot (10, 10).
- Location of Destination (490, 490).

3.3 Outcome of each step

Next step of robot towards the feasible path by avoiding obstacles.

3.4 Final output

An optimized collision free path.

3.5 General assumptions and robotic design

In order to the practical implementation of the proposed solution, following generalizations/assumptions have been made.

- The entire space of the robotic map is finite. This space is divided in a grid of size M * N.
- The obstacles are detected by the sensor which is mounted on the robot. So, the obstacles within the sensor range can be detected with reference to the coordinate system.
- For simplicity, the robot is taken as a point-sized object.
- Robot can make only a unit move in a unit time called the threshold time.
- For this problem the following movements are valid by the robot. The movements have been quantized for algorithmic purposes:
  - Move forward (unit step)
  - Move at an angle of 45 degrees forward (unit step) from the current direction
  - Move clock-wise/anti-clockwise (45 degrees)
  - Move clock-wise/anti-clockwise (90 degrees)
- It is assumed that the robot only rotates/moves in angles of a multiple of 45 degrees by rotations and forward move as given in Fig. 4.

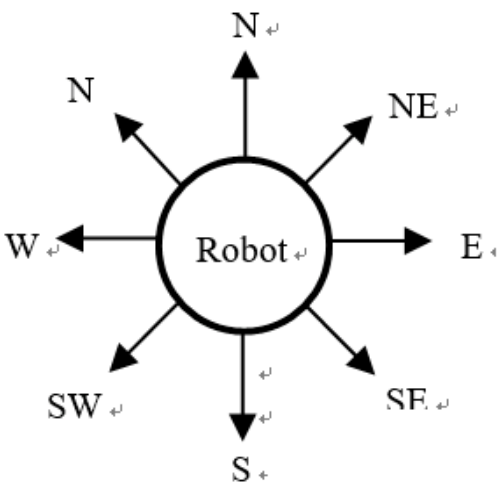

Fig. 4: The Various directions of the movement by the robot

4 Approach description

The main steps of the proposed methodology for solving the problem of robotic path planning are illustrated in this section. The basic methodology is to use a fusion of GA and RBFNN algorithm for mobile robot path planning. The algorithm first identify good points in the map, which are later judges by the RBFNN with the given robots and constraints.

4.1 Path planning using genetic algorithm

In this approach we use GA to generate a set of points in the entire robotic map. These set of points represent characteristic points that the robot might find fruitful for using it in its path from source to goal.

The fitness function of the GA is a specification of the usefulness of these points in carrying effective motion planning. To use GAs for solving the path planning problem, we considered number of steps. These steps are:

• First: Convert the search environment to a grid graph (i.e., node). Thus, the robot shall move in a step fashion on the proposed grid as they appear in the real environment.

• Second: Specify the starting and ending point where the path need to be established.

• Third: No. of points to be optimized in the path.

4.1.1 Individual representation

The primary task in the implementation of the GA is the individual representation. It is assumed that there are a total of P_i points in the map whose positions can be optimized by GA. All these points are come one after the other in the applied individual representation. Here an individual is represented by a series of points (P_i) on the robotic map. The complete individual hence becomes $< P_0, P_1, P_2, P_3, \ldots\ldots P_n, P_{n+1} >$. Here P_0 is the source and P_{n+1} is the target. Each point is a collection of x and y coordinates and is denoted by (x_i, y_i).

4.1.2 Initialization

An initial population is created with a predefined population size. The population contains number of individuals (i.e., chromosomes). Each individual represents a solution for the problem. In fact, each solution is a path between the start and end point in the search space. The initial population with size n can be presented as:

$$Initial\ Population =< P_0, P_1, P_2, P_3, \ldots\ldots P_n, P_{n+1} > \tag{1}$$

In general each is simply an integer string of length L. Each structure P_i represents a vector of node numbers in the grid which can take values of 1, 2 L (i.e., search space). Here, the map is considered as a rectangular image of size M * N grids. The map consists of obstacles that the robot must avoid collision with objects and obstacles in his way. The purpose of the algorithm is to compute the path of robot. The path of the robot P_i is a set of points in the map that results in a collision free movement. Normally, GAs individuals can contain any point value between the starting and ending point. Thus, the individual generated by GAs is in the form of:

$$< P_1, P_2, P_3 \ldots\ldots P_l > \tag{2}$$

Where, l is the number of visited node in the search space. The starting and ending point will not be shown in this individual. This is why we need to make some modification to the individual structure so that we can add the starting and ending point. The modified individual representation will be:

$$< P_{\text{start}}, P_1, P_2, P_3 \ldots\ldots P_l, P_{goal} > \tag{3}$$

These points serve as vertices of the graph.

4.1.3 Fitness function

Fitness function in genetic algorithm measures the usefulness of a solution or individual. In this algorithm the fitness function consists of the path length and the collision penalty where path i denotes the length of the sub path. A heavy penalty is added if any part of the path lies inside the obstacle, while the penalty is proportional to the length of the path inside the obstacle. A penalty proportional to the last point visited by the robot was added if the robot failed to reach the goal.

Let L_i be the last point reached by the robot R. Let Ti be the tile of travel of the robot. G_i is goal of any of the map i. The locations of each of these fixed number of points (both X and Y axis positions) are the optimization variables. Fitness is given by

$$Fit = \sum_{i=1}^{N} path_i + \sum_{i=0}^{N} \text{Pen}\,(L_i - G_i) \tag{4}$$

Here Pen is the penalty constant. N is the number of points in the GA solutions. To compute the fitness function for an individual, we should have the coordinates of each point in the individual (i.e., column and row for each PI from a lookup table). Thus, we can compute the distance between any two points in the search space (i.e., environment of the robot).

4.2 Path smoothing by path interpolation using rbfnn

Smooth paths are important in robotics because smooth paths are more suitable for non-holonomic mobile robots because these kinds of path are more preferable for designing continuous control algorithms to follow the paths [39]. Collision free path generated by GA is not smooth and it is represented as an array of points in two-dimensional space. The results of first step are further improved by running an additional algorithm that smoothes the path. However, that the resulting paths are indeed collision free. Here RBFNN is used solve the purpose. Instead, path-smoothing is used to turn a zigzag path into a curve or even a straight line. RBFNN are three-layer neural networks [16]. Their structure is shown on Fig. 5. These networks are widely used for nonlinear function approximation, as well as Multilayer Perceptrons (MLPs). Let $x = [x_1, x_2, \ldots, x_n]^T$ and $y = [y_1, y_2 \ldots\ldots, y_m]^T$ denote input and output vectors respectively. Activation function of hidden neuron is

$$\Phi_i(x) = e^{\frac{|x-w_i|^2}{2\sigma^2}}, \quad i = 1, 2, \ldots\ldots n \tag{5}$$

Where . denotes Euclidian norm. Activation function $\Phi_i(x)$ is centered in w_i, while σ denotes spread of the function. Output layer is linear, so output of the network is linear combination of the hidden layer outputs:

$$y_i = \sum_{i=1}^{k} l_{ij}\Phi_i(x), \qquad j = 1, 2 \ldots.., m \tag{6}$$

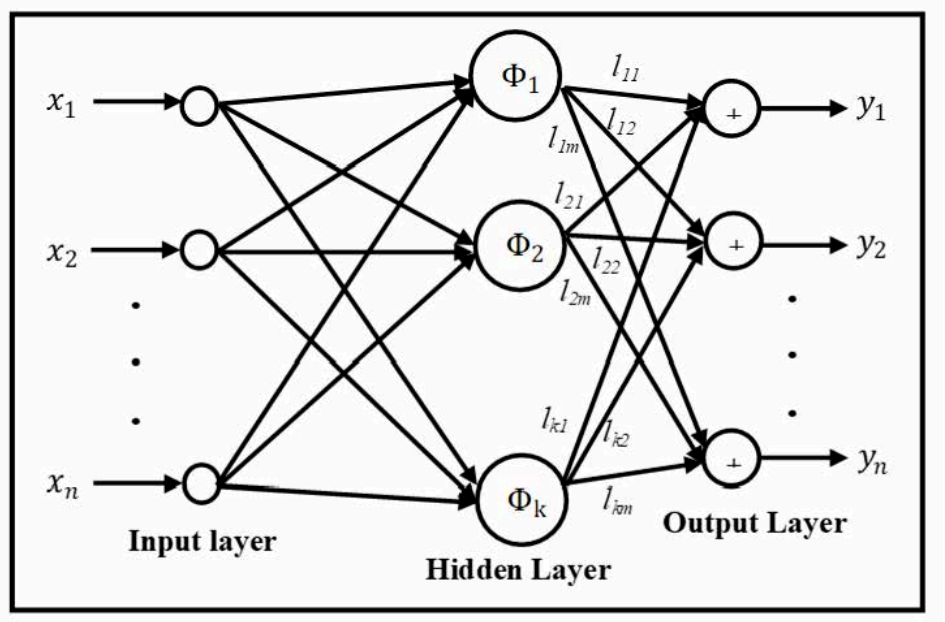

Fig. 5: Architecture of RBFN

It can be seen from Eq. (5) that activation of hidden layer neuron i is the strongest when $x = w_i$, because $\Phi_i(x) = 1$. Activation decreases when input departs from vector. Basic idea is to divide input space onto k overlapping regions, while every hidden neuron will be active only in one region, i.e. some neighborhood of . If region width is too small, network generalizes poorly, while for large values of this parameter, interpolation can be coarse.

Path smoothing is achieved by interpolation using RBFNN. RBFNN gives very good results, accurate approximation and fast training. Neural networks are used for function approximation. This means that in general case they cannot be used here directly, because obtained path may not be a function. So, the idea is to parameterize x and y coordinate of the path, i.e., to associate time instant to every point of the path.

Path interpolation is achieved using RBFNN with one input called time and two outputs (x and y coordinate of path point). The first step is the association of the time instant to every point of the path which is called generation of training set. The simplest way to generate time vector is to adopt uniform velocity of motion along the path. Now, time vector can be evaluated recursively using Eq. (7), where and denote two successive points of the path.

$$t_i = t_{i-1} + \frac{\|z_i - z_{i-1}\|}{v}, \qquad z_i = (x_i, y_i)\,, z_{i-1} = (x_{i-1}, y_{i-1}) \tag{7}$$

Region centers w_i and output layer weights l_{ij} in Eq. (5) and Eq. (6) are determined in training process; region width and hidden layer size are adjusted experimentally. Choice of these two parameters is significant. After training it could happen that path generated by RBFNN is not collision free. So the accuracy of the approximation is not the only criterion that influences choice of the hidden layer size and σ, collision free property is a more important one.

5 Simulation results

The entire algorithm was implemented in MATALB platform. The map was fed into the simulator as a - BMP image with the dimensions of the image as the dimensions of map. In all the experiments we used a map of size 500 $\times$ 500 pixels. The white regions denoted the presence of accessible areas and the black denoted obstacles. In this context, we have created N unknown environments containing static obstacles; (until now we have tested 36 environments), we start with no obstacle until the complexity order is done. As there is no information at advance, this creation can give other configurations of environments that mean that, the user can change the positions of all objects and can change the shapes of obstacle (big, small, different sizes) in the scenes. This has no effect since the environment is unknown. We can give different infinite environment complexity, in order to achieve the desired task of robot success in satisfactory manner to avoid suitably the static obstacles. Here we present the results to four maps out of the various maps used for the testing purposes. All four of these maps are generated with some or the other inspiration to conditions that a robot might face in real life. Many obstacles were places in between the source and the goal. The coordinate (10, 10) was the source specified for all cases. The goal was the bottom right point with the coordinates of (490, 490). For simplicity, the robot is taken as a point-sized object. The initial configuration including, lengths, widths, sources, goals and speeds of the robots is given in Table 1. The GA used had a population count of 50 individuals. The algorithm was executed for a total of 10 generations. The crossover rate fixed for the algorithm was 0.7. The value of the number of points to be optimized by the GA was fixed to 30. All the simulations were carried on a system with 3 GB RAM and 2.40 GHz Core 2 Duo Intel processor. The first map denotes a simple map with a two obstacles in the path from source to destination. This may be a very common situation where a robot has a straight path with some obstacle on its way. The second and the third maps have a variety of obstacles and the robot needs to use some intelligent technique to figure out its way by avoiding all the obstacles. This tests the ability of the robot to calculate the shortest path. The fourth map that we present is the path based map where robot tries to move itself on a non-straight and complex path from the source to the destination. The maps and the paths traced by the robot for each of these cases is given in Fig. 6.We also study the behaviour of the best fitness value in all the discussed paths and configurations. The plots of the best fitness against the generations are given in Fig. 7-10.

It may be easily seen that in all the cases the robot was able to evolve a path from the source to the destination. The path served the set criterion in the fitness function. The graph shown in Fig. 6 (SL. No. 1) is an example of a simple problem where the solution was generated with a two intermediate step or a complexity of 2. It may be easily observed from the path traced that the robot kept a comfortable distance of separation from the obstacle and avoided going too close to it. A similar trend can be seen in the Fig. 6 (SL. No. 2-3); which

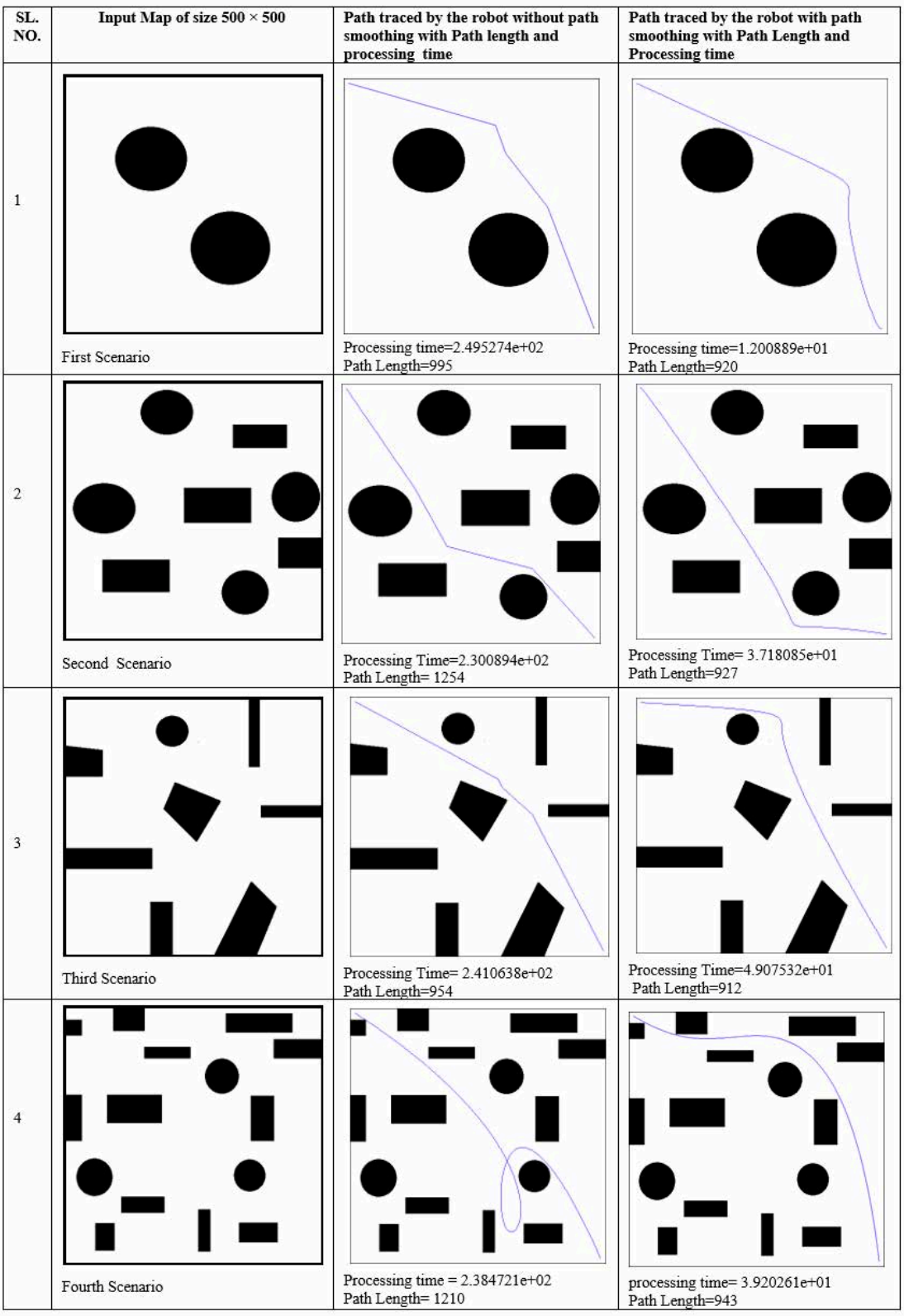

Fig. 6: The Maps and the Path Traced by the Robot for Different Scenarios without Smoothing and with Smoothing

Table 1: The statistics of robot for various scenarios

Property	Robot
Length (pixels)	1
Width (pixels)	1
Speed (pixels/unit time)	1
Source	(10, 10)
Goals	(490, 490)
Travel Time	Given in Fig. 6 for different scenarios
Length of Path	Given in Fig. 6 for different scenarios
Reached	Yes
Collisions	No

represents a slightly more complex problem, even though the solution is just a level more in complexity. Here it is interesting to observe that the robot could have moved diagonally in the top section which might have resulted in shorter path but it preferred not to do but traverse straight. This is due to the heavy penalty of increase in complexity that we added. In the physical movement by a robotic controller the path would get smoothen up. This would enable the robot to traverse the path at smoothly at high speed. This would compensate the increase in length where the robot would have been forced to make two sharp turns.

The graph shown in Fig. 6 (SL. No. 4), the robot has taken a very steep turn in the top section. It can be seen that the conditions were highly chaotic and finding a way out was not very easy. Here a very special path is chosen by the robot as the most optimal path. After this section the work is relatively simple where the robot has a lot of straight path to traverse. This large amount of straight path compensates for the complexity added at start. Various other paths could have been possible of lower initial complexity but the straight path traversed gives this path the edge over other options.

We also study the behaviour of the best fitness value in all the discussed paths and configurations. The plots of the best fitness against the generations are given in Fig. 7-10 for without smoothing and smoothed by RBFNN. Through some results, we give a comparison between this strategy and methods based on road maps, potential fields, Rapidly-exploring random trees (RRT), GA and A* search. In Table 2 and Table 3, a comparative result against execution time and path length of getting the final path for different Scenarios is tabulated which shows that proposed method is practical and reliable as shown in simulated plots in Fig. 11 and Fig. 12.

Table 2: Comparison of Execution Time of Getting the Final Path

Environments	Our Approach	GA	PRM	RRT	Bidirectional RRT	APF	A*
First scenario	1.200889e+01	2.495274e+02	8.108007e+01	5.896705e+01	5.302752e+00	7.489889e+01	9.613828e+02
Second scenario	3.718085e+01	2.300894e+02	9.258600e+01	9.336826e+01	4.267160e+01	3.622861e+01	1.271603e+03
Third scenario	4.907532e+01	2.410638e+02	5.233498e+01	4.853172e+01	4.816639e+01	1.810086e+02	1.727154e+03
Fourth scenario	3.920261e+01	2.384721e+02	3.944015e+01	4.856475e+01	4.230397e+01	3.929066e+01	7.973236e+02

Table 3: Comparison of the Length of the Final Path

Environments	Our Approach	GA	PRM	RRT	Bidirectional RRT	APF	A*
First scenario	920	995	9.225870e+02	9.938193e+02	9.232405e+02	9.664379e+02	9.432590e+02
Second scenario	927	1254	9.981288e+02	9.951112e+02	9.924840e+02	1.268308e+03	9.320171e+02
Third scenario	912	954	9.956767e+02	9.595475e+02	9.660145e+02	1.332604e+03	9.142066e+02
Fourth scenario	943	1210	9.930154e+02	9.834213e+02	9.808483e+02	9.515194e+02	9.432303e+02

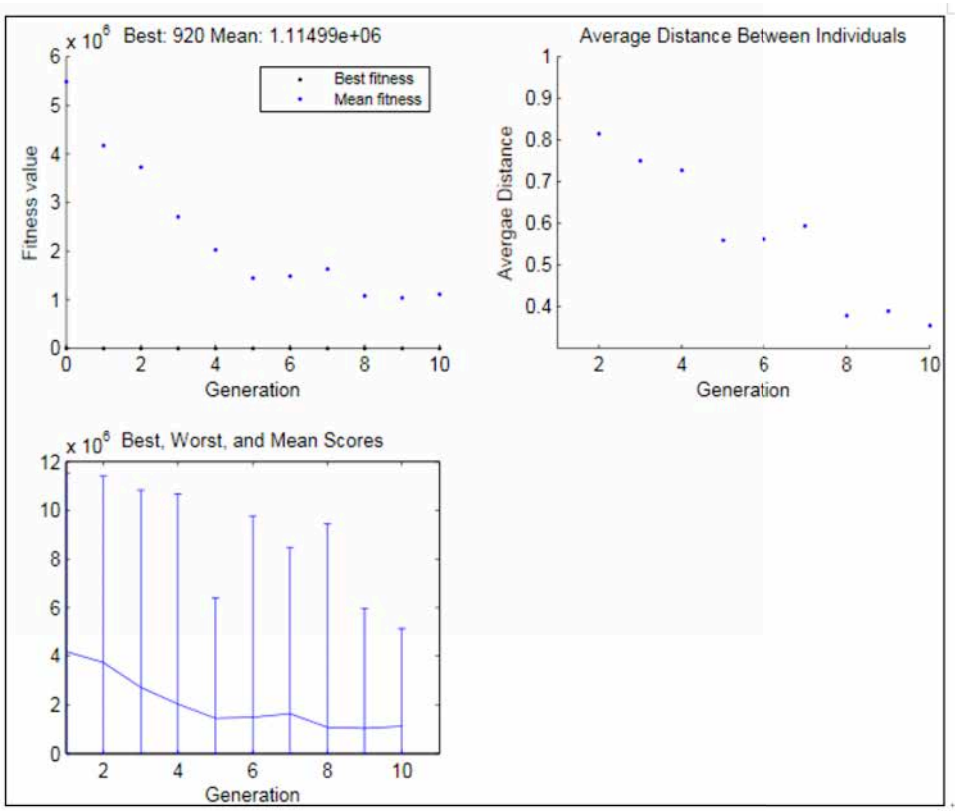

Fig. 7: Convergence process for GA with a 500 × 500 grid with obstacle environment smoothing by RBFNN for the first scenario

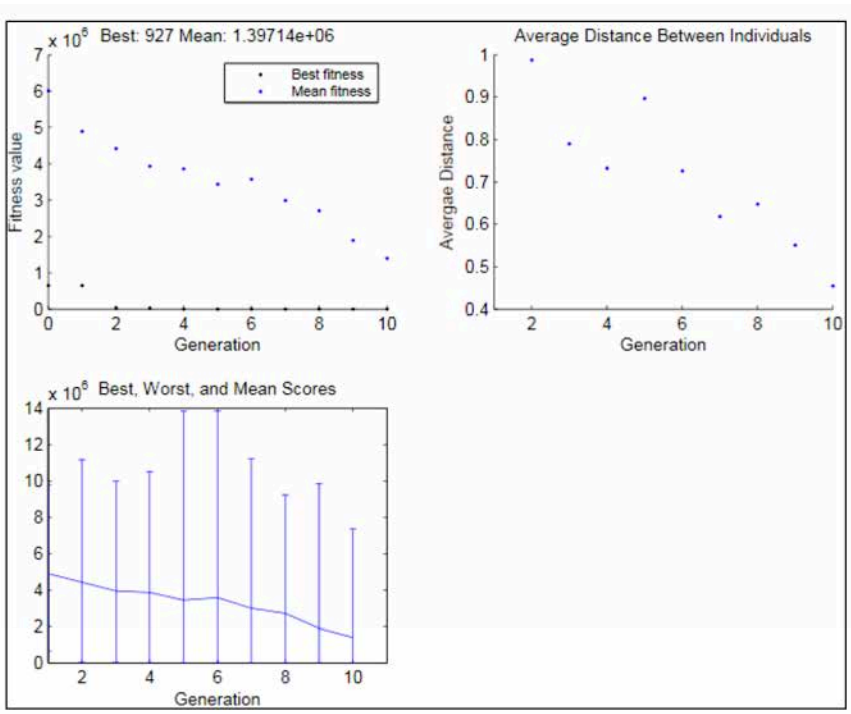

Fig. 8: Convergence process for GA with a 500 × 500 grid with obstacle environment smoothing by RBFNN second scenario

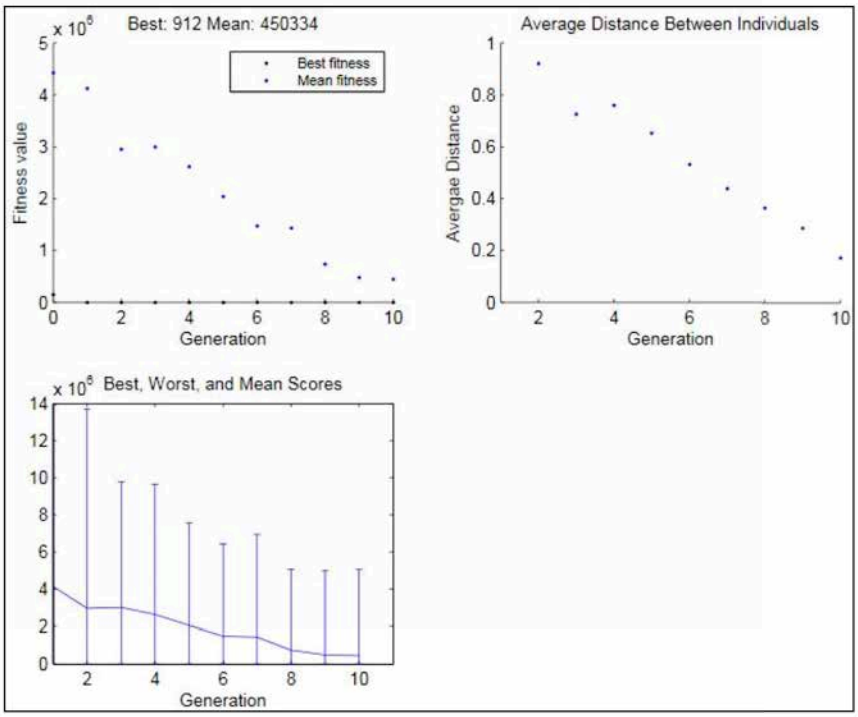

Fig. 9: Convergence Process for GA with a 500 × 500 grid with obstacle environment smoothing by RBFNN for the Third Scenario

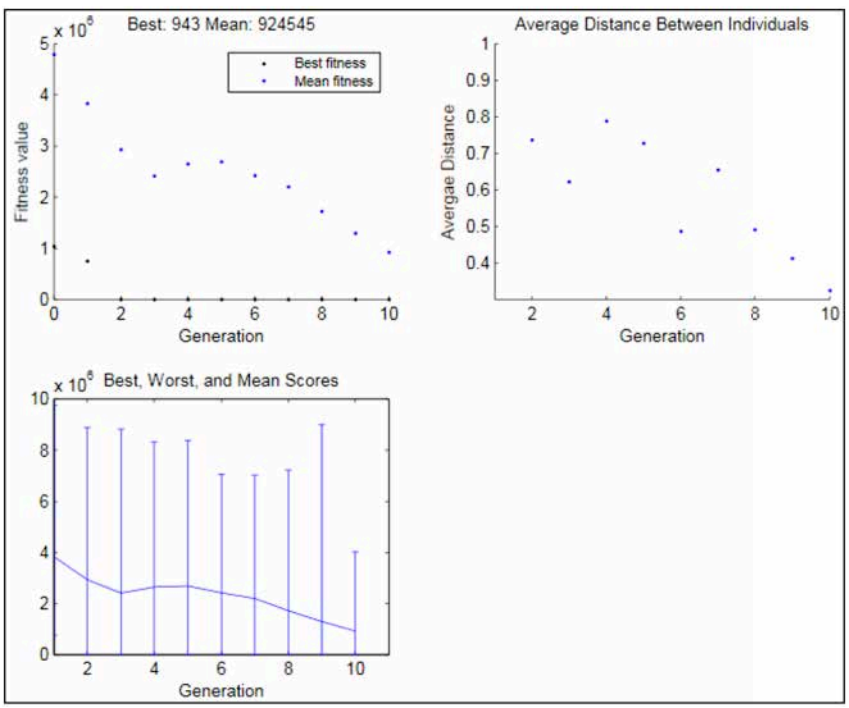

Fig. 10: Convergence Process for GA with a 500 × 500 grid with obstacle environment smoothing by RBFNN for the Fourth Scenario

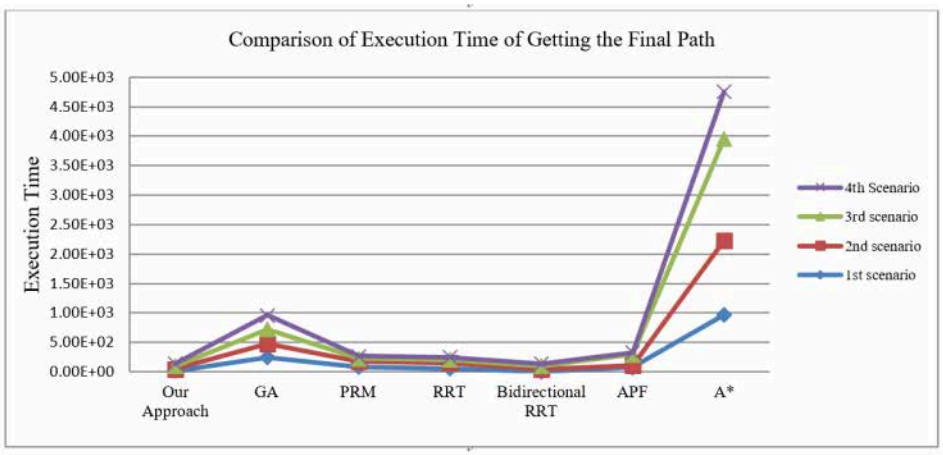

Fig. 11: Simulated Plot of Comparison of Execution Time of Getting the Final Path for different Scenarios

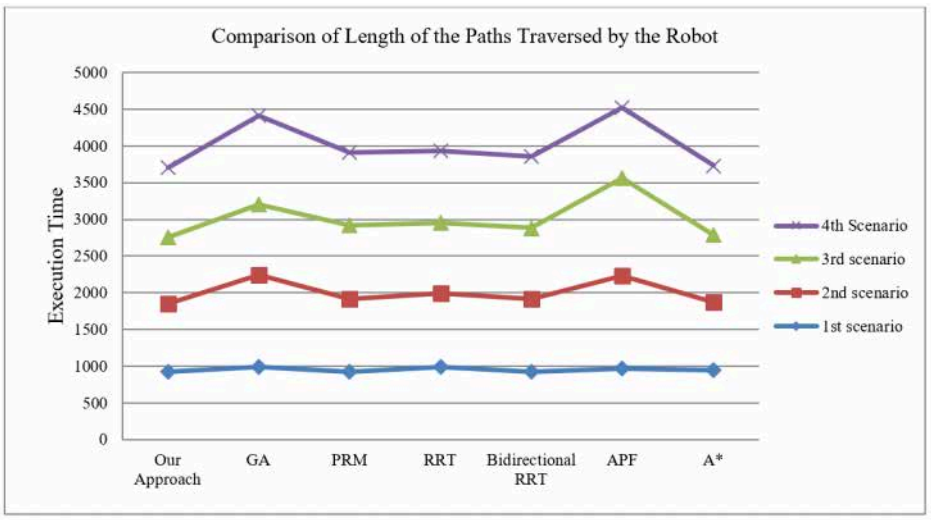

Fig. 12: Simulated Plot of Comparison of the Length Final Path for different Scenarios

6 Conclusion

In this paper, we propose an algorithmic framework for mobile robot path planning problems solution optimally or near optimally using GA integrated with RBFNN. At first GA algorithm tries to identify good points in the map, which are later judges by the RBFNN algorithm by carrying out planning with the given robot and constraints. The RBFNN generates smoother paths catering to the non-holonomic constraints. We also propose a mobile robot controlling algorithm which allows eight-neighbour movements, so that path-planning can handle complicated search spaces with low complexities. The algorithm was experimented on with a variety of scenarios ranging from simple to difficult. Two objectives were simultaneously optimized which included path length and path smoothness. The purpose was to test whether the algorithm is able to generate a feasible trajectory even for scenarios where it may be difficult to maneuver the mobile robot. This ensures that the algorithm is scalable to complex situations whilst at the same time returning solutions within a small execution time. Simulation results clearly indicate that our GA-RBFNN approach shows more stability and efficiency in this two-dimensional mobile path planning optimization scheme than other established algorithms in the literature.

Apart from the promising results presented in this paper, there are different aspects for future research. As future works, we would like to extend this approach to multiple cooperating robots and mobile manipulators. Our future work will cover some further comparisons with more state-of-the-art intelligent algorithms. In future work, consideration of the H/W implementation of the proposed GA-RBFNN algorithm-based path planning on real mobile platform would be made. Another possible future direction is to implement the proposed algorithm in a dynamic environment with a moving goal.

References

[1] M. S. Alam, M. U. Rafique. Mobile robot path planning in environments cluttered with non-convex obstacles using particle swarm optimization. **in:** *2015 International Conference on Control, Automation and Robotics*, IEEE, 2015, 32–36.

[2] A. Bakdi, A. Hentout, H. Boutami, A. Maoudj, O. Hachour, B. Bouzouia. Optimal path planning and execution for mobile robots using genetic algorithm and adaptive fuzzy-logic control. *Robotics and Autonomous Systems*, 2017, **89**: 95–109.

[3] J. Barraquand, B. Langlois, J.-C. Latombe. Numerical potential field techniques for robot path planning. *IEEE transactions on systems, man, and cybernetics*, 1992, **22**(2): 224–241.

[4] P. Bhattacharya, M. L. Gavrilova. Roadmap-based path planning-using the voronoi diagram for a clearance-based shortest path. *IEEE Robotics & Automation Magazine*, 2008, **15**(2): 58–66.

[5] H. Bing, L. Gang, G. Jiang, W. Hong, N. Nan, L. Yan. A route planning method based on improved artificial potential field algorithm. **in:** *2011 IEEE 3rd International Conference on Communication Software and Networks*, IEEE, 2011, 550–554.

[6] R. Bohlin, L. E. Kavraki. Path planning using lazy prm. **in:** *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1, IEEE, 2000, 521–528.

[7] R. Bohlin, L. E. Kavraki. Path planning using lazy prm. **in:** *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 1, IEEE, 2000, 521–528.

[8] N. Bourbakis, D. Goldman, R. Fematt, I. Vlachavas, L. Tsoukalas. Path planning in a 2-d known space using neural networks and skeletonization. **in:** *1997 IEEE International Conference on Systems, Man, and Cybernetics. Computational Cybernetics and Simulation*, vol. 3, IEEE, 1997, 2001–2005.

[9] D. Cagigas, J. Abascal. A hierarchical extension of the d* algorithm. *Journal of Intelligent and Robotic Systems*, 2005, **42**(4): 393–413.

[10] C. Cai, S. Ferrari. Information-driven sensor path planning by approximate cell decomposition. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 2009, **39**(3): 672–689.

[11] C. Cai, S. Ferrari. Information-driven sensor path planning by approximate cell decomposition. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 2009, **39**(3): 672–689.

[12] W. F. Carriker, P. K. Khosla, B. H. Krogh. The use of simulated annealing to solve the mobile manipulator path planning problem. **in:** *Proceedings., IEEE International Conference on Robotics and Automation*, IEEE, 1990, 204–209.

[13] R. R. Cazangi, F. J. Von Zuben, M. F. Figueiredo. Evolutionary stigmergy in multipurpose navigation systems. **in:** *2006 IEEE International Conference on Evolutionary Computation*, IEEE, 2006, 370–377.

[14] I. Chaari, A. Koubâa, S. Trigui, H. Bennaceur, A. Ammar, K. Al-Shalfan. Smartpath: an efficient hybrid aco-ga algorithm for solving the global path planning problem of mobile robots. *International Journal of Advanced Robotic Systems*, 2014, **11**(7): 94.

[15] C. M. Clark. Probabilistic road map sampling strategies for multi-robot motion planning. *Robotics and Autonomous Systems*, 2005, **53**(3-4): 244–264.

[16] C. I. Connolly, J. B. Burns, R. Weiss. Path planning using laplace's equation. **in:** *Proceedings., IEEE International Conference on Robotics and Automation*, IEEE, 1990, 2102–2106.

[17] M. A. Contreras-Cruz, V. Ayala-Ramirez, U. H. Hernandez-Belmonte. Mobile robot path planning using artificial bee colony and evolutionary programming. *Applied Soft Computing*, 2015, **30**: 319–328.

[18] J. Cortés, L. Jaillet, T. Siméon. Disassembly path planning for complex articulated objects. *IEEE Transactions on Robotics*, 2008, **24**(2): 475–481.

[19] R. V. Cowlagi, P. Tsiotras. Multiresolution path planning with wavelets: A local replanning approach. **in:** *2008 American Control Conference*, IEEE, 2008, 1220–1225.

[20] N. L. Doh, N. Cho, K. Lee, J. Lee, W. K. Chung, S.-R. Oh. A systematic representation of edges in topological maps for mobile robots using wavelet transformation. **in:** *Proceedings of the 2005 IEEE international conference on robotics and automation*, IEEE, 2005, 2822–2827.

[21] M. Dorigo, L. M. Gambardella. Ant colony system: a cooperative learning approach to the traveling salesman problem. *IEEE Transactions on evolutionary computation*, 1997, **1**(1): 53–66.

[22] D. Drake, S. Koziol, E. Chabot. Mobile robot path planning with a moving goal. *IEEE Access*, 2018, **6**: 12800–12814.

[23] M. Elbanhawi, M. Simic. Sampling-based robot motion planning: A review. *Ieee access*, 2014, **2**: 56–77.

[24] R. Fierro, F. L. Lewis. Control of a nonholonomic mobile robot using neural networks. *IEEE Transactions on neural networks*, 1998, **9**(4): 589–600.

[25] M. P. Garcia, O. Montiel, O. Castillo, R. Sepúlveda, P. Melin. Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation. *Applied Soft Computing*, 2009, **9**(3): 1102–1110.
[26] M. P. Garcia, O. Montiel, O. Castillo, R. Sepúlveda, P. Melin. Path planning for autonomous mobile robot navigation with ant colony optimization and fuzzy cost function evaluation. *Applied Soft Computing*, 2009, **9**(3): 1102–1110.
[27] S. Garrido, L. Moreno, M. Abderrahim, F. Martin. Path planning for mobile robot navigation using voronoi diagram and fast marching. **in:** *2006 IEEE/RSJ International Conference on Intelligent Robots and Systems*, IEEE, 2006, 2376–2381.
[28] N. Gasilov, M. Doğan, V. Arici. Two-stage shortest path algorithm for solving optimal obstacle avoidance problem. *IETE Journal of Research*, 2011, **57**(3): 278–285.
[29] F. Glover, M. Laguna. Tabu search. **in:** *Handbook of combinatorial optimization*, Springer, 1998, 2093–2229.
[30] D. E. Goldberg, J. H. Holland. Genetic algorithms and machine learning. *Machine learning*, 1988, **3**(2): 95–99.
[31] R. W. Hamming, E. A. Feigenbaum. *Problem solving methods in artificial intelligence*. 2017.
[32] J. Han, Y. Seo. Mobile robot path planning with surrounding point set and path improvement. *Applied Soft Computing*, 2017, **57**: 35–47.
[33] Z. Hong, Y. Liu, G. Zhongguo, C. Yi. The dynamic path planning research for mobile robot based on artificial potential field. **in:** *2011 International Conference on Consumer Electronics, Communications and Networks (CECNet)*, IEEE, 2011, 2736–2739.
[34] M. A. Hossain, I. Ferdous. Autonomous robot path planning in dynamic environment using a new optimization technique inspired by bacterial foraging technique. *Robotics and Autonomous Systems*, 2015, **64**: 137–141.
[35] Y. K. Hwang, N. Ahuja. Gross motion planningła survey. *ACM Computing Surveys (CSUR)*, 1992, **24**(3): 219–291.
[36] F. Imeson, S. L. Smith. A language for robot path planning in discrete environments: The tsp with boolean satisfiability constraints. **in:** *2014 IEEE International Conference on Robotics and Automation (ICRA)*, IEEE, 2014, 5772–5777.
[37] G. W. Irwin, K. Warwick, K. J. Hunt. *Neural network applications in control*, 53. Iet, 1995.
[38] G. E. Jan, K. Y. Chang, I. Parberry. Optimal path planning for mobile robot navigation. *IEEE/ASME transactions on mechatronics*, 2008, **13**(4): 451–460.
[39] R. Kala, A. Shukla, R. Tiwari. Robot path planning using dynamic programming with accelerating nodes. *Paladyn*, 2012, **3**(1): 23–34.
[40] M. Kapanoglu, M. Alikalfa, M. Ozkan, O. Parlaktuna, et al. A pattern-based genetic algorithm for multi-robot coverage path planning minimizing completion time. *Journal of intelligent manufacturing*, 2012, **23**(4): 1035–1045.
[41] L. E. Kavraki, M. N. Kolountzakis, J.-C. Latombe. Analysis of probabilistic roadmaps for path planning. **in:** *Proceedings of IEEE International Conference on Robotics and Automation*, vol. 4, IEEE, 1996, 3020–3025.
[42] O. Khatib. Real-time obstacle avoidance for manipulators and mobile robots. **in:** *Autonomous robot vehicles*, Springer, 1986, 396–404.
[43] A. Konar, I. G. Chakraborty, S. J. Singh, L. C. Jain, A. K. Nagar. A deterministic improved q-learning for path planning of a mobile robot. *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 2013, **43**(5): 1141–1153.
[44] C. Kozakiewicz, M. Ejiri. Neural network approach to path planning for two dimensional robot motion. **in:** *Proceedings IROS'91: IEEE/RSJ International Workshop on Intelligent Robots and Systems' 91*, IEEE, 1991, 818–823.
[45] J. Kuffner, S. L. RRT-Connect. An efficient approach to single-query path planning ieee international conference on robotics and automation. *San Francisco*, 2000, 473–479.
[46] C. Lamini, S. Benhlima, A. Elbekri. Genetic algorithm based approach for autonomous mobile robot path planning. *Procedia Computer Science*, 2018, **127**: 180–189.
[47] J.-C. Latombe. *Robot motion planning*, vol. 124. Springer Science & Business Media, 2012.
[48] S. M. LaValle. Rapidly-exploring random trees: A new tool for path planning. 1998.
[49] S. M. LaValle, J. J. Kuffner Jr. Randomized kinodynamic planning. *The international journal of robotics research*, 2001, **20**(5): 378–400.
[50] J. Lee, D.-W. Kim. An effective initialization method for genetic algorithm-based robot path planning using a directed acyclic graph. *Information Sciences*, 2016, **332**: 1–18.
[51] G. Li, W. Chou. Path planning for mobile robot using self-adaptive learning particle swarm optimization. *Science China Information Sciences*, 2018, **61**(5): 052204.
[52] J.-H. Liang, C.-H. Lee. Efficient collision-free path-planning of multiple mobile robots system using efficient artificial bee colony algorithm. *Advances in Engineering Software*, 2015, **79**: 47–56.
[53] T. Lozano-Pérez, M. A. Wesley. An algorithm for planning collision-free paths among polyhedral obstacles. *Communications of the ACM*, 1979, **22**(10): 560–570.
[54] J. J. M. Lunenburg, S. A. M. Coenen, G. J. Naus, M. J. G. van de Molengraft, M. Steinbuch. Motion planning for mobile robots: A method for the selection of a combination of motion-planning algorithms. *IEEE Robotics & Automation Magazine*, 2016, **23**(4): 107–117.

[55] T. T. Mac, C. Copot, D. T. Tran, R. De Keyser. Heuristic approaches in robot path planning: A survey. *Robotics and Autonomous Systems*, 2016, **86**: 13–28.
[56] E. Masehian, D. Sedighizadeh. Classic and heuristic approaches in robot motion planning-a chronological review. *World Academy of Science, Engineering and Technology*, 2007, **23**(5): 101–106.
[57] J. Minguez, F. Lamiraux, J.-P. Laumond. Motion planning and obstacle avoidance. **in:** *Springer handbook of robotics*, Springer, 2016, 1177–1202.
[58] P. K. Mohanty, D. R. Parhi. Optimal path planning for a mobile robot using cuckoo search algorithm. *Journal of Experimental & Theoretical Artificial Intelligence*, 2016, **28**(1-2): 35–52.
[59] A. Nasrinahar, J. H. Chuah. Intelligent motion planning of a mobile robot with dynamic obstacle avoidance. *Journal on Vehicle Routing Algorithms*, 2018, 1–16.
[60] J. S. Oh, Y. H. Choi, J. B. Park, Y. F. Zheng. Complete coverage navigation of cleaning robots using triangular-cell-based map. *IEEE Transactions on Industrial Electronics*, 2004, **51**(3): 718–726.
[61] J. S. Oh, Y. H. Choi, J. B. Park, Y. F. Zheng. Complete coverage navigation of cleaning robots using triangular-cell-based map. *IEEE Transactions on Industrial Electronics*, 2004, **51**(3): 718–726.
[62] T. Oral, F. Polat. Mod* lite: an incremental path planning algorithm taking care of multiple objectives. *IEEE Transactions on Cybernetics*, 2015, **46**(1): 245–257.
[63] A. Pandey, D. R. Parhi. Optimum path planning of mobile robot in unknown static and dynamic environments using fuzzy-wind driven optimization algorithm. *Defence Technology*, 2017, **13**(1): 47–58.
[64] B. Raveh, A. Enosh, D. Halperin. A little more, a lot better: Improving path quality by a path-merging algorithm. *IEEE Transactions on Robotics*, 2011, **27**(2): 365–371.
[65] B. Raveh, A. Enosh, D. Halperin. A little more, a lot better: Improving path quality by a path-merging algorithm. *IEEE Transactions on Robotics*, 2011, **27**(2): 365–371.
[66] M. R. K. Ryan. Exploiting subgraph structure in multi-robot path planning. *Journal of Artificial Intelligence Research*, 2008, **31**: 497–542.
[67] M. Saska, M. Macas, L. Preucil, L. Lhotska. Robot path planning using particle swarm optimization of ferguson splines. **in:** *2006 IEEE Conference on Emerging Technologies and Factory Automation*, IEEE, 2006, 833–839.
[68] A. Sgorbissa, R. Zaccaria. Planning and obstacle avoidance in mobile robotics. *Robotics and Autonomous Systems*, 2012, **60**(4): 628–638.
[69] J. Sheng, G. He, W. Guo, J. Li. An improved artificial potential field algorithm for virtual human path planning. **in:** *International Conference on Technologies for E-Learning and Digital Entertainment*, Springer, 2010, 592–601.
[70] W.-R. Shi, X.-H. Huang, W. Zhou. Path planning of mobile robot based on improved artificial potential field. *Jisuanji Yingyong/ Journal of Computer Applications*, 2010, **30**(8): 2021–2023.
[71] A. Sud, E. Andersen, S. Curtis, M. C. Lin, D. Manocha. Real-time path planning in dynamic virtual environments using multiagent navigation graphs. *IEEE transactions on visualization and computer graphics*, 2008, **14**(3): 526–538.
[72] A. Sud, E. Andersen, S. Curtis, M. C. Lin, D. Manocha. Real-time path planning in dynamic virtual environments using multiagent navigation graphs. *IEEE transactions on visualization and computer graphics*, 2008, **14**(3): 526–538.
[73] O. Takahashi, R. J. Schilling. Motion planning in a plane using generalized voronoi diagrams. *IEEE Transactions on robotics and automation*, 1989, **5**(2): 143–150.
[74] O. Takahashi, R. J. Schilling. Motion planning in a plane using generalized voronoi diagrams. *IEEE Transactions on robotics and automation*, 1989, **5**(2): 143–150.
[75] R. E. Tarjan. A unified approach to path problems. *Tech. Rep.*, STANFORD UNIV CALIF DEPT OF COMPUTER SCIENCE, 1979.
[76] C. E. Thomaz, M. A. C. Pacheco, M. M. B. Vellasco. Mobile robot path planning using genetic algorithms. **in:** *International Work-Conference on Artificial Neural Networks*, Springer, 1999, 671–679.
[77] K. Uyanik. Artificial potential fields. *Motion Planning for Mobile Robot*, 2010, **5**(1): 1–5.
[78] G. Vachtsevanos, H. Hexmoor. A fuzzy logic approach to robotic path planning with obstacle avoidance. **in:** *1986 25th IEEE Conference on Decision and Control*, IEEE, 1986, 1262–1264.
[79] Y. Wang, G. S. Chirikjian. A new potential field method for robot path planning. **in:** *Proceedings 2000 ICRA. Millennium Conference. IEEE International Conference on Robotics and Automation. Symposia Proceedings (Cat. No. 00CH37065)*, vol. 2, IEEE, 2000, 977–982.
[80] T. Weerakoon, K. Ishii, A. A. F. Nassiraei. An artificial potential field based mobile robot navigation method to prevent from deadlock. *Journal of Artificial Intelligence and Soft Computing Research*, 2015, **5**(3): 189–203.
[81] X. Yang, M. Moallem, R. V. Patel. A layered goal-oriented fuzzy motion planning strategy for mobile robot navigation. *IEEE Transactions on Systems, Man, and Cybernetics, Part B (Cybernetics)*, 2005, **35**(6): 1214–1224.
[82] Z.-z. Yu, J.-h. Yan, J. Zhao, Z.-F. Chen, Y.-h. Zhu. Mobile robot path planning based on improved artificial potential field method. *Harbin Gongye Daxue Xuebao(Journal of Harbin Institute of Technology)*, 2011, **43**(1): 50–55.

[83] Z.-z. Yu, J.-h. Yan, J. Zhao, Z.-F. Chen, Y.-h. Zhu. Mobile robot path planning based on improved artificial potential field method. *Harbin Gongye Daxue Xuebao(Journal of Harbin Institute of Technology)*, 2011, **43**(1): 50–55.

[84] B. Zhang, W. Chen, M. Fei. An optimized method for path planning based on artificial potential field. **in:** *Sixth International Conference on Intelligent Systems Design and Applications*, vol. 3, IEEE, 2006, 35–39.

[85] L. Zhang, D. Manocha. An efficient retraction-based rrt planner. **in:** *2008 IEEE International Conference on Robotics and Automation*, IEEE, 2008, 3743–3750.

[86] Q. Zhu, J. Hu, W. Cai, L. Henschen. A new robot navigation algorithm for dynamic unknown environments based on dynamic path re-computation and an improved scout ant algorithm. *Applied Soft Computing*, 2011, **11**(8): 4667–4676.

[87] L. Zuo, Q. Guo, X. Xu, H. Fu. A hierarchical path planning approach based on a? and least-squares policy iteration for mobile robots. *Neurocomputing*, 2015, **170**: 257–266.