Hybrid Genetic-Fuzzy Approach to Autonomous Mobile Robot

K.S. Senthilkumar[1] K.K. Bharadwaj[2]
1 College of Arts and Science, King Saud University
Wadi Al Dawasir, Riyadh, KSA
2 School of Computer & Systems Sciences
Jawaharlal Nehru University
New Delhi, India
ksskumar16@gmail.com

ABSTRACT - *An Autonomous Mobile Robot (AMR) is a machine able to extract information from its environment and use knowledge about its world to move safely in a meaningful and purposeful manner. Robot Navigation and Obstacle Avoidance are from the most important problems in mobile robots, especially in unknown environments. It must be able to interact with other objects safely. Several techniques such as Fuzzy logic, Reinforcement learning, Neural Networks and Genetic Algorithms, have applied to AMR in order to improve their performance. During the past several years Hybrid Genetic-fuzzy method has emerged as one of the most active and fruitful areas for research in the application of intelligent system design. The objective of this work is to provide a Hybrid method by which an improved set of rules governing the actions and behavior of a simple navigating and obstacle avoiding AMR. Genes are in the form of distances and angles labels. The chromosomes are represented as a rule written in a Boolean algebraic form. The method used to enhance the performance employs a simulation model designed by using Visual Basic software.*

Keywords – Autonomous Mobile Robot, Fuzzy Logic, Genetic algorithms, Hybrid, Path Planning, Soft Computing.

1 Introduction

Autonomous Mobile Robots (AMR) operate in a given environment without being remotely controlled by a human operator [6]. A mobile Robot has to move from its initial position to a target in a minimum traveling time. The basic needs for all autonomous mobile robots are navigation and obstacle avoidance ability that helps mobile robots to move without collision in the environment. Real-time obstacle avoidance is one of the key issues to successful applications of mobile robot systems. All mobile robots have some kind of collision avoidance, ranging from detecting an obstacle and stop the robot short of it, through advanced algorithms, that enable the robot to bypass obstacles [1].

In late eighties much of the work on reactive navigation has been inspired by the layered control system of the subsumption architecture. Then some fuzzy-logic based behaviour control schemes have been proposed by Saffiotti [7]. Soft computing [3] approaches are preferable over conventional control system design, for problems that are difficult to describe by analytical models. The features of fuzzy control, neural networks and evolutionary algorithms are of particular benefit to the type of problems emerging in behavior based robotics [7]. Fuzzy behavior hierarchies, neuro-fuzzy and genetic fuzzy system are valuable methodologies for the design and adaptation of complex robotic behaviors. A new approach in robotic control is Evolutionary Robotics [5] that uses evolution as a tool to create increasingly better robot controllers. The most well known class of Evolutionary Algorithms (EAs) are Genetic Algorithms (GAs), pioneered by Holland and later popularized by Mitchell. Genetic algorithms [4] are search algorithms based on the principles of natural selection and natural genetics, and are particularly suited for optimization problems. The designers specify the behaviours that are desired, by encoding the fitness function for the population to evolve. It does not need to know the environment that it has to operate in a priory. Instead, we need to devise fitness evaluators to guide the evolution of programs towards the required task [2].

A Fuzzy Logic Controller (FLC) is one of the most successful applications of the fuzzy set theory [7]. The performance of an FLC depends largely on the selection of membership function distribution and rule base. It is a suitable tool for handling imprecision and uncertainty. Several researchers used a fuzzy logic approach for robot navigation but they did not use any optimization technique to find optimal solution to the problem. The advantage of using fuzzy logic is that it allows for easy combination of various behaviours' output through a command fusion process. The knowledge representation of fuzzy rule based systems combined with the learning capabilities of neural networks and evolutionary algorithm opens a promising path towards more intelligent and robust robotic systems.

This paper has a five sections including Introduction. Section 2 has the over all idea about the techniques applied in this research area in present and past. And also it includes the basic idea about the soft computing method such as fuzzy logic and Genetic algorithm. In section 3, we introduce the proposed algorithm for hybrid genetic-fuzzy approach to Automobile Robot Navigation. We have shown how genetic algorithms can be used to evolve fuzzy rule combinations for Robot Navigation and Obstacle Avoidance. Section 4 contains the way how this algorithm has been simulates and the results obtained. Finally, the Conclusions discussed in section 5.

2 Hybrid Genetic-Fuzzy Approach in Research

The integration of different methods, adequate for their specific domain of problems, results in more powerful hybrid systems with higher machine intelligence, than using a single method exclusively. In particular, a great number of publications explore the use of GAs for designing fuzzy systems [4]. These approaches have been given the general name Genetic-Fuzzy Systems (GFSs). A priori knowledge in the form of linguistic variables, fuzzy membership function parameters, fuzzy rules, number of rules, etc., may be incorporated easily into the genetic design process. The generic code structure and independent performance features of GAs make them suitable candidates for incorporating *a priori* knowledge. Over the last few years, these advantages have extended the use of GAs in the development of a wide range of approaches for designing fuzzy systems (FSs). As in the general case of FSs, the main application area of GFSs is system modeling and control. Regardless of the kind of optimization problem, i.e., given a system to be modeled or controlled, the involved design or tuning or learning process will be based on evolution. In some cases, the process starts off with an initial population obtained from available knowledge, while in other cases the initial population is randomly generated. Since the evolution operators work in a coded representation of the FSs, certain compatibility between the operators and the structure of the chromosomes is required.

The most extended GFS type is the genetic fuzzy rule based system, where a GA is employed to learn or tune different components of an FRBS. Following the guideline of soft computing GAs have been widely used for automatic design of FLCs. The basic idea is to learn the rules of a knowledge-based system by artificial evolution. GA has been used diversely for FLC design either off-line or on-line although in the latter case computation time is sometimes prohibitive. The design of a FLC by hand requires expert knowledge in order to formulate the fuzzy rules. This design process can be automated by the use of a GA, which adapts the rule base in order to achieve desired control behavior of the fuzzy system. Sakawa and Mori proposed an efficient genetic algorithm for job-shop scheduling problems with fuzzy processing time and fuzzy due date [8]. Yun presented a new genetic algorithm with a fuzzy logic controller (FLC) for dealing with preemptive job-shop scheduling problems and non-preemptive job-shop scheduling problems [9]. The FLC is used to regulate adaptively the crossover ratio and the mutation ratio of the proposed algorithm. A fuzzy logic approach for Mobile Robot Navigation did not use any optimization technique to find optimal collision-free path. Therefore, over the past years more research has been devoted to augment the approximate reasoning method of fuzzy systems with the learning capabilities of Neural Networks and Evolutionary algorithms.

3 Algorithm Over View

In this work, a Hybrid Fuzzy-Genetic Algorithm is proposed. Hybrid Fuzzy-Genetic algorithm has been designed by combining local search capability of fuzzy logic with the global search power of GA to find a collision- free path for a mobile robot. Here, the initial population for a GA has been created randomly for an FLC and standard GA parameters, namely crossover and mutation, are used to optimize. We applied our method to evolve a FLC for an Autonomous Mobile Robot. The given task is to reach a specified goal point while avoiding collisions with static (stationary) obstacles on the way. Beginning with a random population of rules that one would like to exhibit adaptive, intelligent behavior in their simulated environment, each individual is tested in that environment and scored according to their behavioral ability. Those that do better, in terms of succeeding at whatever goals the environment presents them with, will have more "offspring" in the next generation. The graphical representation of the proposed algorithm is in Fig. 1.

3.1 Mathematical Formulation of the Problem

The following assumptions are made to simplify the problem:
(i) Robot is considered to be a single point.
(ii) Each obstacle is represented by points occupied by them.
(iii) No kinematic constraints limit the motion of the robot.
(iv) No speed difference in movement
(v) The motions are only constrained by the fixed obstacles.

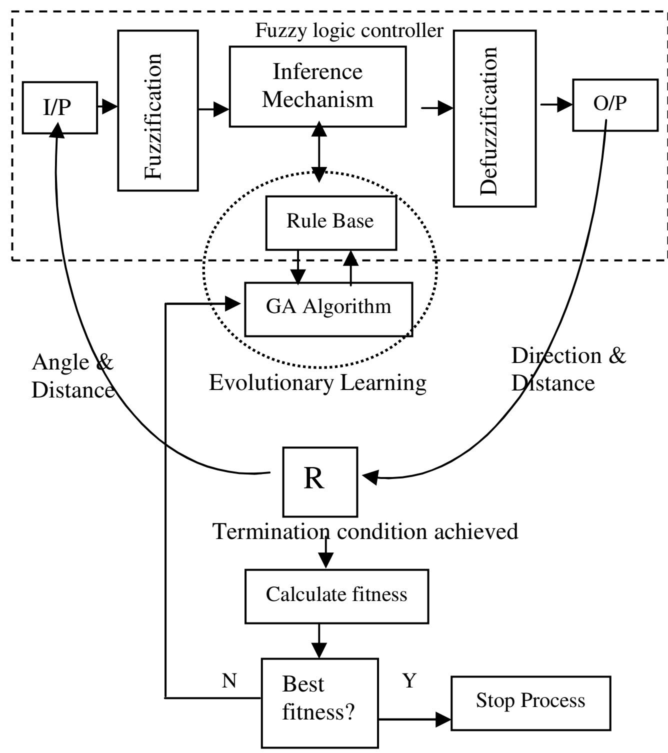

Figure 1 Hybrid Genetic-Fuzzy System

The path of the robot from an initial point to a final point is assumed to be a sequence of 1.. (N - 1) straight-line segments. On each line segment, the robot starts moving from the initial position and reaches the next possible position without collision. Our study takes place in the context of an environment totally unknown to the robot. An acquisition process is needed to obtain information about the robot environment. The scanning takes place in a circle, returning distance and direction types of data from detected obstacles. In the workspace, the robot moves progressing toward its target, scanning in a circle. Robot safety is obtained by avoiding stationary objects along the way. The Genetic-Fuzzy based collision avoidance algorithm admits two inputs for one set of rule and four inputs for another set of rules.

In our problem, the area will be scanned in a distance of six units. Each obstacle detected has a distance variable from the robot. This distance variable has a different membership in each of the four distance subsets. The distance universe of discourse is discretized into levels with 4 fuzzy sets, where parameters and membership functions are chosen heuristically. Around the robot, the area will be scan from –180˚ to +180˚ for the Angle. This scope is divided into eight subsets, where each subset has a membership function. The Fig. 2 shows the distribution of the Angle with respect to the eight fuzzy concepts or subsets, where the Angle universe of discourse is discretized into eight fuzzy sets, where parameters and membership functions are chosen heuristically. The linguistic variables are the same for the input and output, Angle, Distance and Direction sets.

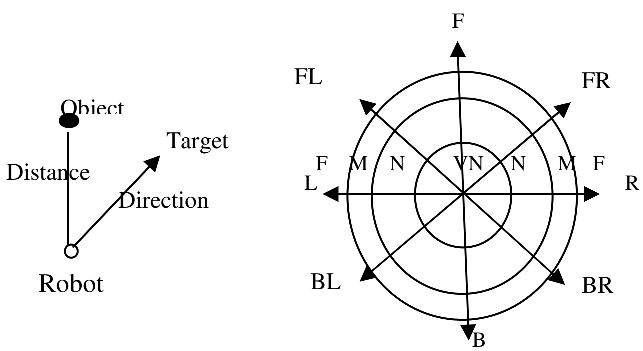

Figure 2. Division of space in Fuzzy

3.2 Membership function

The membership function is a graphical way of representation of the value of each inputs and outputs. It associates a weight factor with each of the inputs and defines functional overlap between inputs. We use the same membership functions for input and output. The membership functions associated with each scalar quantity are shown in fig 3 and fig 4.

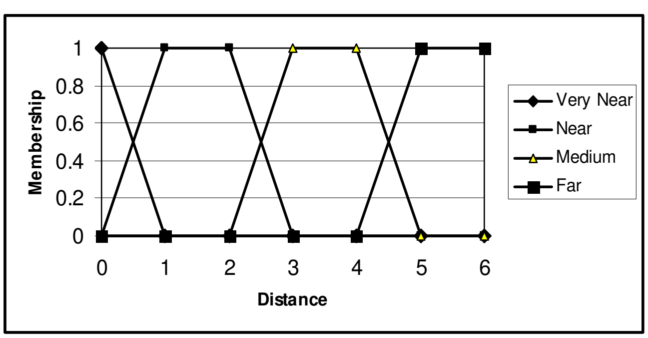

Figure 3. The membership function for distance

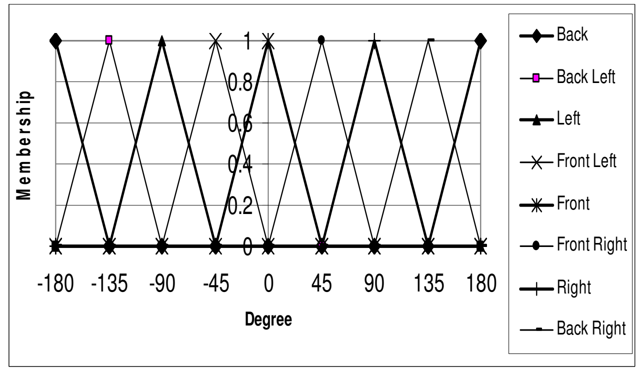

Figure 4. The membership function for Angle

For Autonomous Robot Navigation among stationary obstacle in an unknown environment, it would be computationally expensive to produce 8*3=24 and 8*7*3=168, is 192 rules. Therefore, we thought to find out the minimum set of rule by using this algorithm. In order to give the robot a certain amount of flexibility in reaching its target, this specific angle is broadened into a more general desired direction by using a set of fuzzy rules. If this were not done, the robot would not be able to turn in order to avoid any obstacles. A few examples of the fuzzy rules employed by the target following behaviour are shown below:

If Target Angle is Front **And** Movable Distance is Far **Then** Desired Direction is Front **And** Desired distance is Far

If Target Angle is Back and Movable Distance is Near **Then** Desired Direction is Back and Desired distance is Near

The Obstacle Avoidance behaviour takes its inputs as the Target Angle, possible Turning Angle and possible Distance on that turning Angle. These provide the distance to any nearby obstacles located at the Target angle of the robot. Each sensor input has its own fuzzy set which represents how close it is to an obstacle. The fuzzy set for the target direction sensor considers objects further away to be more hazardous than the other side sensors. This is due to objects having less influence on the side than ones at the Target Angle are. The fuzzy rules for this particular behaviour produce an output, which represents the Turning Direction in which the robot should turn and the Distance the robot should move. Some rules employed by this behaviour are shown below:

If (Target Angle is Front, Movable Distance is Very Near) **And** (Turning Angle is Front Right, Movable Distance is Far) **Then (**Desired Direction is Front Right and Desired Distance is Far)

If (Target Angle is Right, Movable Distance is Very Near) **And** (Turning Angle is Back Right, Movable Distance is Near) **Then (**Desired Direction is Back Right and Desired Distance is Near)

The outputs from each of the two behaviours are not crisp values but are instead fuzzy sets. One represents the desired Direction in which the robot should travel if there is no obstacle while the other represents the allowed Turning Direction of travel due to nearby obstacles. The next task is to defuzzy the outputs to get crisp values for actual movement. The equation is shown as in (1). One of the most commonly used methods for defuzzification is the Centroid method also known as Centre of Area (COA) method.

$$\Sigma\ (f(i) * output(i)) / \Sigma\ (f(i) \qquad (1)$$

Where i = 1.. N. N is the number of rule activated. The result of this calculation gives the actual output setting.

3.3 Fitness function

GA requires a scalar objective function in order to select the best individuals for reproduction of offspring. For control problems with multiple competing tasks the fitness function has to be designed carefully in order to take all objectives into account. The GA tends to converge prematurely on FLCs that are exclusively specialized to individual easy subtasks. Fitness values of GAs always reflect cumulative rewards received by learning algorithms during the whole course of interaction with the environment. They indicate the quality of a sequence of actions, rather than any individual action. The definition of the objective function is not a simple matter. This function measures the goodness of each individual. The fitness function used in our work is as in (2):

Fitness of the Rule set is defined as:

$$\text{Fitness} = A - S + (G*5) - (R*5) \qquad (2)$$

Where

A – Allowed maximum number of steps the robot can move during an episode.

S – The number of Steps used by the robot to reach the target

G – Assign 1 for G, if the goal is achieved, other wise –1.

R– Remaining distance from the current position of the Robot to the target position.

Remaining distance, R is calculated by using the formula as in (3).

$$R = \text{Remaining distance} / \text{Total distance} * 100 \qquad (3)$$

Where the remaining distances is calculated using the coordinates of the robot's current location and the target location. Total distance is calculated by using the coordinate of starting location and target location. To magnify the resulting value, we multiply that value by 100.

In case of the Mobile Robot the performance of each FLC is evaluated in, simulated, two-dimensional environments, which include corridors, dead ends and detached obstacles. It varies the parameters of the environments, such as the width and the geometry of space as well as the position of obstacles, dead ends, the start and the goal point. A training situation achieves a high fitness value. If a collision happens, or unable to achieve the target, then the FLC receives only a small fitness value. The robot is considered to be successful in goal point reaching, if it fails, it will be given negative value (-1), otherwise positive value (+1). Just to magnify the over all fitness value, we multiply this value by a constant value (5). After testing of different constant values, this has been selected.

The rule base is built in an incremental fashion by repeatedly invoking a genetic fuzzy rule generation algorithm. The genetic fuzzy system identifies those fuzzy rules that best match and correctly classify the current distribution of training examples. In each fuzzy state, multiple rules are activated. The rewards apportioned for each rule depend on the rule's contribution in fuzzy inference to final outputs. The rule's contribution in each fuzzy state is proportional to its activated degree in the state. In our algorithm, individual fitness of each rule applied in a particular episode is calculated by using the number of time a particular rule is fired. Therefore, each episode, for each rule, the number of time fired is counted and total number of rules fired is calculated as in (4).

Individual fitness = Number of time a rule is fired / total number of rules fired * 100 (4)

In this problem, the initial position and final position of the robot is known. To begin the motion of the robot, an obstacle-free search direction is first determined using the fuzzy logic control technique. There are two inputs of the FLC, namely free space distance and target angle and the output of the FLC-is the obstacle-free direction. Once the direction and distance to travel are obtained, the robot will move to that location. The same procedure will continue until the robot reaches its destination.

There are three segments in a GA string. The first segment codes the Target angle and possible Distance moved along the direction. Second segment represents the turning angle and the possible distance that robot can be moved. Final segment contains the output direction and distance the robot has to move. The fig 5 shows the instruction format.

Variable	Target Angle	Dist 1	Turning Angle	Dist 2	Direction	Dist3
Position	1	2	3	4	5	6

Condition part　　　Action part

Angle	B	BL	L	FL	F	RL	R	BR
Value	0	1	2	3	4	5	6	7

Distance	VN	N	M	F
Value	0	1	2	3

Figure 5. Instruction Format

4 Simulation Results

As we mention earlier, the objective of this work is to provide a method by which an improved set of rules governing the actions and behavior of a simple navigating and obstacle avoiding Autonomous Mobile Robot. In every episode, the robot gets the information about only the start and target locations. Initially the rules were created randomly. Then those rules were used for finding the route to reach target. Based on the successfulness of the rule set, fitness for each rule was assigned. Our simulation is written in a way it always try to move the robot to maximum locations in one step. If fails then it tries to move next maximum location and so on. The robot is considered to be successful in goal point reaching, if it fails, it will be given negative value (-1), otherwise positive value (+1). Just to magnify the over all fitness value, we multiply this value by a constant value (5). After testing of different constant values, this has been selected. Every rule begins with an initial strength that is then updated at the end of each episode for every rule that has been employed or fired. The total fitness was shared within the rules fired on that episode.

The objective function we used for the AMR model to successfully navigate the course model while maintaining boundaries, avoiding collision with obstacles, and other constraints set forth. Every rule begins with an initial strength that is then updated at the end of each episode that increases the simulation performance for every rule that has been fired. The genetic operators applied in this thesis will be those of crossover, single gene mutation. The manner in which crossover is utilized in this work is to randomly recombine gene pair of the one third of the population with the highest strength. The value is purely subjective. Single gene mutation creates new stimulus-response rules by generating random changes in a gene of an existing rule. This paper uses a completely subjective population size of 20.

The application of the simple genetic-fuzzy algorithmic manipulation is applied in this logic:

1. *Initialize the population randomly.*
2. *Simulate the task using the population.*
3. *Calculate the fitness.*
4. *Did the Robot achieve the target or exceed the maximum steps? If yes then stop, if no continue.*
5. *Assign and update strengths to the rule(s) that participated. Apply genetic operators to generate new rules.*
6. *Sort and arrange the rules by strengths.*
7. *Return to line #2 (simulation).*

The focus of this work is to emphasize that the Genetic-Fuzzy system with fewer rules can work better than a conventional FLC with more rules. The decision on the control action is only based on the current input and is made irrespective of previous information. Nevertheless the controller has learned to turn the robot in a corridor if the goal point is located in opposition to the robot's heading. The robot moves on keeping its original direction and turns around when there is space enough. The second matter of interest in the experiments was the agent's capability to deal with previously unseen situations. In all of the test environments the goal point was always reachable for the robot. Here we have shown how the rules were evolved after number of generations. In fig. 6, after 100 generations, robot was able to move only one step towards the target. The fig. 7 shows the Robot’s positions after 250 generations respectively. The Robot always, tries to move towards the target, by using the available rules. If there is no rule available to move towards the target, it will stop moving. Also it tries to avoid the obstacle by using the rules in the second set otherwise it collides with the obstacle.

Our one of the goal was to test that the rules obtained by means of genetic evolution were able to control successfully an AMR in other environments. After many generations of evolution in different environments, we got the final set of 20 rules. This supported many different starting and target points. The controller has been tested in a number of simulated environments showing good results. The robot drives around the obstacle and moves towards the targets, which shows that the autonomous agent is able to take the higher priority of collision avoidance into account. Some of the Simulation Results are shown in fig. 8 & fig 9.

5 Conclusions

The main drawback of Evolutionary techniques is their slow convergence rate and the considerable amount of time that has to be spent to conduct the evolutionary process on a real robot. Therefore, Evolutionary Algorithms have to be fast enough to get the real advantage of evolutionary robotics. Moreover, the issues related to interaction of learning with evolution have to be dealt with more carefully. The choice of the controller-encoding scheme can have a large impact on the success or failure of the Evolutionary Robotics process. The simulation we used is just a simplified instrument in combination with the application of Genetic Algorithms and Fuzzy logic to achieve the research objective. The effectiveness of the algorithm is tested through computer simulations and found to be satisfactory.

There are number of possible extensions to the work presented in this paper. Since, the work focused on fixed length linear chromosomes, one of the most obvious extensions is to variable length representations. We only concentrated on static obstacles in unknown environment. So other possibility is considering the robot with moving obstacles. In the proposed algorithm, only the rule base is optimized, author defied membership functions are used. There is a possibility of variable membership function through evolution. Evolutionary robotics will also deal with Evolutionary Robotics modeling of artificial emotion, in future. Thus, tomorrow’s evolutionary robot will be intelligent, autonomous and emotional as well.

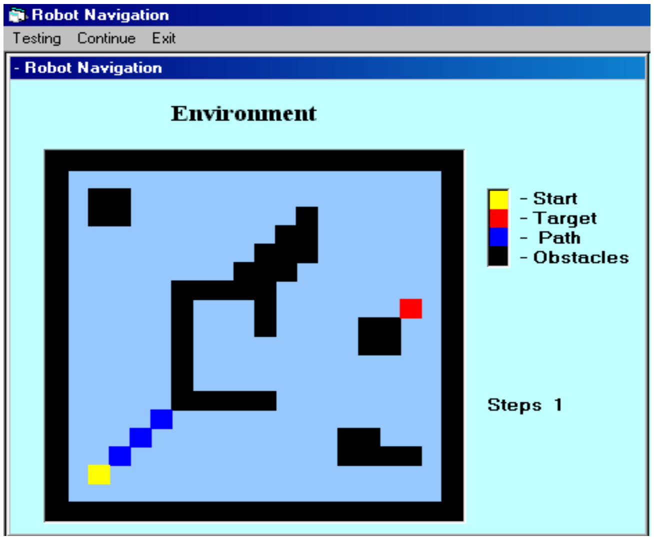

Figure 6. Simulation after 100 generations

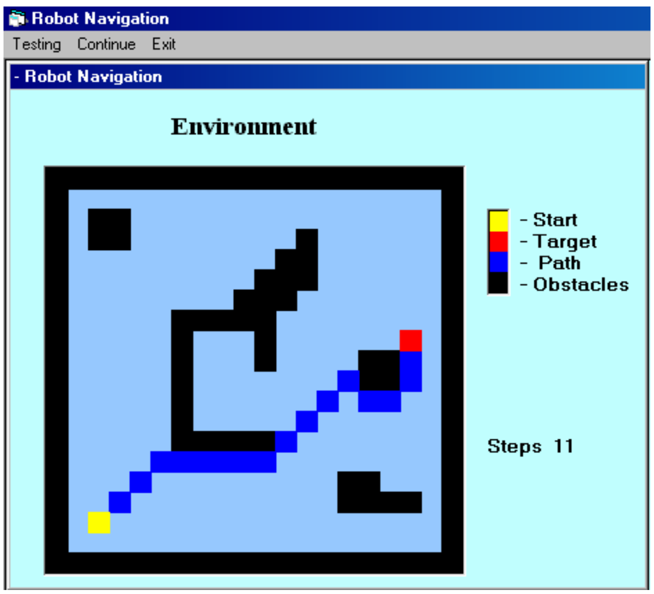

Figure 7. Simulation after 250 generations

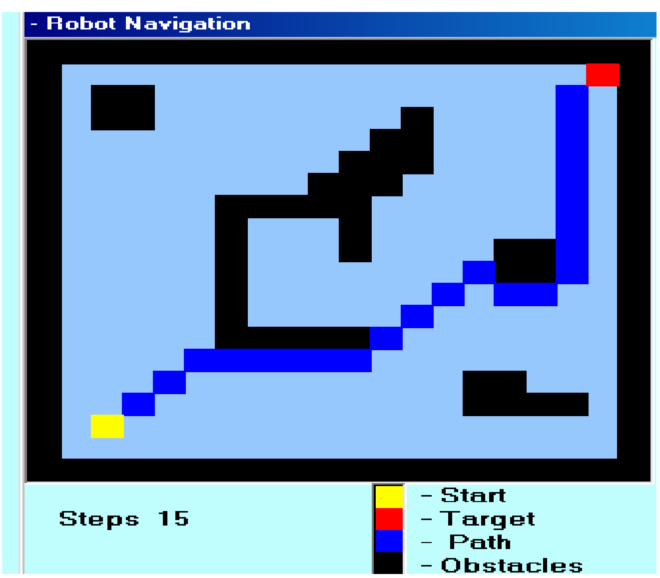

Figure 8. Simulation Results

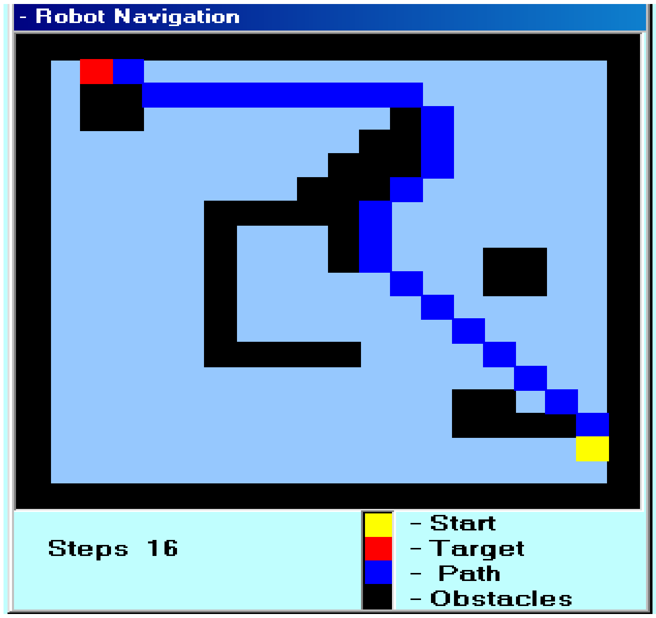

Figure 9. Simulation Results for different start end points

References

[1] Borenstein J. and Koren Y., Real-time Obstacle Avoidance for Fast Mobile Robots, *IEEE Transactions on Systems, Man, and Cybernetics*, 19(5): 1179-1187, 1989.

[2] Christopher Lazarus, Huosheng Hu, Using Genetic Programming to Evolve Robot Behaviours, *proceedings of the 3rd British Conference on Autonomous Mobile Robotics & Autonomous Systems*, Manchester, 2001.

[3] Frank Hoffmann, Soft Computing techniques for the design of Mobile Robot behaviours, *Berkeley Institute in Soft computing (BISC)*, Information Sciences, Elsevier Science Inc., 1994.

[4] Golberg D.E., Genetic Algorithms in search, optimization, and machine learning, *Addison-Wesley*. 1989.

[5] Pratihar D. K., Evolutionary robotics – A review, *Sadhana,* 28(6): 999–1009, 2003.

[6] Roland Siegwart and Illah R. Nourbakhsh, Introduction to Autonomous Mobile Robots, *the MIT Press Edition*, 2004.

[7] Saffiotti A., The use of fuzzy logic for autonomous robot Navigation, *Soft Computing*, 1(4): 180 – 197, 1997.

[8] Sakawa, M., Mori, T., An efficient genetic algorithm for job-shop scheduling problems with fuzzy processing time and fuzzy duedate, *Computers & Industrial Engineering*, 36(2):.325-41, 1999.

[9] Yun, Y., Genetic algorithm with fuzzy logic controller for preemptive and non-preemptive job-shop scheduling problems, *Computers & Industrial Engineering,* 43(3): 623-44, 2002.