Mobile Robot Navigation: Potential Field Approach Vs. Genetic-Fuzzy System

Nirmal Baran Hui[1] and Dilip Kumar Pratihar[2]

[1] Research Scholar
nirmal@mech.iitkgp.ernet.in
[2] Associate Professor
Department of Mechanical Engineering
Indian Institute of Technology, Kharagpur-721 302
India
dkpra@mech.iitkgp.ernet.in

Summary. The present paper deals with the issues related to collision-free, time-optimal navigation of an autonomous car-like robot, in the presence of some moving obstacles. Two different approaches are developed for this purpose. In the first approach, the motion planner is developed by using a conventional potential field method and a fuzzy logic-based navigator is proposed in Approach 2. In the present work, an attempt is made to develop a good knowledge base (KB) of an FLC automatically, by using a genetic algorithm (GA). During training, an optimal rule base of the FLC is determined by considering the importance of each rule. The effectiveness and computational complexity of both the approaches are compared through computer simulations.

Key words: Navigation, Car-Like Robot, Potential Field Method, Fuzzy Logic Control, Genetic Algorithm.

1 Introduction

To meet the increasing demand of robots, design and development of an autonomous robot has become a thrust area in robotic research. An autonomous robot should be able to plan its collision-free path on-line, in varying situations. The problems of collision-free navigation in a known terrain have been extensively studied by several investigators. Both graph-based as well as analytical techniques have been developed. However, all these techniques may not perform effectively in dynamic environments, where the motion of the robot is to be planned in a partially-known environment. Thus, building a complete mathematical model is not possible. Moreover, a car-like robot is subjected to both nonholonomic as well as dynamic constraints [1]. Therefore, it can only move forward or backward in a direction tangent to its trajectory.

Latombe [2] provides an extensive survey on different conventional motion planning schemes of car-like robots. Potential field method [3] has come out to be the most popular of all conventional approaches. But, its performance depends on the chosen potential functions, a proper selection of which is a tough task. Therefore, it may provide some feasible solutions to the present problem, which may not be optimal in any sense. Moreover, most of the traditional methods are unable to deal with uncertain and imprecise sensory informations. Thus, there is a need for an approach that can handle uncertainties at all levels and deal with various situations, those are not known a priori.

Fuzzy set theory had been introduced by Zadeh [4] in the year 1965, to deal with vague, uncertain and imprecise data related to the real-world problems. Recently, some researchers [5, 6] have started thinking, whether they should switch over to a fuzzy logic-based motion planner. However, the performance of an FLC depends on its KB, design of which is not an easy task. Several methods, such as least squares [8], gradient descent [9], back-propagation algorithm of neural network [10] and others, had been proposed by various investigators, to develop an optimal KB of an FLC. But, all such methods failed to give optimal solutions, due to the fact that they might have local minima problem. Moreover, the problem of knowledge acquisition of an FLC is a tough task, due to the fact that a human expert often finds it difficult to express his or her control actions. Several trials had been made by quite a few researchers, for the development of a suitable KB of an FLC, by using a GA [11]. Optimization of both the data base as well as rule base of an FLC had been carried out simultaneously using a GA by a few investigators [12, 6]. In this regard, there are three basic approaches, namely Michigan, Pittsburgh and iterative rule learning. Interested readers may refer to [7], for a detail study of the same. However, in some of these methods, a considerable amount of time was spent on manual design of rule base and a GA was used to further tune it. To overcome this difficulty, some investigators [13, 14] tried to design the FLC automatically by giving the complete task of designing a suitable KB to the GA. It is important to mention that the GA-learned rule base of an FLC may contain some redundant rules, but no effort was made to identify and remove them in the above earlier work. It happens due to the iterative nature of the GA. Realizing all such problems, an attempt is made in the present study, to design the FLC automatically, in such a manner that the redundant rules (if any) will be removed from the rule base. The effectiveness of both these approaches are tested through computer simulations, for solving the navigation problems of a car-like robot moving in a dynamic environment. It is to be noted that the present work differs from the earlier work [6] in the sense that a car-like robot has been considered including its kinematic and dynamic constraints, in place of a point robot. Thus, it is a more realistic problem compared to the earlier.

The rest of the paper is structured as follows: Dynamic motion planning problem of a car-like robot is stated and the motion planning scheme

is explained in Section 2. The developed motion planning algorithms are discussed in Section 3. Results of computer simulations are presented and discussed in Section 4 and some concluding remarks are made in Section 5.

2 Dynamic Motion Planning of a Car-Like Robot

Motion planning problem of a car-like robot navigating in the presence of some moving obstacles, is considered in this paper.

2.1 Statement of the Problem

During navigation, a car-like robot has to find its collision-free, time-optimal paths, after starting from a fixed point and to reach a target point situated in an unknown environment populated with some moving obstacles. Moreover, motion planning problem of a car-like robot is a complicated one, due to the fact that it should satisfy both nonholonomic as well as dynamic constraints during its navigation. In the present work, both the nonholonomic as well as dynamic constraints (such as motor torque constraint, curvature constraint, sliding constraint) of the car-like robot [15] have been considered. To meet these requirements, a suitable motion planning technique has to be developed, which can plan and control the motion of a car-like robot, on-line, in an optimal sense. For simplicity, all the moving obstacles are represented by their bounding circles and the robot is assumed to move due to pure rolling action only. The developed motion planning scheme is discussed below.

2.2 Motion Planning Scheme

The complete path of the robot is considered as a series of small segments, either curved or straight or a combination of them. Each segment is assumed to be traversed during a fixed time ΔT and the robot's path is planned based on the position of the most critical obstacle, which is identified by considering the relative velocity of the robot with respect to the obstacle and the direction of movement of the obstacle. It is important to mention that only one obstacle is considered to be the most critical at a time and no two obstacles are allowed to overlap each other. If the robot finds any critical obstacle ahead of it, in the predicted time step, the motion planner is activated. Otherwise, it moves with the maximum possible velocity by following a straight path, directed towards the goal. The task of the motion planner is to determine the acceleration and deviation required by the robot to avoid collision with the most critical obstacle. This process continues till the robot reaches the target and the total traveling time is calculated by adding all the intermediate time steps.

Our aim is to design the motion planner in such a manner that the robot reaches the goal with a minimum possible traveling time. Thus, the present

problem can be treated as a constrained traveling time minimization problem. It is important to note that a robot will reach the target with the minimum possible time, only when it moves with the maximum possible velocity and takes less deviation to avoid collision with the obstacles. Thus, the problem under this study, is solved by minimizing the error due to both acceleration as well as deviation required by the robot simultaneously, to avoid collisions with the most critical obstacle after satisfying the constraints.

3 Developed Motion Planning Algorithms

Two different approaches of robot motion planning are developed, in the present work, which are discussed below.

3.1 Approach 1: Potential Field Method

Potential field method, introduced by Khatib [16], is widely used for real time collision-free path planning of both manipulators as well as mobile robots. In this approach, the robot is modeled as a particle moving under the influence of an artificial potential field, which is determined by the set of obstacles and the target destination. The target is assumed to have attractive potential and the obstacles generate the repulsive potential. The movement of the robot is then achieved by determining the resultant force due to these two potential fields. However, the performance of the potential field method depends on the chosen artificial potential functions. Several potential functions, such as parabolic-well, conic-well, hyperbolic function, rotational field function, quadratic, exponential function, are tried by various investigators [3, 1], out of which, parabolic and hyperbolic functions are widely used for solving the similar problem [17], due to their nonlinear approximation capability about the system. The attractive $U_{att}(X)$ and repulsive $U_{rep}(X)$ potential fields, used in this study, can be expressed as follows.

$$U_{att}(X) = \frac{1}{2}\xi_{att}\ d^2_{goal}(X), \tag{1}$$

where ξ_{att} is a positive scaling factor of attractive potential and $d_{goal}(X)$ denotes the Euclidean distance of the robot from its goal.

$$U_{rep}(X) = \frac{1}{2}\xi_{rep}\left[\frac{1}{d_{obs}(X)} - \frac{1}{d_{obs}(0)}\right]^2, \tag{2}$$

where ξ_{rep} is a positive scaling factor of repulsive potential and $d_{obs}(X)$ indicates the Euclidean distance of the robot from the obstacle and $d_{obs}(0)$ represents the distance of influence of the obstacle and it is made equal to the center distance between the robot's bounding circle and that of the obstacle.

The attractive and repulsive potential forces are then calculated by differentiating the respective potential fields with respect to the goal distance and obstacle distance, respectively. It is important to mention that in the present study, acceleration of the robot is taken to be proportional to the magnitude of the resultant force and deviation is considered as the angle made between the direction of the resultant potential force and the new reference line joining the CG of the robot at the present time step and the goal position.

3.2 Approach 2: Genetic-Fuzzy System

An FLC may also provide some feasible solutions to the said problem. Two condition variables, such as (i) *distance* of the robot from the most critical obstacle and (ii) *angle* between the line joining the robot and the most critical obstacle and the reference line (joining the robot and its goal) are fed as inputs to the controller. The outputs of the controller are taken to be *deviation* and *acceleration* required by the robot to avoid collision with the most critical obstacle. In the present study, the range of *distance* is divided into four linguistic terms: very near (VN), near (NR), far (FR), very far (VF). Five linguistic terms have been considered for both the *angle* as well as *deviation*: left (LT), ahead left (AL), ahead (AH), ahead right (AR) and right (RT) and *acceleration* is considered to have four terms: very low (VL), low (L), high (H), very high (VH). Therefore, there will be a maximum of twenty input conditions, and for each input condition, there is a maximum of twenty output combinations. Thus, there is a maximum of 400 (i.e., 20×20) rules present in the rule base and a particular rule will look like the following.

IF *distance* is VF AND *angle* is LT, THEN *deviation* is AH and *acceleration* is VH.

For ease of implementations, membership function distributions of both the input as well as output variables are assumed to be symmetric triangles. Thus, the data base of the FLC may be represented by providing the four continuous variables representing the half base-widths of the triangular membership function distributions. The performance of an FLC depends on its both data base as well as rule base, which are to be optimized simultaneously. In the present work, an attempt is made to develop a good KB of an FLC automatically by using a binary-coded GA. A GA-string consisting of 440-bits is considered to indicate the KB of the FLC as shown below.

$$\underbrace{\underbrace{10\cdots1}\ \underbrace{01\cdots1}\ \underbrace{10\cdots0}\ \underbrace{01\cdots0}}_{\textit{Data base}}\quad \underbrace{10\cdots01}_{\textit{Input combinations}}\quad \underbrace{10101\ldots0101\ldots11001}_{\textit{Consequent of the rules}}$$

The first 40-bits in this string represent the half base-widths of the four triangles (10 bits for each variable) and the next 20-bits are used to indicate the presence or absence of the input combinations in the rule base (1 for presence and 0 for absence). Out of the remaining 380-bits of the string, every 19-bits will carry the information regarding the combination of the consequents,

for a particular input condition. We count the number of $1s$ present in each 19-bits long sub-string. If it comes out to be zero, it will represent the first output combination, i.e., deviation is LT and acceleration is VL, and so on. The fitness of a GA-string is then calculated, as follows.

$$Fitness = \frac{1}{N}\sum_{n=1}^{N}\frac{1}{S}\sum_{s=1}^{S}\sum_{v=1}^{2}(T_{nsv} - O_{nsv}), \tag{3}$$

where S denotes the total number of time steps in a planned path and the total number of training scenarios is indicated by N. O_{nsv} and T_{nsv} are representing the values of actual output and target output, respectively, of an output variable (say,v). The target output for deviation is considered to be equal to zero and that for acceleration is taken as the maximum permissible acceleration of the robot. A fixed penalty equals to 200 is added to the said fitness value, when the robot collides with the most critical obstacle during its movement from the present position to the predicted location. Moreover, another fixed penalty equals to 2000 is given to the string, if the FLC represented by it, is unable to provide any solution particularly in case of non-firing situation or the generated motion of the robot fails to satisfy the dynamic and/or kinematic constraints.

During optimization, an optimal rule base of the FLC is determined by considering the importance of each rule, which is calculated as $I_{ij} = p_{ij}\ \overline{\overline{C}}_j$, where p_{ij} denotes the probability of occurrence of j^{th} output combination corresponding to i^{th} input condition of the rule, where $i, j = 1, 2, \cdots, 20$ and $\overline{\overline{C}}_j = \frac{1}{2}\ (\overline{C}_q + \overline{C}_r)$, where $\overline{C}_q$ and $\overline{C}_r$ are the average worth of q^{th} linguistic term of the first output (i.e., deviation) and r^{th} term of acceleration output, respectively. It is important to note that the worth, corresponding to a linguistic term of an output, is determined by following the Gaussian distribution pattern, maximum being occurred for deviation output AH and acceleration output VH.

4 Results and Discussion

In the present study, a car-like robot is allowed to navigate among sixteen obstacles moving in a 2-D environment of size equal to $19.95 \times 19.95m^2$. To provide training to the FLC, two hundred training scenarios are generated at random. A particular training scenario is different from the other, in terms of the initial position of the moving obstacles, their size, speed and direction of movement. During optimization, half-base width of four triangular membership function distributions are varied in the ranges of (0.8,1.0), (20,40), (20,40) and (0.005,0.015), respectively. The ranges of variation for the different variables are selected through a careful study. Moreover, it is to be noted that the time interval (ΔT) is taken to be equal to sixteen seconds and the maximum and minimum accelerations of the robot are set equal to $0.05m/s^2$

and $0.005m/s^2$, respectively. During training, the best results are obtained with the following GA-parameters: crossover probability $p_c = 0.84$, mutation probability $p_m = 0.00166$, population size $Y = 140$, maximum number of generation $Maxgen = 98$.

Twelve good rules have been identified by the GA, out of which, two rules are found to be redundant (refer to Fig. 1). A rule is said to be redundant and may be eliminated, if the *importance factor* of that rule comes out to be smaller than a pre-specified value and the removal of which does not lead to any non-firing situation. It is observed that the number of non-firing incidences increases with the reduction of number of rules present in the rule base, as shown in Fig. 1.

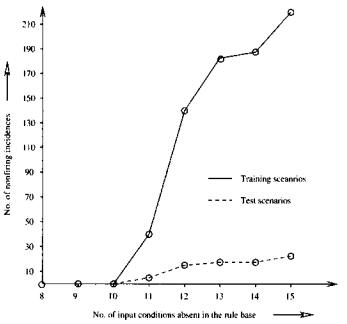

Fig. 1. Number of rules absent in rule base vs. number of non-firing incidences.

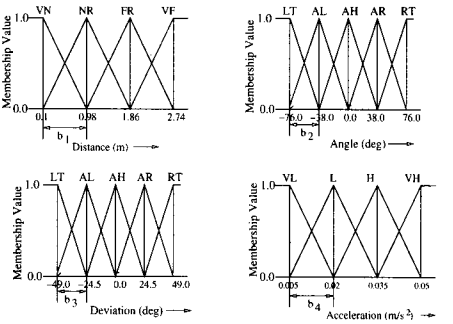

Fig. 2. Optimized membership function distributions of the FLC.

The optimized rule base along with the importance factors of the rules are shown in Table 1. In case of first and third rules of the optimized rule base

Table 1. Optimized rule base of the FLC.

Distance	Angle	Deviation	Acceleration	Worth $(\overline{\overline{C}}_j)$	Probability of occurrences	Importance factor
VN	AL	AL	VH	0.792484	0.038017	0.030128
VN	AH	AH	VH	1.000000	0.067505	0.067505
VN	AR	AR	VH	0.792731	0.046960	0.037227
NR	AH	AH	VH	1.000000	0.153483	0.153483
FR	LT	AH	VL	0.515048	0.052486	0.027033
FR	RT	AH	VL	0.515048	0.045074	0.023215
VF	LT	AH	VH	1.000000	0.029633	0.029633
VF	AL	AH	H	0.918319	0.105903	0.097253
VF	AH	AH	H	0.918319	0.157644	0.144767
VF	RT	AH	L	0.630077	0.040236	0.025351

(refer to Table 1), both the angle as well as deviation are coming out to be the same with respect to the linguistic terms, namely AL and AR, respectively. It is important to note that although the linguistic terms are the same, their numerical values are coming out to be the different during optimization. It happens because the membership function distributions of deviation will try to squeeze, to ensure a minimum traveling time. On the other hand, the optimizer may choose a wider membership function distributions for the angle input, just to increase the view angle of the robot. For example, the crisp values corresponding to AL are coming to be equal to $-38.0°$ and $-24.5°$ for angle and deviation, respectively (refer to Fig. 2). Thus an AL, angle may not be exactly equal to a deviation expressed by the same linguistic term – AL.

A close watch on the 2^{nd} and 4^{th} rules of the optimized rule base as shown in Table 1, reveals the fact that the linguistic terms for both angle as well as deviation are coming to be the same as AH. It may be due to the reason that during training, if there is a chance of the robot, in a particular step, that it collides with the most critical obstacle, a penalty equal to 200 is added to the fitness of the GA-string, to come out of this situation. It occurs, when the direction of movement of the most critical obstacle, in that particular step, intersects the direction of movement of the robot as planned by the motion planner. However, this penalty helps the robot to cross the predicted position of the obstacle as early as possible. Moreover, a VH acceleration as planned by the motion planner helps the robot to ensure this.

For FR and VF distances (irrespective of the angle input), the robot tries to navigate towards the AH direction, to achieve a time-optimal path. The optimized half base-widths of the membership function distributions are found to be as follows: $b_1 = 0.877$ (for *distance*), $b_2 = 37.947$ (for *angle*), $b_3 = 24.496$ (for *deviation*) and $b_4 = 0.0149$ (for *acceleration*) (refer to Fig. 2). The performances of both the approaches are tested for twenty test scenarios, created at random. Results of computer simulations are shown in Fig. 3 and it is observed that Approach 2 has outperformed Approach 1 in all the test scenarios. It could be due to the fact that in potential field method, the robot often may get stuck in the local minima. Moreover, it does not have any in-built optimization module. The movement of the obstacles as well as the robot in a particular test scenario (say, 4^{th}), is shown in Fig. 4, in detail. It has been observed that Approach 1 is unable to provide good solutions, when any obstacle comes near to the line joining the robot and the goal or it moves perpendicular to the direction of movement of the robot. However, the GA-learned FLC has understood these situations well and behaved optimally. The CPU time of both the approaches are found to be small, making them suitable for solving the navigation problems of a car-like robot in a partially-known environment, on-line.

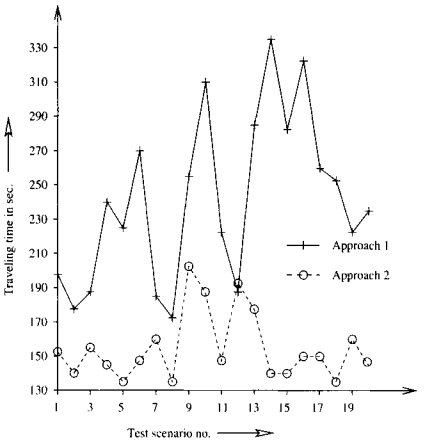

Fig. 3. Comparison of two approaches in terms of traveling time taken by the robot.

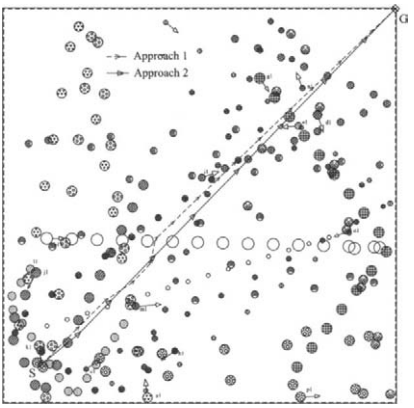

Fig. 4. Collision-free paths obtained by the robot using two different approaches.

5 Concluding Remarks

Potential field method is a widely used approach for robot motion planning, although it has a number of drawbacks. However, fuzzy logic-based motion planners are drawing interest, nowadays, due to its ease of implementation and ability to deal with imprecise and uncertain sensory readings. Several methods have been developed to find a suitable KB of an FLC, but each of these approaches has its inherent limitations. A method for automatic design of FLC is undertaken in the present study, in which a binary-coded GA is used to develop a good KB of an FLC. During training, an optimal rule base of the FLC is determined by the GA and it is further modified by considering the importance of each rule.

The effectiveness of these two approaches are studied, to solve the navigation problems of a car-like robot, in the presence of sixteen moving obstacles, through computer simulations. Approach 2 has proved its supremacy over Approach 1, for twenty randomly-generated test scenarios. It may be due to the fact that there is no in-built optimization module in the potential field method and it may have local minima problem also. It is interesting to note that importance of each rule present in the GA-designed rule base is determined to identify the redundant rules (which may be eliminated), if it does not lead to any non-firing incidence. Therefore, the present genetic-fuzzy approach may be utilized to develop an optimal KB of an FLC, which will contain only the significant rules. Moreover, as optimization of the FLC is carried out off-line, it might be suitable for solving the motion planning problems of a car-like robot, on-line. The performances of the present approaches are tested on computer simulations only. However, it will be more interesting to see their performances on a real robot.

Acknowledgment

This work is supported by the Department of Science and Technology, Govt. of India (Sanction No. SR/S3/RM/28/2003 dt. 12.12.2003). The authors gratefully acknowledge the cooperation of Dr. A. Roy Chowdhury of Department of Mechanical Engineering, IIT–Kharagpur, India.

References

1. Feng D, Krog H, Bruce H (1990), Dynamic steering control of conventionally steered mobile robots, Proc. of IEEE Conference on Robotics and Automation, 390–395.
2. Latombe J C (1991) Robot motion planning. Kluwer Academic Publishers
3. Bemporad A, Luca A D, Orilo G (1996), Local incremental planning for a car-like robot navigating among obstacles, Proc. of IEEE Conference on Robotics and Automation, Minneapolis, Minnesoia, 1205–1211.
4. Zadeh L A (1965), Fuzzy sets, Information and Control 8:338–353.
5. Fraichard T, Garnier P (2001), Fuzzy control to drive car-like vehicles, Robotics and Autonomous Systems, 34:1–22.
6. Pratihar D K, Deb K, Ghosh A (1999), A genetic-fuzzy approach for mobile robot navigation among moving obstacles, International Journal Approximate Reasoning, 20:145–172.
7. Cordon O, Herrera F, Hoffman F, Magdalena L (2001) Genetic fuzzy systems: Evolutionary tuning and learning of fuzzy knowledge bases, World scientific.
8. Pham T D, Valliappan S (1994), A least squares model for fuzzy rules of inference, Fuzzy Sets and Systems, 64:207–212.
9. Nomura H, Hayashi I, Wakami W (1992), A learning method of fuzzy inference rules by descent method, Proc. of IEEE International Conference on Fuzzy Systems, San Diego, CA.
10. Jang J R (1992), Self-learning fuzzy controllers based on temporal back propagation, IEEE Transactions on Neural Networks, 3:714–721
11. Goldberg D E (1989) Genetic algorithms in search, optimization, machine learning. Addison-Wesley, Reading, Mass, USA
12. Pham D T, Karaboga D (1991), Optimum design of fuzzy logic controllers using genetic algorithms, Journal of Systems and Engineering, 1:114–118.
13. Nandi A K, Pratihar D K (2004), Automatic design of fuzzy logic controller using a genetic algorithm-to predict power requirement and surface finish in grinding, Journal of Material Processing Technology, 148:288–300.
14. Magdalena L, Monasterio F (1997), A fuzzy logic controller with learning through the evolution of its knowledge base, International Journal of Approximate Reasoning, 16(3–4):335–358.
15. Shiller Z, Gwo Y R (1991), Dynamic motion planning of autonomous vehicles, IEEE Transactions on Robotics and Automation, 7(2):241–249.
16. Khatib O (1986), Real-time obstacle avoidance for manipulators and mobile robots, International Journal of Robotics Research, 5(1):90–98.
17. Biswas D K (2003) Path planning of multiple robot working in the same workspace–potential field approach. M.Tech thesis, REC Durgapur, India