# AN EMBEDDED GENETIC FUZZY MOTION CONTROLLER FOR A MOBILE ROBOT

**Simon X. Yang** [*,**] **Xiaochuan Wang** [**]
**Guoyin Wang** [*] **Max Q.-H. Meng** [***]

[*] *Institute of Computer Science and Technology, Chongqing University of Posts and Telecommunications, China*
[**] *School of Engineering, University of Guelph, Canada*
[***] *Department of Electronics Engineering, Chinese University of Hong Kong, China*

Abstract: In this paper, an embedded genetic fuzzy motion control system is developed for autonomous navigation and obstacle avoidance of a mobile robot built in the ARIS Lab. A combination of four infrared sensors is equipped to measure the distance to obstacles around the mobile robot. The distance information is processed by the proposed genetic fuzzy controller to adjust the velocities of two separately-driven wheels of the robot. Here the fuzzy logic is used to process the sensor information generating the control commends; and the genetic algorithm is exploited to tune the membership functions of the fuzzy rules. Local search techniques are utilised to enhance the tuning process. The proposed motion controller has been applied on the mobile robot. Experiments have demonstrated the effectiveness of the developed genetic fuzzy controller. 

Keywords: Fuzzy Control, Mobile Robot, Obstacle Avoidance, Genetic Algorithm

## 1. INTRODUCTION

Autonomous navigation and obstacle avoidance have become the subject of many researches in artificial intelligence. Autonomous navigation is related to the ability of a mobile robot moving around in an unknown environment to achieve a goal without hitting any obstacles. The mobile robot is guided by online sensor information acquired while navigation is performed. Such tasks require different abilities to execute actions which lead to goal achievements. Fuzzy logic is an well-known organised method for handling imprecise data. It offers a possibility to mimic expert human knowledge. However, the lack of systematic learning capacity for conventional fuzzy logic system design generated a certain interest in the combination of fuzzy logic and other learning methods like Genetic Algorithm (GA). Genetic Algorithm has the capability to search the optimal solutions automatically and robustly. The integration of fuzzy logic and genetic algorithms, i.e., genetic fuzzy system, gives us a better way to realise motion control for a mobile robot. It inherits the advantages of both systems: fuzzy logic can treat ambiguous and imprecise sensor information based on a set of linguistic rules; genetic algorithm is used to tune the parameters of fuzzy logic system off-line so as to ensure better performance.

In the past two decades, many methods have been developed to realise obstacle-avoidance action on mobile robots. Borenstein and Koren's (1991) proposed 'histogramic in-motion mapping algorithm', which works well in a known environment. But it requires heavy computation and large memory. They also developed and implemented a method of virtual force field for mobile

robots in Borenstein and Koren (1989). However, they did not mention how to apply the theory to obstacle-avoidance action. Akishita *et al.*'s (1993) presented an algorithm using Laplace potential theory. All these methods can only work well in a known environment.

Fuzzy logic and artificial neural networks (Zhang and Peng, 1998) are normally used for obstacle avoidance in unknown environments. Neuro-fuzzy model combines the advantages of both fuzzy logic system and neural networks. It keeps the simplicity of fuzzy system while endowing the system with learning capability. Godjevac's (1995) proposed a neuro-fuzzy model for a mobile robot to avoid obstacles. But more than 600 basic rules are used while many of them are redundant. The method of how to suppress useless rules is not introduced. In addition, the physical meanings of membership functions have been lost during the training.

In this paper, a novel GA-fuzzy model is presented to realise obstacle avoidance for a nonholonomic mobile robot. It provides a simple but effective approach to the robot motion planning. Under the control of the proposed GA-fuzzy model, the mobile robot can "see" the environment around and and explore the unknown environment automatically without hitting any obstacles. A local search method has been combined with GA tuner so as to enhance the tuning process.

## 2. THE MOBILE ROBOT AND ITS KINEMATIC MODEL

A mobile robot (Fig. 1) is developed for testing the proposed neuro-fuzzy controller. The mobile robot is composed by three parts: the locomotion system, the sensing system and the programmable reasoning (control) system, as shown in Fig. 2.

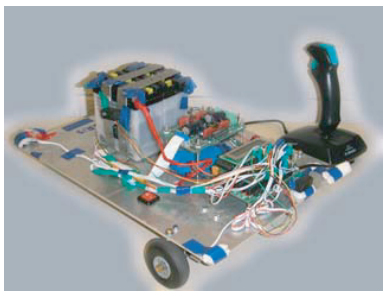

Fig. 1. The mobile robot.

Differential drive is used for the locomotion of the robot. It has two co-axle fixed wheels driven by different motors separately, and a third passive omni-directional caster to keep balance. Through adjusting the velocity of the two driven wheels respectively, the velocity and motion direction of

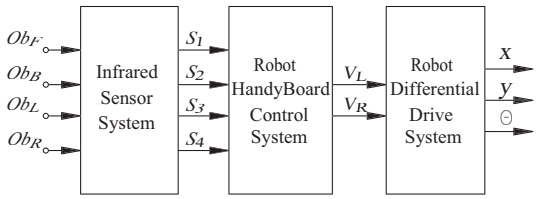


Fig. 2. The control architecture of the mobile robot.

the mobile robot can be determined. The kinematic model of the drive system is shown in Fig. 3. At any time, it can be described as

$$\begin{pmatrix} \dot{x} \\ \dot{y} \\ \dot{\theta} \end{pmatrix} = \begin{pmatrix} 0.5cos\theta & 0.5cos\theta \\ 0.5sin\theta & 0.5sin\theta \\ -1/L & 1/L \end{pmatrix} \begin{pmatrix} V_L \\ V_R \end{pmatrix}, \qquad (1)$$

where $\dot{x}$, $\dot{y}$ and $\dot{\theta}$ are the velocities in $X$, $Y$ direction and the angular velocity of the cart, and $V_L$ and $V_R$ are the velocities of the left and right driven wheels.

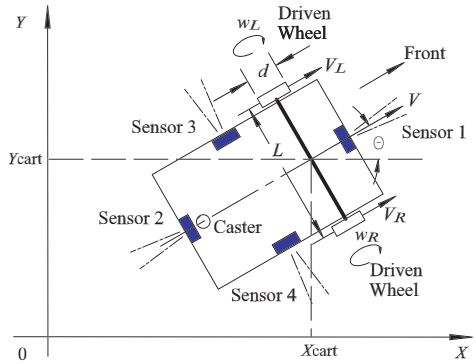


Fig. 3. The kinematic model of the mobile robot.

To detect the obstacles around effectively, a combination of 4 SHARP GP2D02 infrared sensors is used to facilitate navigation and obstacle avoidance. The sensors are located on the four sides of the cart to detect the obstacles around (Fig. 3). The effective detection scope for these sensors is approximately 10cm to 80cm. The sensor data are obtained through an 8-digit AC/DC sampler before they are processed by the microprocessor. So the data range is from 0 to 255. Here the supplementary part of sensor data are used as the sensor input $I_i$ of the neuro-fuzzy controller, i.e., $I_i = 255 - S_i$, where $i = 1, 2, 3, 4$ is the index of the sensors. Thus $I_i$ increases when the distance increases in the detection range.

Handy Board is applied on this mobile robot as the central control system. The Handy Board is a hand-held, battery-powered micro-controller board developed by MIT. Based on the Motorola 68HC11 microprocessor, 32K RAM, it can be programmed by language of Interactive C and execute some certain tasks including receiving data from sensors, reasoning and driving the DC motors. It is ideal for this mobile robot.

*2.1 The genetic fuzzy model*

To achieve obstacle avoidance, four genetic fuzzy controllers in parallel are used (Fig. 4). The inputs are the readings from the infrared sensors ($I_{1-4}$), the outputs are the sub-velocities ($V_{L1}$, $V_{L2}$, $V_{R1}$ and $V_{R2}$) of the driven wheels. Then the velocities of left and right wheels, $V_L = V_{L1} + V_{L2}$ and $V_R = V_{R1} + V_{R2}$, are applied to the motion control of the mobile robot.

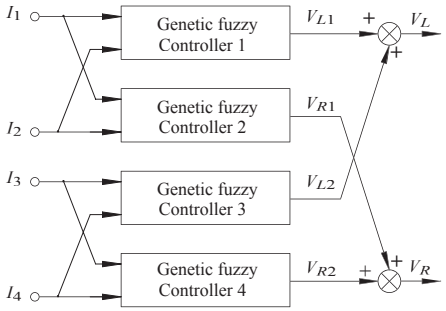


Fig. 4. Block diagram of the proposed neuro-fuzzy controller.

The four genetic fuzzy controllers have the same architecture except different linguistic rule bases. So in the following part we can use Genetic Fuzzy Controller 1 as an example. The schematic diagram of Genetic Fuzzy Controller 1 is shown in Fig. 5. It consists of two blocks: fuzzy logic controller block and GA tuner block. Fuzzy logic controller ties the input and output together. It treats the sensor information and generate the output. GA tuner can tune the parameters of fuzzy logic controller off-line by comparing the output and the target values. Both blocks will be discussed in details in the following part.

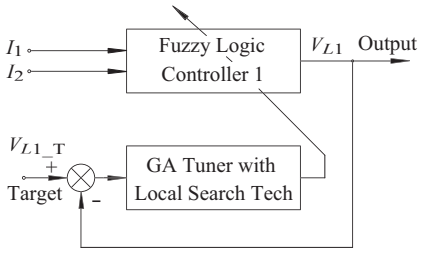


Fig. 5. Schematic diagram of the GA-fuzzy controller 1.

*2.2 Fuzzy logic controller*

The fuzzy logic controller 1 is composed of four-layer networks constructed under fuzzy logic rules, as shown in Fig. 6. The first layer is input layer. There are two inputs ($I_1$ and $I_2$) on this layer. Then the second layer fuzzifies each input into four membership functions. The third layer performs the inference mechanism on the base of fuzzy rules. The result can be obtained from the output layer through defuzzification.

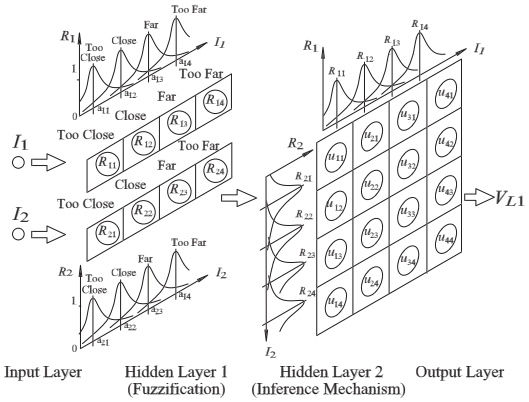


Fig. 6. Schematic architecture of Controller 1.

The fuzzification process is to map the crisp input values into the linguistic fuzzy terms with the membership functions between 0 and 1. In this paper, Gaussian functions are used to represent the fuzzy membership function.

Four membership functions are defined for each infrared sensor according to the distance to the obstacles (shown in Fig. 7). The definition of membership functions is given as

$$R_{ij} = \exp\left(-\frac{(I_i - a_{ij})^2}{2\sigma_{ij}^2}\right), \qquad (2)$$

where $i = 1, 2, 3, 4$; and $j = 1, 2, 3, 4$.

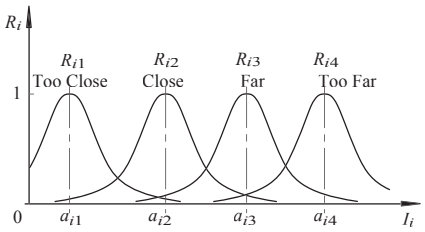


Fig. 7. Input membership functions.

The obstacle detection is defined on four levels - Too Close (TC), Close (C), Far (F) and Too Far (TF). The linguistic rule base is the combination of a set of "IF-THEN" rules. To achieve the obstacle avoidance, 32 rules are defined in Tables 1 and 2 for the front/rear infrared sensors $I_1/I_2$ and left/right infrared sensors $I_3/I_4$ respectively. For example, the first rule in Table 1 can be described as: IF $I_1$ is Too Close and $I_2$ is Too Close, THEN $V_{L1}$ is $Medium$ and $V_{R1}$ is $Medium$.

Table 1. Fuzzy rule base 1 for $I_1$&$I_2$.

| $V_{L1}/V_{R1}$ | | | $I_2$ | | | |
|---|---|---|---|---|---|---|
| | | | $R_{21}$ | $R_{22}$ | $R_{23}$ | $R_{24}$ |
| | | | TC | C | F | TF |
| $I_1$ | $R_{11}$ | TC | M/M | M/M | M/M | M/M |
| | $R_{12}$ | C | S/M | S/M | S/M | S/M |
| | $R_{13}$ | F | S/F | S/F | S/F | S/F |
| | $R_{14}$ | TF | Z/FF | Z/FF | Z/FF | Z/FF |

Table 2. Fuzzy rule base 2 for $I_3$&$I_4$.

| $V_{L1}/V_{R1}$ | | | $I_4$ | | | |
|---|---|---|---|---|---|---|
| | | | $R_{41}$ | $R_{42}$ | $R_{43}$ | $R_{44}$ |
| | | | TC | C | F | TF |
| $I_3$ | $R_{31}$ | TC | Z/Z | S/Z | M/Z | F/Z |
| | $R_{32}$ | C | Z/S | Z/Z | S/Z | M/Z |
| | $R_{33}$ | F | Z/M | Z/S | Z/Z | S/Z |
| | $R_{34}$ | TF | Z/F | Z/M | Z/S | Z/Z |

From the rule bases mentioned above, we can find that $I_1/I_2$ and $I_3/I_4$ have different contributions to the final velocities of left and right wheels, $V_L$ and $V_R$. In Table 1, the outputs of $V_{L1}$ and $V_{R1}$ are mainly related to the distance to the front obstacles. If there is an obstacle lying in the front of the robot, the robot will turn a certain angle to avoid the obstacle. The shorter the distance, the larger the turning angle. In Table 2, if there is an obstacle on the left/right of the robot, the robot will turn to the opposite direction to avoid the obstacle. The output velocity of $V_{L2}$ and $V_{R2}$ depends on the difference between $I_3$ and $I_4$. When the difference becomes larger, the difference between the output velocities of $V_{L2}$ and $V_{R2}$ becomes larger too. If some rules in Tables 1 and 2 are fired simultaneously, the combination of the motion will let the robot go forward/backward while turning at a certain angle so as to avoid the obstacles on each direction.

There are many algorithms for fuzzy inference. Normally the Mamdani arithmetic rule (minimum operation) is used. But this algorithm is not applicable to our study, because the derivatives of the functions do not exist, which are required for the training of the proposed genetic fuzzy system. Since some important parameters in the derivatives are lost in the minimum operation, the neural network cannot be learnt quickly. Here the Larson arithmetic rule – the product operator is used. So the firing strength of each rule can be expressed as

$$u_{ij} = R_{1i}R_{2j}, \quad i = 1,2,3,4; j = 1,2,3,4. \quad (3)$$

The defuzzification process maps the output from the fuzzy inference mechanism to a crisp value. The method of 'centroid of area' has been used in the proposed fuzzy logic controller, which combines the outputs represented by the implied fuzzy sets from all rules and generates the gravity centroid of the possibility distribution for a control action. Thus the crisp value of output can be obtained. In this case, the evaluation of the velocity of $V$, including $V_{L1}$, $V_{L2}$, $V_{R1}$ and $V_{R2}$, is given by

$$V = \frac{\sum_{i=1}^{4}\sum_{j=1}^{4} u_{ij}w_{ij}}{\sum_{i=1}^{3}\sum_{j=1}^{3} u_{ij}}. \quad (4)$$

### *2.3 GA tuner with local search techniques*

Due to the nonlinear characteristics of the sensor system and fuzzy logic control system, it is very hard to tune the parameters of fuzzy logic controller and get desired values by trials. GA tuner endows learning capacity to fuzzy logic controller. It plays an important role in deriving the fit solution. It provides an effective way to optimise the fuzzy logic controller.

*2.3.1. Chromosome Encoding* The following parameters have been encoded genetically

$$z = \{a_{11}, \cdots, a_{14}, a_{21}, \cdots, a_{24}, \sigma_{11}, \cdots, \sigma_{14}, \sigma_{21}, \cdots, \sigma_{24}, w_{11}, \cdots, w_{44}\}. \quad (5)$$

Here 24-digit binary code is used to represent each parameter. Each Chromosome can be treated as a *grammar*, i.e., it can be translated into an independent robot, like the chromosome to man. So the total length of the chromosome is $(4+4+4+4+16)*24 = 768$.

*2.3.2. Fitness Measurement* The environment determines the evolution objective of the mobile robot. In order to evaluate the navigation and obstacle-avoidance performance of the evolved solution, the chromosomes need to be first translated into controller parameters, then a group of target data sets is provided as the expert opinion to judge these parameters. Each data set includes the sensor values and the desired velocities on this certain condition. Thus all the situations the robot may encounter in the real environment can be tested. Here the objective is to reduce the difference between the real outputs and desired outputs. The fitness function can be given as

$$F = \sum_{m=1}^{M} V_m - V_m^T, \quad (6)$$

where $M$ is the number of data sets, $V$ is the controller output, and $V_T$ is the desired output.

*2.3.3. Local search technique* GA uses parallel search technique to explore the whole solution domain. It could avoid local optima, but it is not well suited for finely tuned search. In other words, the tuning process is quite slow. On the contrary, improvement heuristics like local search technique can find local optima quickly, but it is hard to find a global solution. Thus the hybridisation of GA with local search technique, can give the tuning process higher efficiency and better effect.

The least mean square method is used for the local search. The cost function is defined as

$$E(z) = \frac{1}{2}(V - V_T)^2 = \frac{1}{2}\Delta V^2, \tag{7}$$

where $V$ is the output velocity and $V_T$ is the target velocity. The iterative procedure is defined as

$$z^{(n+1)} = z^{(n)} - \eta \frac{\partial E}{\partial z^{(n)}}, \tag{8}$$

where $n$ is the iteration times, and $\eta$ is the learning rate. So the equations for the local adaptation of the model parameters shall be

$$\begin{aligned} a_{1i}^{(n+1)} &= a_{1i}^{(n)} - \eta_a \frac{\partial E}{\partial a_{1i}^{(n)}} \\ &= a_{1i}^{(n)} - \eta_a \frac{\Delta V^{(n)}(I_1 - a_{1i}^{(n)}) R_{1i}^{(n)} A_2^{(n)}}{(\sum_{i=1}^{4}\sum_{j=1}^{4} u_{ij})(\sigma_{1i}^{(n)})^2}, \end{aligned} \tag{9}$$

$$\begin{aligned} a_{2j}^{(n+1)} &= a_{2j}^{(n)} - \eta_a \frac{\partial E}{\partial a_{2j}^{(n)}} \\ &= a_{2j}^{(n)} - \eta_a \frac{\Delta V^{(n)}(I_2 - a_{2j}^{(n)}) R_{2j} A_1^{(n)}}{(\sum_{i=1}^{4}\sum_{j=1}^{4} u_{ij})(\sigma_{2j}^{(n)})^2}, \end{aligned} \tag{10}$$

$$\begin{aligned} \sigma_{1i}^{(n+1)} &= \sigma_{1i}^{(n)} - \eta_\sigma \frac{\partial E}{\partial \sigma_{1i}^{(n)}} \\ &= \sigma_{1i}^{(n)} - \eta_\sigma \frac{\Delta V^{(n)}(I_1 - a_{1i}^{(n)})^2 R_{1i}^{(n)} A_2^{(n)}}{(\sum_{i=1}^{4}\sum_{j=1}^{4} u_{ij})(\sigma_{1i}^{(n)})^4}, \end{aligned} \tag{11}$$

$$\begin{aligned} \sigma_{2j}^{(n+1)} &= \sigma_{2j}^{(n)} - \eta_\sigma \frac{\partial E}{\partial \sigma_{2j}^{(n)}} \\ &= \sigma_{2j}^{(n)} - \eta_\sigma \frac{\Delta V^{(n)}(I_2 - a_{2j}^{(n)})^2 R_{2j} A_1^{(n)}}{(\sum_{i=1}^{4}\sum_{j=1}^{4} u_{ij})(\sigma_{2j}^{(n)})^4}, \end{aligned} \tag{12}$$

$$\begin{aligned} w_{ij}^{(n+1)} &= w_{ij}^{(n)} - \eta_w \frac{\partial E}{\partial w_{ij}^{(n)}} \\ &= w_{ij}^{(n)} - \eta_w \frac{u_{ij}^{(n)} \Delta V^{(n)}}{\sum_{i=1}^{4}\sum_{j=1}^{4} u_{ij}^{(n)}}, \end{aligned} \tag{13}$$

where

$$\Delta V^{(n)} = V^{(n)} - V_T, \tag{14}$$

$$A_2 = \sum_{j=1}^{4}[(w_{ij} - V) R_{2j}], \tag{15}$$

$$A_1 = \sum_{i=1}^{4}[(w_{ij} - V) R_{1i}], \tag{16}$$

and $\eta_a$, $\eta_\sigma$ and $\eta_w$ are the learning rates for parameters training. The local search process stops when it finished the preset iterations ($t > N$).

*2.3.4. GA tuning procedure* In order to get more suitable motion controller for free navigation and obstacle avoidance, a generation of chromosomes is produced randomly at first. Each chromosome represents the grammar of an independent motion controller. Then these controllers are evaluated by the fitness measures mentioned above. The better controller gets higher possibility to participate crossover process when the next generation is produced. To enhance the tuning performance, local search technique has been applied to each new chromosome. Repeating this process, the solution evolved in each generation will become better and better little by little. Finally, the motion controller with good obstacle-avoidance performance can be derived.

In order to improve the GA evolution process, some strategies, like tournament selection and elitism strategies, have been used. The tuning process stops when the program finished the preset generations ($t > N$). The evolution procedure can be generalised as Fig. 8:

```
BEGIN
  Population Generation;
  Do
    Population Fitness Evaluation;
      Chromosome Translation;
      For n=1:T
        Determine SensorLocation (x, y, Θ);
        Determine Min. Distance to Obstacles;
        Calculate Output of Controller;
        Calculate Fitness Function;
      End For
    Crossover;
    Mutation;
    Local Search;
    Reproduce New Population;
  Until (t > N);
END
```

Fig. 8. Evolution Procedure.

## 3. EXPERIMENTS AND RESULTS

In order to demonstrate the effectiveness of the proposed genetic fuzzy controller in the real world, experiments have been conducted through three steps: first, programming the source code of the embedded genetic fuzzy controller and tuning the parameters on computer; second, simulating the embedded controller in the virtual environment on PC to check the performance; At last, applying the modified program on the Handy Board on the real robot to control its navigation. Here off-line supervised tuning is executed. The tuning input data are obtained from the four infrared sensors in real environment. Desired outputs in certain situations are derived from the expert's opinion. There are 100 groups of tuning data.

The comparison of GA with/without local search is shown in Fig. 9. It shows that local search techniques can improve the tuning process.

To prove the effectiveness of the evolved motion control model, several tests have been simulated

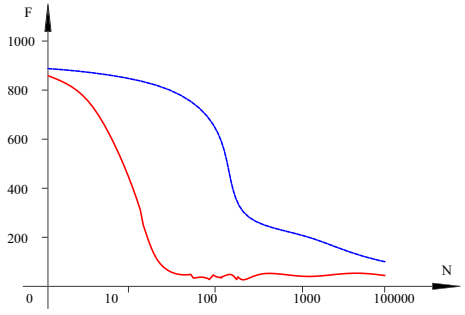


Fig. 9. Fitness changes during tuning iteration.

on computer. Here some approximations have been applied to simplify the simulation. First, the surface roughness of all obstacles and walls are 100%, thus, the value of infrared sensor is only related to the distance to obstacles no matter what reflection angle is. Second, the detection angle of infrared sensors is ignored.

In Fig. 10, the mobile robot is placed in a room ($10m \times 10m$) with some static obstacles. The mobile robot shows enough flexibility and intelligence to avoid all obstacles smoothly. In Fig. 11, the mobile robot 1, 2 and 3 are identical, but they take different actions to handle different situations. There is no change for Robot 1 during the navigation, thus it keeps going straightforward. Robot 2 detects an obstacle on the path, and it turns a certain angle to avoid the obstacle. When Robot 3 reaches Point A, a obstacle suddenly appears on the planning path of Robot 3. It makes a sharp turn and avoid the suddenly appeared obstacles. It shows the capability to handle the suddenly appeared changes in the environment.

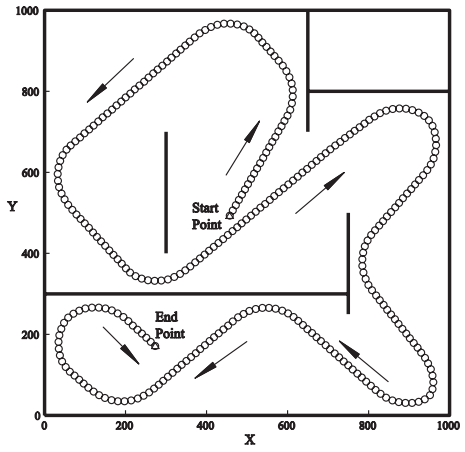


Fig. 10. Free navigation in an unknown environment.

At last the proposed genetic fuzzy controller is embedded in the real mobile robot. The mobile robot demonstrates satisfactory ability to avoid obstacles in the navigation autonomously. According to the distance between the robot and the obstacles, the robot took different obstacle-avoidance actions (go in straight, turning a little

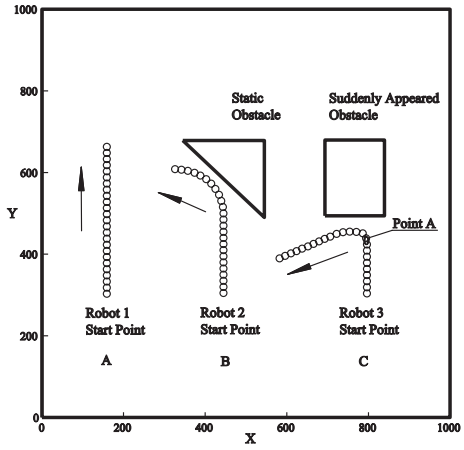


Fig. 11. Navigation in various situations. A: No obstacle; B: Static obstacle; C: Obstacle suddenly appears when robot reaches Point A.

bit to avoid the obstacle, or make a sharp turn. It can navigate autonomously in a room while avoiding bumping into any obstacles.

## 4. CONCLUSION

In this paper, a new embedded genetic fuzzy controller for a mobile robot is developed and tested successfully. Compared to the fuzzy logic controllers, it provides a systematic learning method to obtain suitable membership functions through an automatic learning process, so that the time-consuming trial and error procedure could be avoided. The proposed controller is applied to a mobile robot built in the ARIS Lab and the effectiveness of the proposed method is demonstrated through experimental studies.

## REFERENCES

Akishita, S., T. Hisanobu and S. Kawamura (1993). Fast path planning available for moving obstacle avoidance by use of laplace potential. In: *Proc. of IEEE/RSJ Intl. Conf. on Intelligent Robots and Systems*. pp. 673–678.

Borenstein, J. and Y. Koren (1989). Real-time obstacle avoidance for fast mobile robots. *IEEE Trans. on Systems, Man and Cybernetics* **19**(5), 1179–1187.

Borenstein, J. and Y. Koren (1991). Histogramic in-motion mapping for mobile robot obstacle avoidance. *IEEE Trans. on Robotics and Automation* **7**(4), 535–539.

Godjevac, J. (1995). A learning procedure for a fuzzy system: application to obstacle avoidance. In: *Proc. of Intl. Symp. on Fuzzy Logic*. Zurich. pp. 142–148.

Zhang, M. and S. Peng (1998). New approach for mobile robot obstacle avoidance based on multi-sonar information. *Automatica Sinica* **24**(5), 671–674.