Behavior-based Learning Fuzzy Rules for Mobile Robots

Siripun Thongchai*

Abstract

This paper describes a method to build a learning system for mobile robots based on fuzzy control technique. Range sensors are analyzed and used. The learning system generates a set of fuzzy rules based on the robot's behaviors. Two examples illustrate the learning capabilities: 1) learn-to-avoid-obstacle and 2) learn-to-detect-landmark behaviors. Reinforcement learning is applied to correct the fuzzy rules. Fuzzy controllers are used to control the robot using a set of fuzzy rules. The results of learn-to-avoid behavior indicate a good learning performance in dynamic environments. The results of learn-to-detect-landmark behavior show the robot can detect landmarks correctly from 68.43% to 90.09%.

Keywords : Learning fuzzy rules, fuzzy control, mobile robot, reinforcement learning, and behavior-based.

1. Introduction

An intelligent system is needed for the most autonomous mobile robots to navigate themselves around the real environment without any collisions and reach their target point as well. The learning technique is one way that most autonomous mobile robots are used. It may be very difficult in real experiments, because most learning techniques require a number of trials. The methods are different and are based on the physical characteristics of the robots, the various environments, and the software systems. Matarić [4] presents a method for mobile robot navigation and learning based on active sensing using sonar and a compass. Several behaviors were designed to extract the environment features from the sensory data. A distributed graph, as described in Matarić [5], consists of a collection of the robot's behaviors which are represented as the nodes. Matarić and Brooks describe an implemented mobile robot which successfully builds maps and uses them to navigate [2, 6]. Rosenstein and Cohen [9] describe a way for a mobile robot to learn by itself through the various environments. Clustering techniques are used for categorizing data such as objects, walls, etc. These techniques are similar to Ram and Santamaría's work which recorded sensory inputs and control outputs as a time series using a case-based reasoning approach to control a mobile robot for a navigation task [8].

In this work, a learning system for a mobile robot is designed based on fuzzy control technique. Laser and compass readings are used as the fuzzy inputs. The learning system generates a set of fuzzy rules for building behaviors of mobile robots. Two examples, avoid-obstacle and detect-landmark behaviors are illustrated. Each set of fuzzy rules is produced based on the sensory information and motor action which are related to the robot's behaviors. The results are used for mobile robot navigation and tested using an ATRV-Jr mobile robot [3].

* Department of Teacher Training in Electrical Engineering, Faculty of Technical Education, King Mongkut's Institute of Technology North Bangkok.

2. Learning Fuzzy Rules

Fuzzy control consists of fuzzification, fuzzy rules, and defuzzification processes. In the real-world, the information can be classified into two kinds: numerical information obtained from sensor measurements, and linguistic information obtained from human experts [10]. Wang and Mendel [10] developed a method to generate fuzzy rules from numerical data. It is very interesting to note that this work has better performance than a neural network. Fuzzy rules are expressed in the form of IF-THEN rules, providing the necessary connection between input and output fuzzy sets. A set of fuzzy rules needs to generate properly in a control manner that relates to perception and action of a system such as mobile robots.

2.1 Generate a Set of Fuzzy Rules

A set of fuzzy rules can be generated based on the robot's learning experiences and the conditions that it needs to react to. Whenever the robot is moved, the sensory-motor readings are fuzzified and mapped using IF-THEN rules. If no rule reports a match, the current rule will be added to the rules set, otherwise the matching rule will be counted and the highest degree will be recorded.

The rules generated in this method are "AND" rules which are used in the IF part. From the IF-THEN, the IF part must meet the conditions in order for the result of the THEN part to occur. This learning algorithm can be described as follows

$$r^i(t) = \text{If } m^i_{A_l}(t) \text{ and } m^i_{B_m}(t) \text{ THEN } m^i_{C_n}(t)$$
$$r^i(t) = \text{c}\,(m^i_{A_l}(t)\ m^i_{B_m}(t)\ m^i_{C_n}(t))$$

where $r^i(t)$ is the rule number *i*, *c* is a positive value called *learning constant*, and $m^i_{A_l}, m^i_{B_m}, m^i_{C_n}$ are the membership functions from inputs A, B and output *C*. There is a high probability that some rules can occur many times but with different degrees $(r^i_d(t))$. Thus, for this conflict only, the rule that has maximum degree $(r^i_d(t))$ will be selected and the number of occurrences will be recorded. This will solve the conflict problem and the number of rules are reduced as well [10]. The maximum degree $(r^i_d(t))$ will also be multiplied with the number of occurrences for this rule and will result as the output gain of this rule. This is similar to the output weights in neural network.

For avoid-obstacle learning, the reinforcement learning is combined to evaluate the situations. A negative reinforcement signal is given if the robot approaches an obstacle within less than the emergency distance. Since the obstacle distance is fuzzified, the emergency distance will be defined as a membership function. Thus, the reinforcement learning gain for each rule will be calculated as follows:

$$r^i_c(t) = r^i_c(t) + r^i_c(t-1) \quad (1)$$

$$r^i_c(t) = \begin{cases} 1 & \text{success} \\ -1 & \text{failure} \end{cases} \quad (2)$$

As a reinforcement technique, learning is based on the following idea: if the robot fails to move away from an obstacle, the reinforcement learning will give a negative action result which can be replaced as a negative number (-1). Otherwise the number $r^i_c(t)$ is increased as a positive number (1) based on the mapping between input and output results which indicate the confidence values for each rule.

However, the reinforcement learning for learn-to-detect-landmark slightly differs from learn-to-avoid-obstacle behavior. While the robot is learning to detect the landmark, if the fuzzy rule is the same as the previous fuzzy rule (which was generated in the same landmark), the reinforcement learning will give a positive value of one and add it into the rule as the gain. On the other hand, if the fuzzy rule is the same as the previous fuzzy rule (which was generated

in a different landmark), the reinforcement learning will give a negative value of one which will delete this fuzzy rule from a set of fuzzy rules. This will solve the conflict problem of identical fuzzy rules.

2.2 Integrate a Set of Fuzzy Rules

A set of fuzzy rules will be integrated into the robot control system after the learning phase. The fuzzy rules consist of a set of conditions compared with the current sensory readings. These fuzzy rules will map sensory space to control space in the following form:

IF *(a set of conditions is fulfilled) THEN (a set of outputs can be determined)*

Each fuzzy rule of learn-to-avoid-obstacle behavior gives two outputs; linear velocity and angular velocity. The antecedents of two outputs for all matching rules are combined using product operation. An output such as linear velocity[1] can be determined using the following centroid defuzzification:

$$y_v = \frac{\sum_{i=1}^{k} y_v^i c_v^i r_c^i r_d^i}{\sum_{i=1}^{k} y_v^i} \qquad (3)$$

$$y_v^i = (M_A^l[m_{A_i}^i(t)] M_B^m[m_{B_m}^i(t)] M_C^n[m_{C_n}^i(t)])$$

where K is the total number of control rules, c_v^i denotes the center value of y_v^i which is the output region of rule i, M_A^l is the current degree of membership function l for sensory reading input A at rule i, r_d^i is maximum degree for rule i, and $r_c^i(t)$ is reinforcement learning. The membership functions M_A^l are Gaussians and the fuzzy conjunction is implemented as the product method.

If the total number of control rules (K) and the output gain $(r_c^i r_d^i)$ have a large difference, the output of the controller may also have a large change. Therefore, the actual output of the controller can be obtained by adding the previous values [7]:

$$A(t) = A(t-1) + \sum_{i=1}^{k} (y_v^i c_v^i r_c^i r_d^i),\ A(0) = 0$$

$$B(t) = B(t-1) + \sum_{i=1}^{k} (y_v^i),\ B(0) = 0$$

$$y_v(t) = \frac{A(t)}{B(t)} \qquad (4)$$

The fuzzy output of learn-to-detect-landmark is the number of landmarks. While the robot is moving, a match fuzzy rule gives the output as the number which is represented as the landmark.

3. Learn to Avoid Obstacle

3.1 Map Sensory-Motor Readings

The laser sensor of ATRV-Jr mobile robot will be used forlearn-to-avoid-obstacle behavior. There are three important sensor features for avoid obstacle that can be extracted from laser readings: *direction of obstacle force from all obstacle distances* (θ_{force}^{ob}), *closest obstacle distance* $(d_{\min}^{ob})$, *and direction of the closest obstacle* $(\theta_{\min}^{ob})$. The direction of obstacle force is an excellent parameter for learn-to-avoid-obstacle, because it is the summation of all

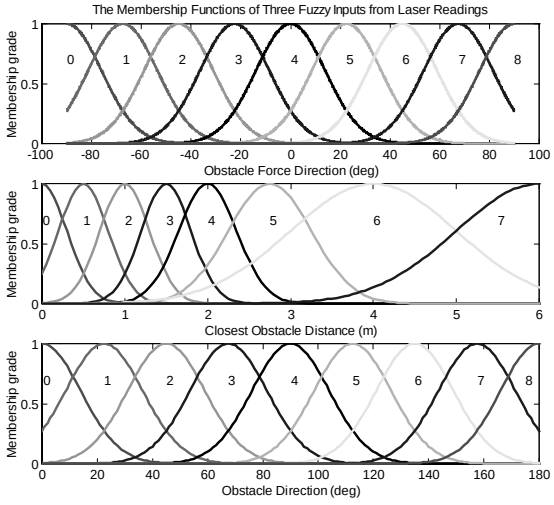

Figure 1 The membership functions of three fuzzy inputs for learn-to-avoid-obstacle using laser readings.

[1] The angular velocity has the same calculation method as the linear velocity.

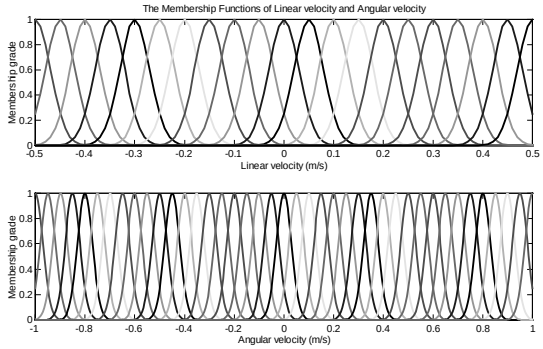

Figure 2 The membership functions of linear velocity and angular velocity.

forces which relates to all obstacle directions and distances. The result gives the direction that the robot should go and relates to the obstacles around the robot. This parameter is similar to the parameter that is used to design avoid obstacle behavior as used in [1]. The closest obstacle distance and the direction of the closest obstacle relate to the direction that the robot should not go. All these laser readings are fuzzified as Figure 1 show. The motor actions, including linear velocity and angular velocity, are fuzzified as illustrated in Figure 2.

3.2 Learning Fuzzy Rules

By using the learning methods described previously, the robot is able to generate a set of fuzzy rules for avoid obstacles. The number of fuzzy rules may differ depending on the robot's experience and time to learn. If the robot learns to generate the fuzzy rules in a complex environment, such as many static and dynamic obstacles around the robot, the fuzzy rule may be increased suddenly. However, the number $r_c^i(t)$ may be reduced for each rule at the same time. For better learning, the robot should learn in more diverse environments and time. An example of a set of fuzzy rules is shown in Table 1.

Table 1 Example of fuzzy rules for avoid-obstacle behavior

Rules	θ_{force}^{ob}	$d_{\min}^{ob}$	$\theta_{\min}^{ob}$	v	ω	$r_c^i(t)$	$r_d^i(t)$
1	6	1	3	21	22	1	0.13
2	5	2	7	22	20	1	0.30
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮
n	4	1	3	21	27	3	0.32

3.3 Fuzzy Controller

The set of fuzzy rules is given to the robot after learning is successful. In this step, the current sensory readings are fuzzified and search for matching rules. All matching rules are defuzzified and multiplied with the gain $r_c^i(t)$. This defuzzification method is given by equation (3). However, the current output, such as linear velocity, may be obtained by equation (4). Since the robot learns to move around the environment and avoid obstacles, the defuzzification results will enable the robot to act in avoid-obstacle behavior.

3.4 Adaptive Fuzzy Rules

In the real application, it may not easy for the robot to learn and generate a set of fuzzy rules for all environments. Adaptive fuzzy rules will generate a new rule that has never happened before for the robot, while the robot is moving around the new environment. This allows the robot to adapt itself from time to time as an intelligent system. Avoid-obstacle behavior is the first priority that the robot needs to do. An algorithm for adaptive fuzzy rules is given as follows.

Algorithm 1 *Adaptive fuzzy rules for avoid-obstacle behavior*

IF obstacle is not close **THEN** using a set of matching rule[2]

IF no matching rules **THEN** move forward and learn a new rule[3]

[2] Using learning set of fuzzy rule

[3] Add new fuzzy rule

ELSE IF obstacle is close **THEN** delete the matching rule and move-back and learning a new rule[4]

IF no matching rules **THEN** move-back and learning a new rule[5]

END IF

A set of fuzzy rules can be changed constantly as long as the robot learns in a new environment or fails to use the rules from previous learning experiences. Thus, the robot will adapt itself automatically and act more intelligently.

4. Learn-To-Detect-Landmark

4.1 Map Sensory Readings and Landmarks

The combination between laser and compass readings should provide some features for the robot to learn to detect landmarks. Compass reading provides the robot's heading while the laser gives the objects' location related to the current robot's heading. The laser will be used for learn-to-detect-landmark behavior as follows: *average obstacle distance* (d_{avg}^{ob}), *closest obstacle distance* ($d_{\min}^{ob}$) *and direction of the closest obstacle* ($\theta_{\min}^{ob}$). The average obstacle distance is calculated from all the detected obstacles which relate to the robot's position. The distance and the direction of the closest obstacle give important features that the robot can recognize as a landmark for each location. Landmarks are defined as fuzzy outputs using a sequence of number (1, 2, ...n). The fuzzification of the average obstacle distance, the closest obstacle distance, and the direction of the closest obstacle results are given in Figure 3. The compass is fuzzified into nine membership functions as Figure 4 shows.

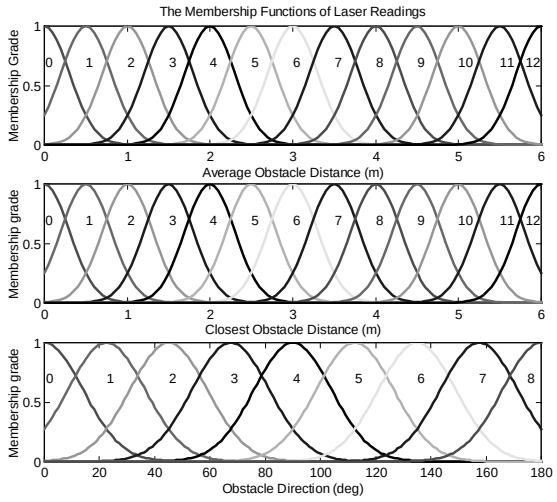

Figure 3 The membership functions of three fuzzy inputs for learn-to-detect-landmark using laser reading.

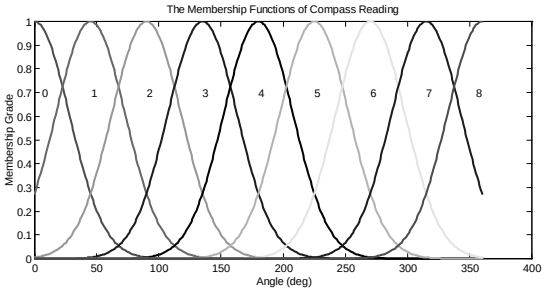

Figure 4 The membership functions of compass reading.

4.2 Fuzzy Mapping

The set of fuzzy rules from learn-to-detect-landmark is given to the robot after learning is successful. This is the same as avoid-obstacle behavior where the current sensory readings are fuzzified and searched for matching rules. All matching rules are defuzzified and multiplied with the gain $r_c^i(t)$. If the rules are matched with the current status, the current landmark is obtained and the confidence value is calculated based on the matching rules and the gain. If the fuzzy rules are identical with different landmarks, then two or more landmarks can be obtained. However, only the highest gain will be selected for each landmark. This will also provide a less confidence value than rules having a unique landmark.

[4] Delete failed fuzzy rules and add new fuzzy rule

[5] Add new fuzzy rule

5. Experimental Results

5.1 Learn-To-Avoid-Obstacle

To illustrate the performance of learning capabilities in each situation, two experiments are conducted as the following:

5.1.1 Experiment 1 : Learning in Static Environment: No objects are moving and the robot is commanded to move forward with avoid-obstacle behavior. While the robot moves around the various environments, a set of fuzzy rules is generated. The number of fuzzy rules is increased from time to time based on the environment. When the robot passes the same points, the same previous fuzzy rules will be generated, but will not be counted as new fuzzy rules. The reinforcement gain of matching fuzzy rules is increased as one each time. During this experiment, 3,547 fuzzy rules are generated in 35 minutes. Since some fuzzy rules are the same, the total fuzzy rules are reduced to 957 rules. The histogram of fuzzy rules for learn-to-avoid-obstacle behavior is shown in Figure 5.

5.1.2 Experiment 2 : Learning in Dynamic Environment: The environment is setup in the same location as Experiment 1, but some objects change locations from time to time. The robot is commanded to move forward with avoid-obstacle behavior, the same as in Experiment 1. While therobot moves around the different environments, many objects change their location and the human moves around the robot. Thus, new fuzzy rules are generated and the number of fuzzy rules is increased from time to time based on the environment. During this experiment, 4,148 fuzzy rules are generated in 41 minutes and reduced to 1,875 rules.

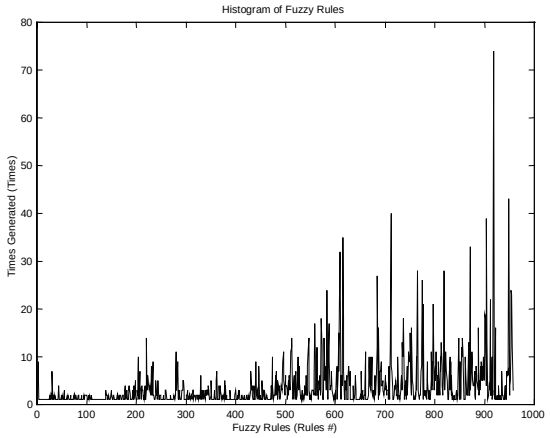

Figure 5 Histogram of fuzzy rules for learn-to-avoid-obstacle in a static environment.

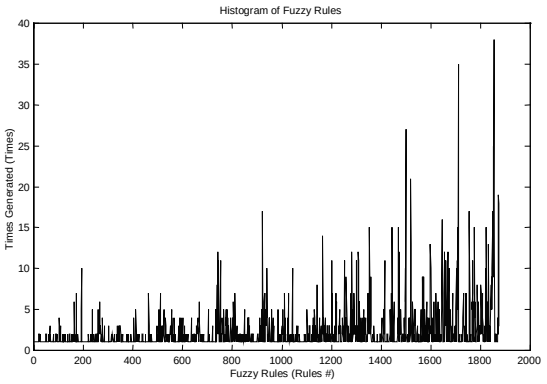

Figure 6 Histogram of fuzzy rules for learn-to-avoid-obstacle in dynamic environment.

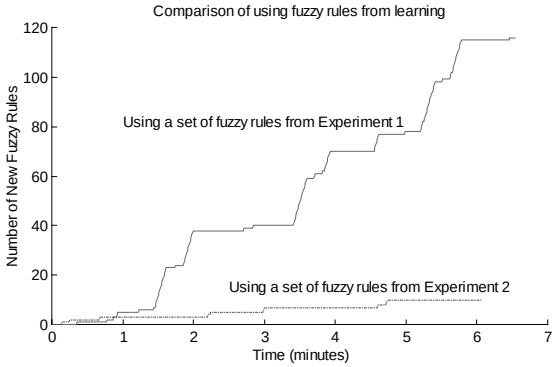

Figure 7 Comparison of the number of times the robot gets stuck by using fuzzy rules from experiment 1 and experiment 2.

5.2 Avoid-Obstacle

By using a set of fuzzy rules from the first experiment, the results show that the robot can move around the environment and avoid obstacles. However, there are many situations when the robot may get stuck, thus, new fuzzy rules are generated. By using a set of fuzzy rules from the second

experiment, the robot moves better. The number of times that robot gets stuck is less than using a set of fuzzy rules from the first experiment. Each set of fuzzy rules is tested for 6 minutes and new fuzzy rules are generated as Figure 7 shows. The robot gets stuck 10 times while using a set of fuzzy rules from the second experiment and it gets stuck 116 times while using a set of fuzzy rules from the first experiment. These results are summarized in Table 2.

Table 2 Results of using fuzzy rules from experiment 1 and experiment 2

Experiment	Time (min)	New Fuzzy Rules
1	6:53	116
2	6:05	10

Table 3 Results of fuzzy rules from learn to detect landmarks

Landmark No. ⇒	1	2	3	5	6
Total Rules	92	56	73	67	71
Max Time Gererated	10	15	8	18	9

5.3 Learn to Detect Landmark

In this experiment, the robot learns to detect 5 landmarks from the environment in the lab environment experiment. The robot learns to recognize each landmark in approximately 100 seconds. A set of fuzzy rules is generated from the robot's learning. Table 3 shows the number of fuzzy rules from each landmark.

After learning, a set of fuzzy rules is generated and the total number of fuzzy rules is 359 rules. The histogram of fuzzy rules for learn-to-detect-landmark behavior is given in Figure 8. The maximum number of times generated is 18.

5.4 Detect Landmark

A set of fuzzy rules from learn-to-detect-landmark behavior is used for mapping fuzzy inputs.

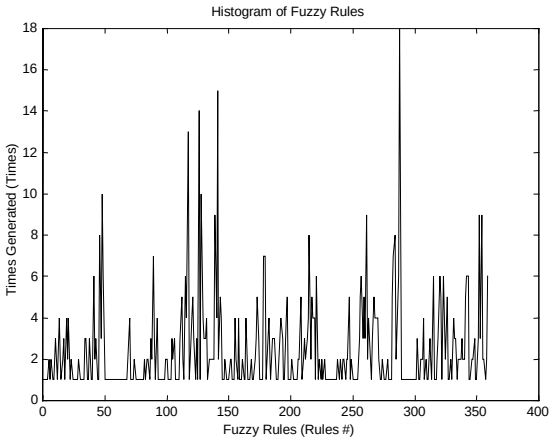

Figure 8 Histogram of fuzzy rules for learn-to-detect-landmark.

Table 4 The performance of detect-landmark behavior using a set of fuzzy rules from learn-to-detect-landmark behavior

True → / Recognized as ↓	1	2	3	4	5
1	68.43	11.63	0	0	0
2	25.10	80.19	0	0	0
3	0	0	90.08	11.05	0
4	0	0	4.61	87.83	0
5	0	0	0	0	90.09
% N/A	6.45	8.17	5.30	1.10	9.90
% accuracy	68.43	80.19	90.08	87.83	90.09

Landmark numbers are the results of mapping. If there is no fuzzy rule matching the current sensory information from compass and laser readings, the robot will give -1 which means there is no landmark detected. The performance accuracy is calculated as the percentage of correct detected landmarks while the robot is moving to that location. Table 4 shows the performance of the fuzzy detect landmarks.

5.5 Discussion

During the learning phase, learn-to-avoid-obstacle behavior performs better in a dynamic environment where the objects are moving and obstacles are

removed or introduced. The number of fuzzy rules from learning in the dynamic environment is higher than learning in a static environment. It is approximately 50% better as Table 5 shows. Therefore, the robot will move better while it is using a set of fuzzy rules from the learn-to-avoid-obstacle in a dynamic environment. This can be verified by the number of times the robot gets stuck. While the robot is using a set of fuzzy rules from learn-to-avoid-obstacle in the dynamic environment, the number of times the robot gets stuck is less than using a set of fuzzy rules from learn-to-avoid-obstacle in the static environment. However, both sets of fuzzy rules can be used to control the robot and the results show that the robot can move safely around the environment. Additionally, two sets of fuzzy rules from these experiments can be combined and used to control the robot, therefore, the number of times that the robot gets stuck should occur less often when using these two results.

Table 5 Comparison of two experiments from learn-to-avoid obstacle

Experiment	1	2
Environment	static	Dynamic
Time (min)	35.51	41.54
Fuzzy Rules	957	1875
Max Times Activated	74	38

By using the same method as learn-to-avoid-obstacle and adding compass readings as one of the fuzzy inputs, the robot is able to learn to detect a landmark called learn-to-detect-landmark behavior. The landmarks in this context are any object in a specific location which can be detected by the laser sensor. The robot will distinguish each landmark by three different features from the laser reading which includes the average of obstacle distance around the robot, the closest obstacle direction, and the closest obstacle distance. These features are combined with a compass reading which gives the robot heading. Table 4 shows the results of this detect-landmark behavior. The results show that the robot can detect landmarks correctly from 68.43% to 90.09%. The chance that the first landmark is identical to the second landmark is 25%. Therefore, the robot can detect the first landmark only 68.43%. As with other landmark detection methods, it is not easy to identify the identical landmarks.

6. Conclusion

The learning fuzzy rules are designed for a mobile robot. Two examples demonstrate that the robot can learn from its experiences using our two approaches learn-to-avoid-obstacle and learn-to-detect-landmark behaviors. The first two experiments of learn-to-avoid-obstacle are implemented in different indoor environments. The results show that the robot has a better learning rate in the dynamic environment. By using a set of fuzzy rules from learn-to-avoid-obstacle behavior in dynamic environment, it yields better results when comparing learn-to-avoid-obstacle behavior in a static environment. The last experiment result illustrates the capability of learn-to-detect-landmark behavior using the laser and compass readings as the fuzzy inputs. Results show that the accuracy of detection ranges between 68% to 90%. The overall results show that the average is 83.32% accuracy.

References

1. Borenstein, J. and Koren, Y. “Real-time obstacle avoidance for fast mobile robots.” *IEEE Transactions on Systems, Man, and Cybernetic.* 19, 5 (September/October 1989) : 1179-1187.
2. Brooks, R. A. *Cambrian Intelligence : The Early History of the New AI.* Cambridge, Massachusetts : The MIT Press, 1999.
3. Kawamura, K., et al. “Supervisory control of mobile robots using sensory EgoSphere.” In *IEEE International Symposium on Computational*

Intelligence in Robotics and Automation, Banff, Canada, July 29 - August 1, 2001.

4. Matarić, M.J. "Qualitative sonar based environment learning for mobile robots." In *Proceedings of SPIE 1989 Mobile Robots IV,* 305-314, November 1989.
5. Matarić, M.J. "Environment learning using a distributed representation." In Proceedings of the *IEEE International Conference on Robotics and Automation*, 402-406, May 1990.
6. Matarić, M.J. and Brooks, R. A. "Learning a distributed map representation based on navigation behaviors." In *Proceedings of the 1990 USA-Japan Symposium on Flexible Automation,* 499-506, Kyoto, Japan, June 1990.
7. Mendel, J.M. *Uncertain Rule-Based Fuzzy Logic Systems : Introduction and New Directions.* Upper Saddle River, New Jersey : Prentice-Hall, 2001.
8. Ram, A. and Santamaria, J. C. "Continuous casc-based reasoning." *Artificial Intelligenc*. 90, 1-2 (February 1997) : 25-77.
9. Rosenstein, M.T. and Cohen, P.R. "Continuous categories for a mobile robot." In *Proceedings of the Sixteenth National Conference on Artificial Intelligence*, 634-640, 1999.
10. Wang, L.X. and Mendel, J.M. "Generating fuzzy rules by learning from Examples." *IEEE Transactions on Systems, Man, and Cybernatics,* 22, 61 (November/December 1992) : 1414-1427.