2009 Ninth International Conference on Intelligent Systems Design and Applications

Design Based on a Shared Lookup-Table for an Obstacle Avoidance Fuzzy Controller for Mobile Robots

Jinwook Kim, Yoon-Gu Kim, Young-Duk Kim, Won-Seok Kang, Jinung An
Pragmatic Applied Robot Institute (PARI)
Daegu Gyeongbuk Institute of Science & Technology (DGIST)
Daegu, Republic of Korea
robot@dgist.ac.kr

***Abstract*—Fuzzy algorithms provide intuitive method for robot obstacle avoidance. Fuzzy controllers incorporating a design based on lookup tables (LUT) enable faster obstacle avoidance in environment with multiple obstacles. In an earlier study, we introduced a full LUT-based architecture for an 18-rule Positive/Negative (P/N) fuzzy controller. In this study, the number of fuzzy rules is expanded to 50. Because of the extra rules, the controller apparently needs more LUT(s) buffers. In other words, the buffer size increases with the complexity of the fuzzy controller. Therefore, we propose a LUT sharing method to reduce the buffer size without significantly degrading the performance of the controller. The final objective of this work is to design a LUT-based fuzzy controller whose buffer size is independent of the complexity of the fuzzy system. The proposed method is evaluated by simulating a 50-rule P/N fuzzy controller using Microsoft Robotics Developer Studio (MSRDS). The simulation results show that in comparison with the method not using LUT(s), full LUT-based method and the LUT sharing method reduce the operational time by nearly 80% and 70%, respectively. Although the LUT sharing method needs 1.5 times more operational time than the full LUT methods, it reduces the buffer size by more than 90%.**

Keywords- robot navigation; obstacle avoidance; fuzzy controller; lookup table

I. INTRODUCTION

Obstacle avoidance algorithms for autonomous navigation aim to navigate a robot to a target position without human control. The robot acquires information about the obstacles from sensors and updates a navigation map. The robot's path on the map from the current position to the target position is determined by these obstacles. If the map does not change during the navigation, obstacle avoidance is not necessary because a well-defined path safely leads the robot to the target position. However, if unexpected obstacles suddenly appear on the path, the robot must avoid them to arrive at the target position without collisions. To avoid these obstacles and navigate to the target position in real time, the robot must rapidly determine its steering direction and velocity. This study investigates fast real-time obstacle avoidance for robot navigation and focuses on improvement of its performance and optimization of the implementation.

Several algorithms were proposed for robot obstacle avoidance, including potential field algorithms [1], vector field histogram algorithms [2], and fuzzy controller algorithms [3]. The fundamentals of fuzzy controller methods were presented by Wang [4], Palm [5], Passino [6], Aguirre [7] and others. As the results of these papers, fuzzy controller methods have an advantage in real-time applications because they provide simpler and more intuitive methods for obstacle avoidance. Lilly [8] presented a P/N fuzzy obstacle avoidance controller and provided more intuitive method for robot obstacle avoidance [8]; this method used negative fuzzy rules to eliminate redundant fuzzy rules.

This paper is based on the P/N rule fuzzy controller method proposed in [8] and is an extension of our previous work [9].

The rest of this paper is organized as follows. Section 2 introduces the P/N rule fuzzy controller method and discusses the design of a 50-rule P/N fuzzy controller. Section 3 proposes a lookup table (LUT) sharing method for the controller. Section 4 describes the hardware design of the controller, and Section 5 presents our simulation and comparison results. Finally, Section 6 summarizes our conclusions.

II. P/N RULE FUZZY CONTROLLER

The positive rules of traditional fuzzy systems only describe the operations to be carried out. In Positive/Negative (P/N) rule fuzzy system, negative fuzzy rules are introduced to prescribe actions to avoid rather than execute. The positive rules induce proper system outputs of the fuzzy controller as the same manner of traditional fuzzy system rules while negative-rules prevent the system producing improper fuzzy outputs. One of the applications of the P/N rule fuzzy system is obstacle avoidance for autonomous robot navigation. In this application, the positive rules guide the robot to target while the negative rules prevent collisions. The positive rules dominant when only a few obstacles are present, and the negative rules begin to have an effect only when obstacles appear in the robot navigation path. If the obstacles are far from the robot's current position, the negative rules rarely affect the system. When the distance between the obstacles and the robot decreases, the negative rules begin to dominate the fuzzy system.

A. Design of 50-Rule P/N Fuzzy System

The fuzzy controller proposed in this paper is based on

978-0-7695-3872-3/09 $26.00 © 2009 IEEE
DOI 10.1109/ISDA.2009.218

TABLE I
CHARACTERISTICS OF FUZZY MEMBERSHIP FUNCTIONS

		Target		Obstacle	
		Center (c)	Variation (σ)	Center (c)	Variation (σ)
Direction (0.5°)	HL	-60	18	-60	18
	SL	-30	18	-30	18
	S	0	18	0	18
	SR	30	18	30	18
	HR	60	18	60	18
Distance (mm)	Z	0	1500	300	150
	VN	2500	1500	550	150
	N	5000	1500	800	150
	F	7500	1500	1050	150
	VF	10000	1500	1300	150

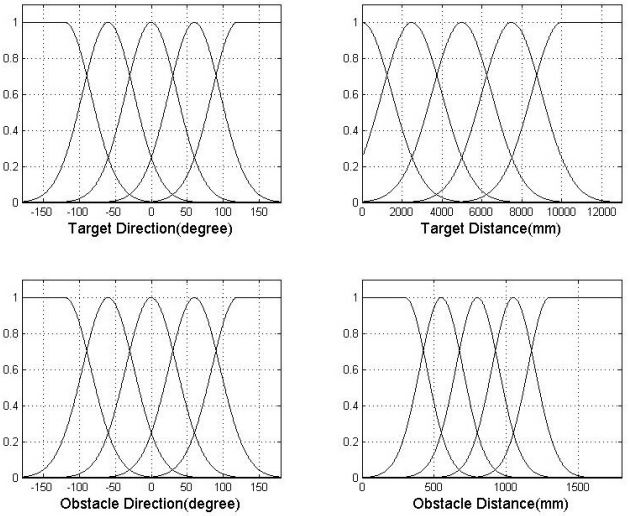

Figure 1. Membership functions of designed 50-rule fuzzy controller.

two types of inputs; laser range finder (LRF) sensor data and a relative target position. The LRF sensor provides a distance vector of directions over a 180° range with an angular resolution of 0.5°. Once the target position has been determined, the position is fixed on the navigation map. However, the relative target position should vary as the robot moves. The robot navigation controller must update the relative target information, which consists of the target distance and target direction. The fuzzy controller output is the steering direction for the robot. The controller can also convert the output into specific values such as velocities for the two differential wheels. Of the 50 P/N fuzzy rules, 25 are positive rules and relate to the target position, and 25 are negative rules and relate to obstacle positions.

B. Fuzzification

Fuzzification of the inputs is achieved by the membership functions shown in Figure 1. Each membership function of fuzzy input is Gaussian distribution function and is

TABLE II
RULE BASE FOR 25-RULE POSITIVE FUZZY SYSTEM (EXPERT-PROVIDED).

Steering direction		Target direction				
		HL	SL	S	SR	HR
Target distance	Z	HL	HL	S	HR	HR
	VN	HL	L	S	R	HR
	N	L	L	S	R	R
	F	L	SL	S	SR	R
	VF	SL	S	S	S	SR

TABLE III
RULE BASE FOR 25-RULE NEGATIVE FUZZY SYSTEM (EXPERT-PROVIDED).

Steering direction		Obstacle direction				
		HL_i	SL_i	S_i	SR_i	HR_i
Obstacle distance	Z_i	$\overline{HL}$	$\overline{HL}$	$\overline{S}$	$\overline{HR}$	$\overline{HR}$
	VN_i	$\overline{HL}$	$\overline{L}$	$\overline{S}$	$\overline{R}$	$\overline{HR}$
	N_i	$\overline{L}$	$\overline{L}$	$\overline{S}$	$\overline{R}$	$\overline{R}$
	F_i	$\overline{L}$	$\overline{SL}$	$\overline{S}$	$\overline{SR}$	$\overline{R}$
	VF_i	$\overline{SL}$	$\overline{S}$	$\overline{S}$	$\overline{S}$	$\overline{SR}$

TABLE IV
MEMBERSHIP DEGREES OF INPUT FUZZY SETS.

Degrees of positive-rule fuzzy sets		Degrees of negative-rule fuzzy sets	
HL	μ_{HL}	HL	$\mu_{\overline{HL}}$
SL	μ_{SL}	SL	$\mu_{\overline{SL}}$
S	μ_{S}	S	$\mu_{\overline{S}}$
SR	μ_{SR}	SR	$\mu_{\overline{SR}}$
HR	μ_{HR}	HR	$\mu_{\overline{HR}}$
Z	μ_{Z}	Z	$\mu_{\overline{Z}}$
VN	μ_{VN}	VN	$\mu_{\overline{VN}}$
N	μ_{N}	N	$\mu_{\overline{N}}$
F	μ_{F}	F	$\mu_{\overline{F}}$
VF	μ_{VF}	VF	$\mu_{\overline{VF}}$

expressed as

$$f(x) = e^{-\frac{(x-c)^2}{2\sigma^2}}$$

where c denotes the center position of the distribution and σ denotes the variation in the function. Table I presents the c and σ values for each fuzzy membership function. The fuzzy sets Hard Left (HL), Soft Left (SL), Straight (S), Soft Right (SR), and Hard Right (HR) relate to the target and obstacle directions. The fuzzy sets Zero (Z), Very Near (VN), Near (N), Far (Far), and Very Far (VF) relate target and obstacle distance. The output fuzzy sets Hard Left (HL), Left (L), Soft Left (SL), Straight (S), Soft Right (SR), Right (R), and Hard Right (HR) are for the positive rules, while $\overline{HL}$, $\overline{L}$, $\overline{SL}$, $\overline{S}$, $\overline{SR}$, $\overline{R}$, and $\overline{HR}$ are for the negative rules.

C. Rule Base

The expert-provided positive rule base is given in Table II; it can be represented by following rules.

Rule 1P
If $x_{TargetDirection}$ *is HL and* $x_{TargetDistance}$ *is Z then y is HL*

Rule 2P
If $x_{TargetDirection}$ *is HL and* $x_{TargetDistance}$ *is VN then y is HL*

.
.
.

Rule 25P
If $x_{TargetDirection}$ *is HR and* $x_{TargetDistance}$ *is VF then y is SR*

The expert-provided negative rule is given in Table III; it can be represented by the following rules.

Rule 1N
If $x_{ObstacleDirection}$ *is* HL_i *and* $x_{ObstacleDistance}$ *is* Z_i *then y is not HL*

Rule 2N
If $x_{ObstacleDirection}$ *is* HL_i *and* $x_{ObstacleDistance}$ *is* VN_i *then y is not HL*

.
.
.

Rule 6N
If $x_{ObstacleDirection}$ *is* SL_i *and* $x_{ObstacleDistance}$ *is* Z_i *then y is not HL*

.
.
.

Rule 25N
If $x_{ObstacleDirection}$ *is* HR_i *and* $x_{ObstacleDistance}$ *is* VF_i *then y is not SR*

The notation of the membership degree of each fuzzy set is given in Table IV. The weight of the positive rule *Rule 1P* is calculated by multiplying μ_{HL} and μ_Z, and the weight of *Rule 2P* is calculated by multiplying μ_{HL} and μ_{VN}. Other weights are calculated in the same ways. The combined degree of the positive rule *Rule 1P* and the negative rules is calculated as

$$\mu_1 = (\alpha + \mu_{HL} \cdot \mu_Z) \prod_{i=1}^{\#ofObstacles} (1 - \mu_{HL_i} \cdot \mu_{Z_i})(1 - \mu_{HL_i} \cdot \mu_{VN_i})(1 - \mu_{SL_i} \cdot \mu_{Z_i})$$

The negative terms in this equation are the relevant terms from the positive output fuzzy set. Because the positive output fuzzy set is HL, the negative terms must be determined from the corresponding negative output fuzzy set, $\overline{HL}$. Therefore, the positive term is multiplied by the terms of *Rule 1N*, *Rule 2N and Rule 6N*. As a result, the negative terms decrease the degree of the positive term. Note that *i* is used to index each obstacle in a multiple-obstacle space. Because multiple obstacles constitute the input for a negative-rule fuzzy system, accumulated multiplication is applied.

There are two parameters in the equation. Parameter α is an offset value that maintains a minimum level of the positive term. If the positive term is near zero, negative terms cannot have any effect on the fuzzy output. We performed simulations and found that one optimal value of the offset is 0.5. Similarly, the other combined degrees are calculated as follows.

$$\mu_2 = (\alpha + \mu_{HL} \cdot \mu_{VN}) \prod_{i=1}^{\#ofObstacles} (1 - \mu_{HL_i} \cdot \mu_{Z_i})(1 - \mu_{HL_i} \cdot \mu_{VN_i})(1 - \mu_{SL_i} \cdot \mu_{Z_i})$$

$$\mu_3 = (\alpha + \mu_{HL} \cdot \mu_N) \times \prod_{i=1}^{\#ofObstacles} (1 - \mu_{HL_i} \cdot \mu_{N_i})(1 - \mu_{HL_i} \cdot \mu_{F_i})(1 - \mu_{SL_i} \cdot \mu_{VN_i})(1 - \mu_{SL_i} \cdot \mu_{N_i})$$

$$\mu_4 = (\alpha + \mu_{HL} \cdot \mu_F) \times \prod_{i=1}^{\#ofObstacles} (1 - \mu_{HL_i} \cdot \mu_{N_i})(1 - \mu_{HL_i} \cdot \mu_{F_i})(1 - \mu_{SL_i} \cdot \mu_{VN_i})(1 - \mu_{SL_i} \cdot \mu_{N_i})$$

$$\mu_5 = (\alpha + \mu_{HL} \cdot \mu_{VF}) \prod_{i=1}^{\#ofObstacles} (1 - \mu_{HL_i} \cdot \mu_{VF_i})(1 - \mu_{SL_i} \cdot \mu_{F_i})$$

$$\mu_6 = (\alpha + \mu_{SL} \cdot \mu_Z) \prod_{i=1}^{\#ofObstacles} (1 - \mu_{HL_i} \cdot \mu_{Z_i})(1 - \mu_{HL_i} \cdot \mu_{VN_i})(1 - \mu_{SL_i} \cdot \mu_{Z_i}) \cdot$$

.
.
.

$$\mu_{24} = (\alpha + \mu_{HR} \cdot \mu_F) \times \prod_{i=1}^{\#ofObstacles} \beta \cdot (1 - \mu_{\overline{SR}i} \cdot \mu_{\overline{VN}i})(1 - \mu_{\overline{SR}i} \cdot \mu_{\overline{N}i})(1 - \mu_{\overline{HR}i} \cdot \mu_{\overline{N}i})(1 - \mu_{\overline{HR}i} \cdot \mu_{\overline{F}i})$$

$$\mu_{25} = (\alpha + \mu_{HR} \cdot \mu_{VF}) \prod_{i=1}^{\#ofObstacles} (1 - \mu_{\overline{SR}i} \cdot \mu_{\overline{F}i})(1 - \mu_{\overline{HR}i} \cdot \mu_{\overline{VF}i})$$

D. Defuzzification

Finally, the output value of the 50-rule P/N fuzzy controller is calculated as a weighted average of the degrees of all the output fuzzy sets.

$$\begin{cases} y = \dfrac{w_{HL} \cdot HL + w_L \cdot L + w_{SL} \cdot SL + w_S \cdot S}{w_{HL} + w_L + w_{SL} + w_S} & (w_{HL} + w_L + w_{SL} \geq w_{HR} + w_R + w_{SR}) \\ y = \dfrac{w_{HR} \cdot HR + w_R \cdot R + w_{SR} \cdot SR + w_S \cdot S}{w_{HR} + w_R + w_{SR} + w_S} & (w_{HL} + w_L + w_{SL} < w_{HR} + w_R + w_{SR}) \end{cases}$$

The weights of the output fuzzy sets are calculated as follows.

$$w_{HL} = \mu_1 + \mu_2 + \mu_6$$
$$w_L = \mu_3 + \mu_4 + \mu_7 + \mu_8$$
$$w_{SL} = \mu_5 + \mu_9$$
$$w_S = \mu_{10} + \mu_{11} + \mu_{12} + \mu_{13} + \mu_{14} + \mu_{15} + \mu_{20}$$
$$w_{SR} = \mu_{19} + \mu_{25}$$
$$w_R = \mu_{17} + \mu_{18} + \mu_{23} + \mu_{24}$$
$$w_{HR} = \mu_{16} + \mu_{21} + \mu_{22}$$

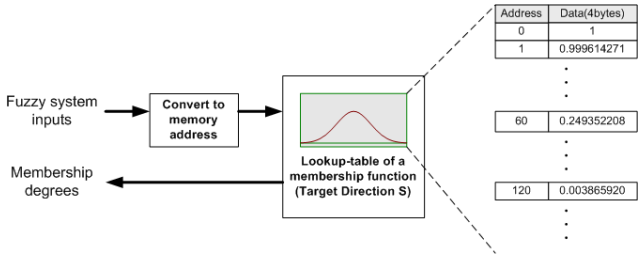

Figure 2. Diagram of fuzzy membership degree acquisition using lookup tables.

III. SHARED LUT-BASED FUZZY CONTROLLER

To enhance the performance of the obstacle avoidance controller, we aimed to reduce its processing time. Faster decisions about avoiding obstacles lead to more accurate avoidance. The fuzzy controller method is a simpler and more intuitive method for obstacle avoidance. However, fuzzy membership functions formed from Gaussian distributions increase the operational complexity because the controller must multiply numerous floating point numbers to calculate each fuzzy membership degree. Without high-performance processors, these operations could make real-time navigation difficult. Hence, we tried to reduce the computation time of the membership degrees by using LUT(s) for membership functions. Because all the input and output values for the membership functions are determined in advance, it is possible to search the membership degrees by using the LUT(s) without carrying out any computations. Figure 2 shows an example of obtaining the fuzzy membership degrees for a fuzzy set S using the LUT(s). In this architecture, each fuzzy input is converted into a corresponding buffer address and the data value at that address is the membership degree for the given input. If the fuzzy controller obtains each membership degree from the LUT(s) buffers, the robot can rapidly avoid unexpected obstacles because the calculation time associated with the multiplication of floating point operands is eliminated.

LUT-based fuzzy controller design is attractive because it allows rapid avoidance, but there are drawbacks. To store the LUT(s) of all the membership functions, it is necessary to ensure that the controller system has buffers of a sufficient size. Clearly, the more the number of rules in the controller, the larger is the buffer size. An excessively large buffer size may limit the application of general mobile embedded processors. To overcome this limitation, we tried to reduce the buffer size by sharing a LUT(s). The shared LUT contains only one basis function. The other fuzzy membership functions with a similar shape or characteristics obtain their outputs via a rescaling of this basis function. Because the membership functions in this paper are based on Gaussian distributions, the equation of the basis function can be expressed as

$$f_{basis} = e^{-\frac{x_{trans}^{\ 2}}{2\sigma_{BasisFunction}^{\ 2}}}$$

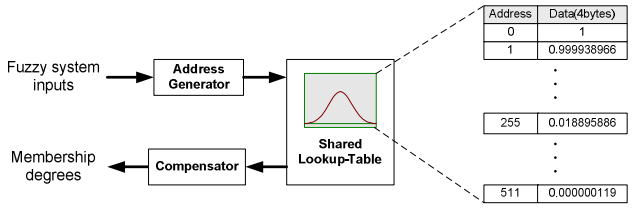

Figure 3. Diagram of fuzzy membership degree acquisition using the shared lookup table.

The variance of the basis function, $\sigma_{BasisFunction}$, is set to 128 for the 50-rule P/N fuzzy controller. Because a Gaussian distribution is characterized by its variance (σ) and center (c), the output value of each membership function is simply obtained from the basis function by rescaling the input value using the formula:

$$x_{Converted} = \frac{\sigma_{BasisFunction}}{\sigma_{MembershipFunction}}(x_{input} - c_{MembershipFunction})$$

The result, $x_{Converted}$, may not be an integer. However, the shared LUT requires integer input values. Since we cannot use non-integer indices, we obtain approximate values by rounding off $x_{Converted}$ to the nearest integer.

As noted above, f_{basis} has a Gaussian distribution, which is an even function, so the left part of the function in the shared LUT can be eliminated to remove redundancy of buffers. To index an address in the left part of the basis function, the absolute value of the rescaled address is used. The indexing address of the shared LUT is calculated as

$$Address_{LUT} = \left| round(\frac{\sigma_{BasisFunction}}{\sigma_{MembershipFunction}}(x_{input} - c_{MembershipFunction}) \right|$$

Figure 3 shows the fuzzy membership degree acquisition process for the LUT sharing method. The *Address Generator* generates the indexing address of the shared LUT, $Address_{LUT}$. In each buffer address, 32-bit floating point data is stored and the indexing address ranges from 0 to 511, as shown in Figure 3. If the calculated address, $Address_{LUT}$, exceeds 511, LUT returns 0 as the output. This means that we consider output values below 0.000000119 to be 0 in order to reduce the buffer size.

After reading the data from the shared LUT, the *Compensator* adjusts the data value because it is not exactly the same as the expected membership degree. However, the Compensator may not be necessary here because the errors between the shared LUT output and the original outputs are negligible. We have confirmed this by simulation experiments and have decided to remove the *Compensator* in our design.

IV. HARDWARE DESIGN OF THE FUZZY CONTROLLER

Figure 4 shows the hardware design for the proposed

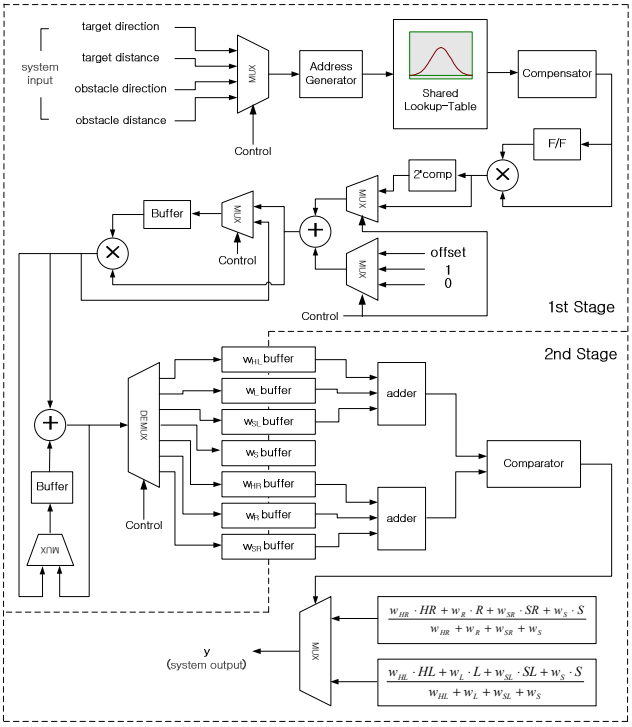

Figure 4. Hardware architecture of proposed P/N fuzzy controller.

obstacle avoidance fuzzy controller. To calculate the fuzzy operations, the membership degrees of the inputs, namely, target direction and distance and obstacle direction and distance, must first be calculated. The purpose of our design is to calculate all the membership degrees using the shared LUT. Although the architecture incorporates only one shared LUT, the hardware must immediately start the fuzzy operations and perform all these operations long before the input data arrive. Because the design permits only one set of input data at each cycle, adders and multipliers accumulatively calculate the operands, and buffers temporarily store the results of the calculations. The results of w_{HL} , w_L , w_{SL} w_S , w_{SR} , w_R , *and* w_{HR} are stored in buffers during the first stage. If the buffers for these values are full, the final output of the controller is calculated in the second stage; the first stage of the fuzzy controller is then able to process subsequent obstacle avoidance operations.

V. Simulation Results

We simulated the 50-rule P/N fuzzy controller using MSRDS [10] on a computer system with Intel® Core™ 2 Quad 2.4 GHz CPU and 3.24 GB RAM. If there are no obstacles, the robot moves in a straight line from the starting position to the target position. However, if there are many obstacles, the robot must change its path to avoid them. To evaluate our method using the simulator, we created a service module for each approach and used it in an MSRDS visual programming language (VPL) diagram. The service modules are programmed in C# programming language and compiled using Microsoft Visual Studio.

Because a graphic simulation environment requires high-performance processors, the simulation was performed on a high-performance computer system. We consider only the fuzzy operation time because the simulation system spends most of its time on graphic processing.

Table V shows the simulation results for each method. The buffer size for the full LUT design is (204+8469+847) × 4 bytes and that for the shared LUT is 512 × 4 bytes. The processing clock cycle for each method in the table is an average of the computation clock cycles for the avoidance operations during the navigation from the starting position to the target position. The controller without LUT(s) does not need any LUT(s) buffer space, but it requires eight times more time for performing fuzzy operations than the full LUT-based controller. If all the membership functions have LUT(s), the operation time is greatly reduced. However, many buffers are required. The simulation results show that by using the LUT sharing method, the controller can operate with just 5.34% of the buffer size of the full LUT(s). Although the shared LUT method is slower than the full LUT method, the enormous reduction in buffer size is a major advantage.

The simulation results show that the LUT-based controller reduces the computational requirements and it can obviously enhance the performance of real-time applications. Moreover, the LUT sharing method offers a way to use an LUT-based architecture without requiring a large-size buffer.

VI. Conclusions

This work aimed to design a real-time, low-power system for mobile robots. To perform fast operations in real-time systems, it is enough to use high-performance processors. However, real-time embedded systems have some limitations. We should aim to reduce the operation clock frequency so as to lower the power consumption because a high clock frequency elevates the power consumption of the systems. Thus, reduction in operational complexity is important from the viewpoint of achieving faster operations and lower power consumption.

We utilized an LUT(s) approach to avoid complicated and repetitive calculations of the membership degrees. Our full LUT-based fuzzy controller reduced the average operation time by 80%. However, this design requires buffers with large sizes to hold all the membership function values. To reduce the buffer size, we tried a shared LUT approach. The simulation shows that the LUT sharing method takes 1.5 times more operation cycles on average than the full LUT-based design, but it reduces the buffer requirements by 94.5%.

In conclusion, our work offers advantages with regard to real-time obstacle avoidance of mobile robots because the LUT-based fuzzy controller design enables faster obstacle avoidance and the LUT sharing method keeps the controller compact by greatly reducing the buffer size.

ACKNOWLEDGMENT

This work was supported by the Daegu Gyeongbuk Institute of Science and Technology (DGIST) and funded by the Ministry of Education, Science and Technology (MEST) of the Republic of Korea.

REFERENCES

[1] Koren Y, Borenstein J, "Potential field methods and their inherent limitations for mobile robot navigation," IEEE Int. Conf. Robotics and Automation, April 1991, pp. 1398 – 1404.

[2] Borenstein J, Koren Y, "The vector field histogram-fast obstacle avoidance for mobile robots," IEEE Int. Conf. Robotics and Automation, June 1991, pp. 278 – 288.

[3] Thongchai S, Kawamura K, "Application of fuzzy control to a sonarbased obstacle avoidance mobile robot," IEEE Int. Conf. Control Applications, September 2000, pp. 425 – 430.

[4] L.-X. Wang, Adaptive Fuzzy Systems and Control: Design and Stability Analysis, Pientice-Hall, Englewood Cliffs, New Jersey, 1994.

[5] R. Palm, D. Driankov, and H. Hellendoorn, "Model Based Fuzzy Control: Fuzzy Gain Schedulers and Sliding Mode Fuzzy Controllers," Springer-Verlag Berlin, USA, 1997.

[6] K. M. Passino and S. Yurkovich, "Fuzzy Control," Addison Wesley Longman, Menlo Park, CA, 1998.

[7] E. Aguirre, A. Gonzalez, "Fuzzy behaviors for mobile robot navigation", International Journal of Approximate Reasoning, 2000, pp. 255-289.

[8] John H.Lilly, "Evolution of a Negative-Rule Fuzzy Obstacle Avoidance Controller for an Autonomous Vehicle," IEEE Trans. Fuzzy Systems, Vol.15, No.4, August 2007, pp. 718 - 728.

[9] Jinwook Kim, Youngduk Kim, Wonseok Kang, Jinung An, "Design of a Lookup-table Based Positive/Negative Fuzzy Controller for Obstacle Avoidance of Autonomous Vehicle," 3rd ICUT, Dec 2008, pp. 648-652.

[10] http://msdn.microsoft.com/en-us/robotics/default.aspx.