FUZZY BEHAVIORS FOR CONTROL OF MOBILE ROBOTS

Saleh Zein-Sabatto, Ali Sekmen and Poolsak Koseeyaporn
Tennessee State University, Nashville, TN 37209, USA.
(mzein@tnstate.edu and *asekmen@tnstate.edu)*

Abstract

In this research work, an RWI B-14 robot has been used as the development platform to embody some basic behaviors that can be combined to build more complex robotics behaviors. Emergency, avoid-obstacle, left wall-following, right wall-following, and move-to-point behaviors have been designed and embodied as basic robot behaviors. The basic behaviors developed in this research are designed based on fuzzy control technique and are integrated and coordinated to from complex robotics system. More behaviors can be added into the system as needed. A robot task can be defined by the user and executed by the intelligent robot control system. Testing results showed that fuzzy behaviors made the robot move intelligently and adapt to changes in its environment.

Keywords: Intelligent behaviors, fuzzy logic techniques, mobile robot controls.

1. INTRODUCTION

Mobile robots are complex systems functioning in real world environments thus, it is difficult to design an adaptive control system that can control robots to act as desired. Hence, robot researches use theories and concepts from the intelligent control theory, described as *behavior-based* control in the robotic literature. *Behavior-based* approach has been established as the main alternative to conventional robot control in recent years. Behavior-based control applications become important for most mobile robots because real world cannot be accurately characterized or modeled [1]. In late 1985, Brooks [2] proposed his famous *subsumption architecture* that has been successfully applied in mobile robotics [3]. This approach makes it easy to design robots that pursue multiple goals, respond to multiple sensors and are incrementally extendable. The architecture is based on decomposing the problem of autonomous control by task rather than by function [2]. Intelligence in a mobile robot is considered as an *adaptive behavior* that makes a robot acts intelligently in its environment. Many events that can be attributed to adaptive behaviors have been observed and reported by many research groups, including *biologists, ethnologists,* and *philosophers* [4]. *Fuzzy control* provides a mechanism for incorporating human-like reasoning capabilities and computationally in control systems. The linguistic variables are used to mimic the human action into a system more closely than traditional control. Fuzzy logic is a logical system that aims at a formalization of approximate reasoning [5]. These can be represented as the concept of a linguistic variable, canonical form, fuzzy if-then rule, fuzzy quantifiers, modes of reasoning [6].

Fuzzy control is one of the intelligent control techniques that pertain to the realization of intelligent control systems. Fuzzy control is derived from the *fuzzy logic* and *fuzzy set* theory introduced in 1965 by L. A. Zadeh [7]. Fuzzy logic is a departure from the classical two-valued sets and logic that uses "*soft*" *linguistic* (e.g., large, hot, tall) system variables and a continuous range of truth-value in the interval [0,1]. Formally, fuzzy logic was a structured, model-free estimator that approximates a function through linguistic input/output associations. All behaviors (controllers) developed in this work are based on the fuzzy control techniques.

2. DEVELOPMENT OF BASIC BEHAVIORS

The architecture used to build and embody the robot behaviors consists of several fuzzy controllers. These behaviors include emergency, avoid-obstacles, left wall-followings, right wall-followings, and move-to-point. The behavior-based control architecture, as shown in Figure 1, organized vertically which shows that each behavior has full access to all sensor readings and processes its own commands to control the robot [8]. The outputs of each behavior are the linear velocity and angular velocity of the mobile robot. The fuzzy rules are defined based on the tasks. The final robot command is dependent on the fuzzy selection that integrates and coordinates all behaviors.

2.1. Emergency Behavior

The most fundamental behavior that all mobile robots should have is the emergency behavior. It should have the highest priority in taking the control of the robot. In some situations, avoid-obstacles behavior may not

`

function properly and the robot move too close to an obstacle. In such cases, an emergency behavior is required to stop the robot or even, in some instances, move the robot backward. These two sub-behaviors are combined together to from one behavior called emergency behavior. This behavior depends on the safest allowable distance between the robot and objects. The two distance values defined for the range between a robot and obstacle are, frontal distance (d_f) and side distance (d_s) as shown in Figure 2-a.

The minimum distance $S_{\min}(l)$ for the robot move-backward sub-behavior is calculated by equation (1)

$$S_{\min}(l) = \frac{d_s}{\cos(\theta_l)} \qquad (1)$$

For sensor numbers $l = 4,5,...,11$ and sensor angle calculated by equation (2)

$$\theta_l = 11.25 + 22.5(l-4) \qquad (2)$$

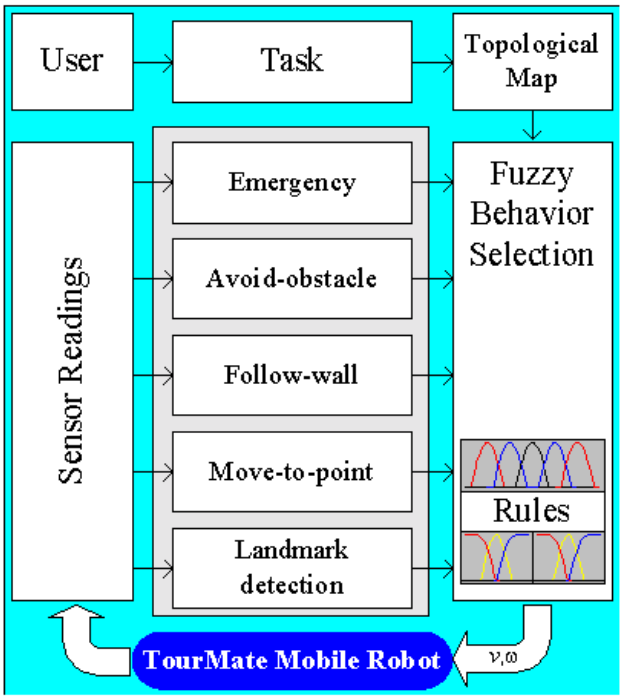

Figure 1. Intelligence Embodiment Architecture.

2.2. Avoid-obstacles Behavior

A basic need for all autonomous mobile robots is an obstacle avoidance behavior. This behavior helps mobile robots move freely without colliding in unstructured environments. In this work, avoid-obstacle behavior is considered as a basic behavior that uses four (4) sonar sensors on the front-left and four (4) sonar sensors on the front-right. The fuzzy controller for the avoid-obstacle behavior is designed based on two fuzzy inputs receiving information from two symmetric sonar sensors one from the left and one from the right. Thus, four fuzzy controllers are used for the development of the avoid-obstacle behavior. The sensor notations are shown again in Figure 2-b. The process of developing this behavior is described in the following section.

2.2.1. Input Fuzzification: We designed four fuzzy controllers using sonar readings. The sonar readings provided by the eight sonar sensors on the front-left and the front-right of the robot ($l = 4,5,...,11$). Each fuzzy controller has two inputs and receives reading from a pair of sensors. Figure 2-b also shows an example of one of these fuzzy controllers. Each sensor sends data directly to the controller input.

2.2.2. Membership Functions for the Input: for this behavior we construct three membership functions for each input on its universe. These are; ZR= zero distance (300 mm), MD= middle distance (1300 mm) and LG= large distance (2300 mm). Each membership function is constructed as a Gaussian curve. The Gaussian curve depends on two parameters σ_i and c_i as given by equation (3).

$$\mu_{F_i}(x_l) = \exp\left[-\left(\frac{(x_l - c_i)}{2\sigma_i}\right)^2\right], \qquad (3)$$

where: x_l is a sonar reading for $l = 4,5,...,11$ and $i = 0,1,2$ for three membership functions.

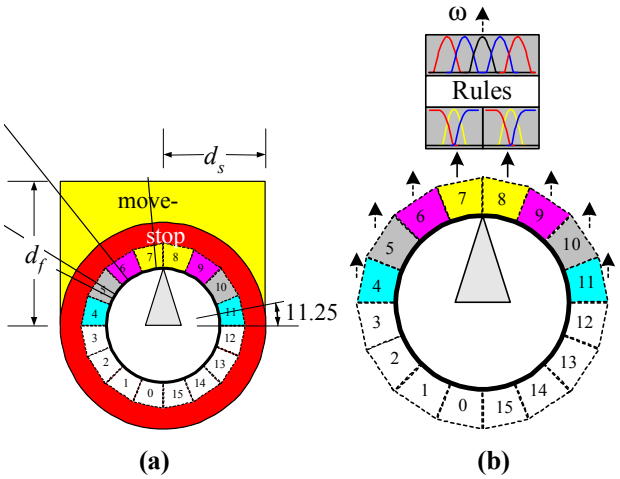

Figure 2. Fuzzy Controller for a) Emergency and b) Avoid-Obstacles Behavior.

2.2.3. Membership Functions for the Outputs: There are two outputs for each fuzzy controller, angular

`

velocity and linear velocity. From our experiments, we constructed five membership functions for each output. The angular velocity memberships are; PB= positive-big (2 rad/sec), P=positive (1 rad/sec), ZR=zero (0), N=negative (-1 rad/sec) and NB=negative-big (-2 rad/sec). The membership functions for the linear velocity are; PB= positive-big (40 mm/sec), P=positive (30 mm/sec), PM=positive-medium (20 mm/sec), PS=positive-small (10 mm/sec), ZR=zero (0) and N=negative (-10 mm/sec).

2.2.4 Fuzzy Rules: Nine rules are developed for each fuzzy controller in the avoid-obstacles behavior. The number of rules is determined by the number of the fuzzy membership functions of the controller input. As an example, the rules used for the angular velocity in the avoiding frontal obstacles controller are shown in Table 1. The rules for the linear velocity for the same controller are shown in Table 2. It should be noticed that all membership functions for the linear velocity are positive except for one. This is true because the robot is expected to actually move forward. However, if an obstacle becomes too close to the robot, the robot should be able to back off with a negative velocity. The results from Tables 1 and 2 are obtained using the following nine product-inference rules listed in Table 3. Similar rules were used for the other sensor pairs, i.e., 6&9, 5&10, and 4&11.

Table 1.
Rules for Angular Velocity (ω) in Avoid-obstacles

ω	x_{2l},(e.g. s_8)		
x_{1l},(e.g. s_{07})	ZR	MD	LG
ZR	ZR	N	NB
MD	P	ZR	N
LG	PB	P	ZR

Table 2.
Rules for Linear Velocity (v) in the Avoid-Obstacles

v	x_{2l},(e.g. s_8)		
x_{1l},(e.g. s_{07})	ZR	MD	LG
ZR	N	ZR	PS
MD	ZR	PM	P
LG	PS	P	PB

2.2.5. Defuzzification: In this step, the membership functions for the control action of angular velocity were defuzzified using the centroid method. Using min-operation rule of fuzzy implication, the results were obtained as follows in Table 4.

Finally, the output for angular velocity control action is computed by equation (4).

Table 3.
Inference Rules for the Avoid-obstacles

If $s_{07} = ZR$ and $s_{08} = ZR$	Then $\omega = ZR$ and $v = N$
If $s_{07} = ZR$ and $s_{08} = MD$	Then $\omega = N$ and $v = ZR$
If $s_{07} = ZR$ and $s_{08} = LG$	Then $\omega = NB$ and $v = PS$
If $s_{07} = MD$ and $s_{08} = ZR$	Then $\omega = P$ and $v = ZR$
If $s_{07} = MD$ and $s_{08} = MD$	Then $\omega = ZR$ and $v = PM$
If $s_{07} = MD$ and $s_{08} = LG$	Then $\omega = N$ and $v = P$
If $s_{07} = LG$ and $s_{08} = ZR$	Then $\omega = PB$ and $v = PS$
If $s_{07} = LG$ and $s_{08} = MD$	Then $\omega = P$ and $v = P$
If $s_{07} = LG$ and $s_{08} = LG$	Then $\omega = ZR$ and $v = PB$

Table 4.
Fuzzy Implication for the Avoid-obstacles Behavior

$$
\begin{aligned}
y_{l1} &= \min\left(\mu_{s_7=ZR}, \mu_{s_8=ZR}\right) \times \mu_{\omega=ZR} \\
y_{l2} &= \min\left(\mu_{s_7=ZR}, \mu_{s_8=MD}\right) \times \mu_{\omega=N} \\
y_{l3} &= \min\left(\mu_{s_7=ZR}, \mu_{s_8=LG}\right) \times \mu_{\omega=NB} \\
y_{l4} &= \min\left(\mu_{s_7=MD}, \mu_{s_8=ZR}\right) \times \mu_{\omega=P} \\
y_{l5} &= \min\left(\mu_{s_7=MD}, \mu_{s_8=MD}\right) \times \mu_{\omega=ZR} \\
y_{l6} &= \min\left(\mu_{s_7=MD}, \mu_{s_8=LG}\right) \times \mu_{\omega=N} \\
y_{l7} &= \min\left(\mu_{s_7=LG}, \mu_{s_8=ZR}\right) \times \mu_{\omega=PB} \\
y_{l8} &= \min\left(\mu_{s_7=LG}, \mu_{s_8=MD}\right) \times \mu_{\omega=P} \\
y_{l9} &= \min\left(\mu_{s_7=LG}, \mu_{s_8=LG}\right) \times \mu_{\omega=ZR}
\end{aligned}
$$

$$
y_l = \frac{\sum_{i=1}^{n} \mu_A(y_{li}) y_{li}}{\sum_{i=1}^{n} \mu_A(y_{li})} \tag{4}
$$

where: y_l is the output of each fuzzy controller, n is the number of clusters (rules), and μ_A is the membership function of the fuzzy set A (output).

2.3. Move-To-Point Behavior

The next behavior developed is the move-to-point behavior that consists of angle and distance controller. This behavior receives from the user a target position and orientation and compute the angle and the distance to the target from the current position of the robot. The angle controller keeps robot heading to the target while distance controller regulates the distance difference between the robot's current position and the target. The design process of these controllers is presented in the following:

`

2.3.1. Input-Output Fuzzification/Defuzzification: A target point is specified by the user as an *x-y* position. The angle and distance to the target point is calculated with-respect-to the robot's current position. These two values are used by the fuzzy controllers as error values. The developed fuzzy controllers for this behavior are: angle and distance controllers:

2.3.2. Angle Fuzzy Controller: The angle error is computed and supplied as input to the angle fuzzy controller. For this input, we constructed seven membership functions. Each membership function is a Gaussian curve function. The output of this controller is the angular velocity. The output is constructed using seven Gaussian membership functions as follows: LP=large-positive (0.4 rad/sec), P=positive (0.2 rad/sec), SP=small-positive (0.1 rad/sec), ZR=zero (0), SN=small-negative (-0.1 rad/sec), N=negative (-0.2 rad/sec) and LN=large-negative (-0.4 rad/sec). The rules for angle control are listed in Table 5. The result is obtained by the defuzzification of the output membership functions.

Table 5.
Rules for the Angle Control in the Move-To-Point

θ_{error}	-40	-20	-8	0	8	20	40
ω	-0.4	-0.2	-0.1	0	0.1	0.2	0.4

2.3.3. Distance Fuzzy Controller: Distance fuzzy controller is similar to the angle fuzzy controller in that it has one fuzzy input. The distance error is received as input by the distance fuzzy controller and it computes the required linear velocity for the robot to reach its final target. In this case, we also constructed seven Gaussian membership functions for the distance error input. The linear velocity output of the controller is constructed also using seven Gaussian membership functions as follows: VLP=very-large-positive (40 cm/sec), LP=large-positive (30 cm/sec), P=positive (20 cm/sec), small-positive (15 cm/sec), very-small-positive (10 cm/sec), zero (0), and negative (-5 cm/sec). The rules for distance control are shown in Table 6. The result is obtained by the defuzzification of output membership functions.

Table 6.
Rules for the Distance Control of the Move-To-Point

d_{error}	-15	0	15	35	55	75	100
v	-5	0	10	15	20	30	40

2.4. Wall-Following Behavior

For indoor mobile robot navigation wall-following behavior is essential. The sonar sensors on the left of the robot are used for wall-following behavior on the left side of the robot and the sonar sensors on the right side of the robot are used for wall-following behavior on the right side of the robot. Using fuzzy logic control, the distance between the robot and the wall is regulated. While the distance error is close to zero, the robot will move forward and follow a wall. Two fuzzy controllers are developed for the wall-following behavior, one for left side and another for right side of the robot. The robot will be forced by this behavior to follow the closest wall to its two sides. The design process of these two controllers is described in the following:

2.4.1. Input Fuzzification: Sonar readings on both left and right sides of the robot are used to develop the wall-following behavior. Sonar numbers 10, 11, 12, and 13 are used as input for the right wall-followings controller. In this behavior each fuzzy controller has two inputs. The results from sonar numbers 11 and 12 are compared and only the shorter distance is provided to the fuzzy controller as first input. The next input is received from comparing readings of sonar sensor numbers 12 and 13.

2.4.2. Fuzzy Membership Functions for the Input/ Outputs: We used four membership functions for each fuzzy input and seven membership functions for each fuzzy output. Each membership function is considered as a Gaussian curve function.

2.4.3. Fuzzy Rules: The wall-following behavior has two outputs; angular velocity and linear velocity. The rules for angular velocity are shown in Table 7 and the rules for linear velocity are shown in Table 8. The results are obtained by the defuzzification of the output membership functions.

Table 7.
Fuzzy Rules for the Wall-Following Angular Velocity Output

$\frac{x_L \rightarrow}{x_R \downarrow}$	0	1	2	3
0	-0.2	-0.1	-0.2	-0.3
1	0.1	0.0	-0.1	-0.2
2	0.2	0.1	0.1	0.0
3	0.2	0.2	0.2	0.3

`

Table 8.
Fuzzy Rules for the Wall-Following Linear Velocity Output

$\frac{x_L \rightarrow}{x_R \downarrow}$	0	1	2
0	2	2	2
1	2	3	3
2	2	3	3

3. EMBODIMENT OF BASIC BEHAVIORS

3.1. Hardware Arrangement

An RWI mobile robot called "TourMate" shown in Figure 3 has been used as the testbed for embodiment of the developed basic behaviors. TourMate has a Sony CCD camera, a frame grabber, a pan-tilt unit, 16 transmitter/receiver ultrasonic transducers, 22 tactile sensors, a speech synthesizer for text-to-speech conversion, a radio modem operating at 900 MHz for wireless telecommunication and two optical shaft encoders for odometer measurements. The robot has a Pentium-II 200 MHz CPU and uses the Linux operating system. It has distributed programming capability provided by its software architecture. The software structure has servers to reach the resources of the robot. The *tcxServer* is responsible for the communication and synchronization between all programs that may be running on the robot itself or on different computers. This server is the main tool for distributed programming. For example, obstacle avoidance algorithm can be run on one computer, and the image processing algorithms can be run on another computer and the communication between the two computers is done by means of *tcxServer*. The connection between the computers is TCP/IP connection. The *baseServer* helps to communicate with the sensors (such as sonar sensors) and the mobile base of the robot. The *speechServer* allows the user to send phrases that the robot should say. The *cameraServer* provides some useful functions to grab and manipulate images.

3.2. Behavior Embodiments and Testing Results

The fuzzy behaviors developed in the previous section were embodied on the TourMate mobile robot and individually tested whenever possible. The individual testing results are described and illustrated in the following section.

3.2.1. Emergency Behavior: This behavior was tested by manually moving an object towards the robot while the robot is moving forward. The results showed that the robot moved backward when the moving object became too close to the robot. That is when the distance between the robot and the moving object became less than the frontal distance (d_f) defined in Figure 2. However, the robot stopped when the moving object became closer than the established safe distance (d_s). In all the testing seniors the robot moved without hitting the object and stopped when it was necessary.

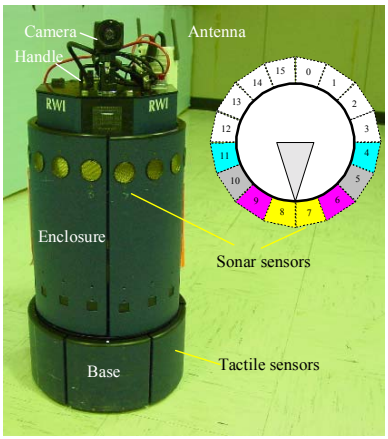

Figure 3. TourMate Mobile Robot and Its Sensors Arrangement

3.2.2. Avoid-obstacles Behavior: This behavior implements fuzzy controllers to regulate the robot's dynamic related to obstacle distance, position, and size. The robot was commanded to move forward inside the laboratory and outside in the hallway. We observed that the Avoid-obstacles behavior was activated when the frontal sonar sensors detected an obstacle. The robot turned slowly and avoided the obstacle while the obstacle is far away from the robot. However, the robot turned very quickly when an obstacle is presented close to the robot. The results showed that the robot was able, in all cases, to move forward and avoid obstacles. However, obstacles behind the robot were not considered in the testing of avoid-obstacle behavior. This was the case because it was assumed that the robot always moves forward.

3.2.3. Move-To-Point Behavior: In testing of this behavior, the robot was commanded to move from a point A to a point B then move back from point B to point A. The robot was positioned at the point A (0,0) with heading in the direction of x-axis. First, the robot turned left about 26 degrees towards point B (304, 152) then moved straight to the point B. After, the robot reached point B, it turned right about 180 degrees and moved back to point A. The robot stopped when it reached point A. Figure 4 shows the testing results for this senior. As we discussed before, move-to-point behavior consists of two fuzzy controllers. First, the

`

angle controller controls the robot heading by controlling the angular velocity of the robot. Second, the distance controller controls the linear velocity of the robot related to the distance from a target point.

3.2.4. Wall-Following Behavior: Wall followings behavior uses readings from sonar sensors on the left and right sides of the robot as it was described in pervious section. During the testing of this behavior, the robot moved forward while the desired distance between the robot and a wall was set to 600 mm. It was observed that while the robot following a wall, it turns away from the wall when the distance to the wall is shorter than 600 mm and the robot turns to the wall when its distance from the wall is larger than 600 mm. Otherwise the robot moves forward with faster linear velocity. In the cases when the robot turns to or away from the wall the linear velocity of the robot was significantly regulated (reduced/ increased). This action presented some problems when the test wall was not straight and included sharp edges or turns. One of the examples used in the testing of wall-following behavior is shown in Figure 5. The experiment showed that the robot moves forward and keeps the distance about 500 mm from the wall.

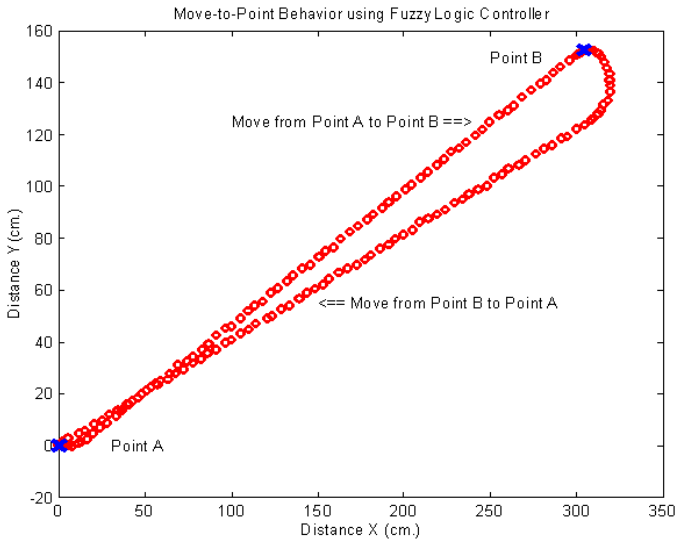

Figure 4. Move-To-Point Behavior Moving the Robot from Point A to Point B and then Back to Point A.

4. INTEGRATION OF BASIC BEHAVIORS

One of the basic tasks of mobile robots is to navigate towards a desirable target. To accomplish such a task without having the robot collides with obstacles, a combination of fuzzy basic behaviors scheme has been developed and implemented.

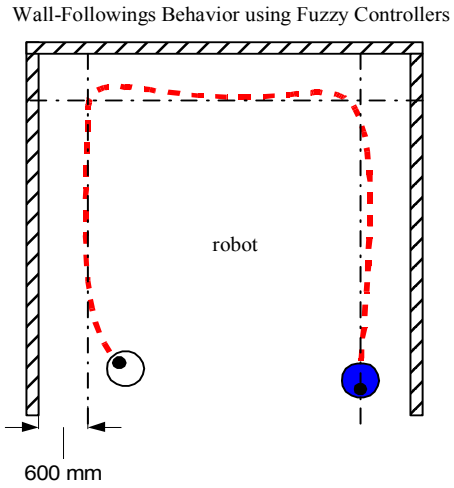

Figure 5. Example of Wall-Following Behavior

In this scheme, the basic move-to-point behavior is activated as the main behavior while other behaviors are used to aid the robot navigate under different circumstances and prevent collision along the path to the desired target. The fuzzy robot behaviors used in the scheme are: 1) *Move-To-Point*, 2) *Avoid-Obstacles,* 3) *Follow-Wall* and 4) *Follow-Center.*

In the developed scheme, the *avoid-obstacle* behavior is concurrently activated at all times with any other running behavior in order to limit the linear velocity of the robot. This guarantees the robot a collision free path at all times. The maximum linear velocity is calculated based on linear combination between the robot front safety distance and the linear velocity produced by the active behavior.

The *follow-wall* behavior is used to move the robot along general walls, obstacle walls and also to prevent the robot from trapping itself in complex obstacle shapes. During the execution of this behavior, the robot checks if it can move straight towards the target or if it has to execute other behaviors.

The *follow-center* behavior is similar to the follow-wall behavior except it is activated when there are two walls close to each other (narrow ally) to the left and right sides of the robot. In such a case the wall-following behavior may confuse the robot by forcing it to follow a zigzag path. The robot behavior must be switched to follow-center behavior in such a case. The transition between these behaviors is depicted in Figure 6. In this figure FW, M2P, and FC stand for Follow-wall, Move-to-point and Follow-center behaviors respectively.

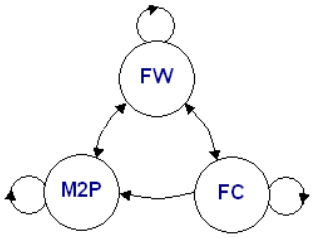

Figure 6. State Follow-Diagram of Three Basic Behaviors.

Table 9 lists the set of rules used for switching between behaviors relying on information gathered by the robot sonar sensors.

Table 9
Rules for Switching Between Behaviors

Current active behavior	Conditions	Switch to behavior
Wall-following	Two narrow walls	Follow-center
	No object close to robot or Robot moves slowly and No object on target direction	Move-to-point
Move-to-point	Robot far from target and Robot moves slowly or Obstacle is on the front	Wall-following
Follow-center	No wall on left or right	Wall-following
	No object on target direction	Move-to-point

The simulation result of using the above behavior integrations is shown in Figure 7, where the cross symbols (X) on the figure represent target points.

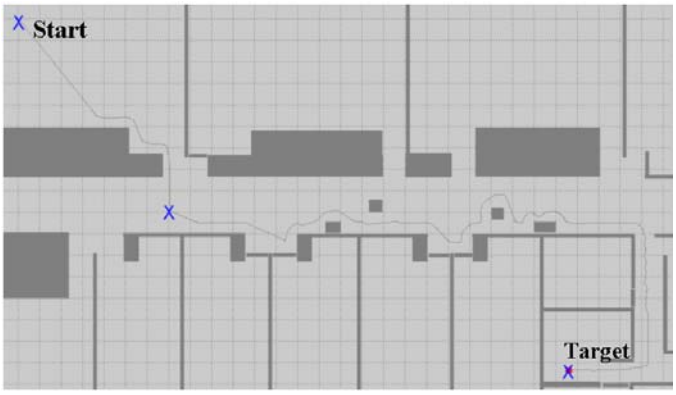

Figure 7. Robot Navigates Floor in a Building Using Integrated Behaviors

5. CONCLUSION

Four basic robot behaviors were designed using fuzzy control techniques. Each behavior was developed, implemented and tested separately. The individual testing results showed successful performance of each behavior. During the testing, TourMate mobile robot was able to move from point-to-point, avoid obstacle, follow walls, and stop when necessary. It was also demonstrated that integration of several basic behaviors allows the robot to navigate successfully to desired targets in a complex partially known environment without collision or deadracking.

Future work shall include development of more fuzzy behaviors, generate more testing results and implementation of adaptation mechanism into the fuzzy behaviors using Genetic Algorithms.

ACKNOWLEDGMENT

The authors would like to acknowledge and thank NASA, Ames Research Center (ARC) for funding and providing support for this research effort.

REFERENCES

1. R.C.Arkin. *Behavior-Based Robotics*. MIT Press, Cambridge Massachusetts, 1998.
2. R. A. Brooks. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*, RA-2(1):14—23, April 1986.
3. R.A.Brooks. *Cambrian Intelligence: The Early History of the New AI*. The MIT Press, Cambridge, Massachusetts, 1999.
4. M. Zimmermann. Intelligent control of mobile robots. In M. M. Gupta and N. K. Sinha, editors, *Intelligent Control Systems: Theory and Applications*, chapter 20, pages 546—566. IEEE Press, New York, 1996.
5. L. A. Zadeh. *Fuzzy logic, neural networks, and soft computing*. Communications of the ACM, 37(3):77—84, March 1994.
6. D. Driankov, H. Hellendoorn, and M. Reinfrank. *An Introduction to Fuzzy Control*. Springer-Verlag Berlin, USA, 2 edition, 1996.
7. L. A. Zadeh. *Fuzzy sets*. Information and Control, 8:338—353, 1965.
8. I. Ahrns, J. Bruske, G. Hailu, and G. Sommer. Neural fuzzy techniques in sonar-based collision avoidance. In L. C. Jain and T. Fukuda, editors, *Soft Computing for Intelligent Robotic Systems*, chapter 8, pages 185—214. Physica-Verlag Heidelberg, Germany, 1998.