Fuzzy Logic Control of an Autonomous Mobile Robot

Sawsan Abdel-Latif El-Teleity
Teaching Assistant of Communications and Electronics
Shoubra Faculty of Engineering, Benha University.
Cairo, Egypt.

Hala M. Abdel-Kader Mansour
Associate professor of Communications and Electronics
Shoubra Faculty of Engineering, Benha University.
Cairo, Egypt.

Zaky B. Nossair
Associate professor of Communications and Electronics
Electrical Engineering, Helwan University.
Cairo, Egypt.

Adly TagElDein
Assistant professor of Electrical Engineering,
Benha University.
Cairo, Egypt.

Abstract— **Navigation of autonomous mobile robots in unpredictable and dynamic environments is restricted by uncertainty, complexity, and unreliability issues of robots and their environments. In this context a navigation system for an autonomous mobile robot using intelligent fuzzy logic technique will be presented. Fuzzy logic control is able to emulate human experience about how best to control a system without needing accurate model equations, and can handle any perturbation in the system. A hierarchical behavior based control strategy has been devised in which four different reactive behaviors are combined by means of a fuzzy supervisor. For testing and validation, many simulations have been proposed which focus on: moving towards static or movable goal; escaping from local minima whenever it's detected; and traversing a cluttered environment with unknown static and dynamic obstacles.**

I. INTRODUCTION

Autonomous mobile robots should have ability to work in unpredictable and changing environments; hence they have to sense their environment and carry out their tasks regardless of the presence of obstacles in their working area. Classical motion control approaches are unable to handle small perturbations in the system. Soft computing techniques like fuzzy logic, genetic algorithm and neural network are considered for expressing the uncertainties in human.

Fuzzy control approach has been used by many researchers in sensor based navigational control of mobile robots. A comparison between proportional (P) controller and fuzzy logic controller for wall follow behavior was presented in [1]; FLC had shown better performance than P controller in early reaching the steady state. Ref [2] presented a behavior based fuzzy logic controller. The robot in that design was able to reach its goal avoiding obstacles but it might get into trap situation. Trap situation was studied in [3], but the trap could not have been detected until the robot goes many times within it then trying to escape it. An adaptive neuro-fuzzy controller was presented [4] for the reactive navigation of mobile robots in unknown environments. A state memory strategy was proposed for resolving the "dead cycle" problem depending on detecting virtual temporary targets for the robot leading it to escape trap. Ref [5] presented 124 rules fuzzy logic controller for a mobile robot to achieve safe navigation to the target point, comparing the results for two different types of fuzzy membership functions (MFs); Triangular and Gaussian; their results shown that Gaussian MFs performs better than triangular MFs. There are many other recent publications in using fuzzy logic in robot navigation problem [6-10]

In this paper, the navigation of Khepera I mobile robot in a simulated KIKs static and dynamic environment has been investigated using fuzzy logic control approach. Three individual behaviors have been implemented; the first behavior was to reach target point, the second to avoid obstacles within the robot path, and the third to escape from trap whenever it is detected. Escaping trap would be done through following either left wall or right wall till exit the situation. Lastly a command fusion between all behaviors has been made using another fuzzy logic controller to organize the operation and achieve the required task.

II. PROBLEM FORMULATION

For a mobile robot to operate in unknown static or dynamic environment; the navigation algorithm has to implement a reactive paradigm relying only on sensory information. The objective of this work is to implement the fuzzy controller on simulated Khepera robot in order to enable it to reach specific target avoiding any collisions and escaping from trap.

Khepera robot [11] is a differential-drive mobile robot that is useful for rapid testing in real world environments, and is currently in use in many research centers for various robotics experiments and applications.

III. PROGRAMMING ENVIRONMENT

Matlab has been selected as the programming platform for its ease of use, and graphical presentation capabilities. Fuzzy logic toolbox has been utilized; it presents great simplicity and flexibility in adjusting system parameters.

978-1-4577-0914-2/11/$26.00 ©2011 IEEE

The presented algorithm is tested on Khepera robot simulator (KIKS). The simulation is a Matlab application that simulates a Khepera robot connected to the computer in a very realistic way.

IV. THE PROPOSED FUZZY CONTROLLER

A hierarchical fuzzy logic control system has been proposed in which at first, two simple behaviors, namely "*Reach the Target*" (RT) and "*Avoid Obstacles*" (AO), have been carried out with two fuzzy controllers, called FLC1 and FLC2 respectively. RT behavior depends on relative position between the robot and target, while AO behavior uses only infra-red (IR) sensor signals, their position and orientation are shown in Fig. 1. AO takes place only if an obstacle appears on the robot path.

These controllers are sufficient to guarantee satisfactory navigation performances for Khepera robot in most of the navigation tasks, except for the risk of local minima (trap, dead cycle or infinite loop). To avoid this possibility, one further behavior, namely "*Escaping from Trap*" is carried out. This behavior is achieved through one of two sub-behaviors; "*Left Wall-Follow*" (LWF) or "*Right Wall-Follow*" (RWF), each of them has been implemented by another fuzzy controller; FLC3 and FLC4 respectively. These two controllers prevent Khepera from entering infinite loop situation by making the robot follow either left or right wall until it exits that situation, then continuing with previous two FLCs till reaching the target. The structure of the whole control scheme is shown in Fig. 2. The output from all sub-behaviors is entered as inputs to fuzzy supervisor to combine their results into one command to be sent to Khepera, this depends on a number of selection inputs (will be mentioned soon). The modular architecture of the controller has the following main advantages:

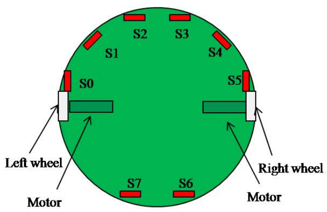


Figure 1. The position and orientation of sensors on Khepera

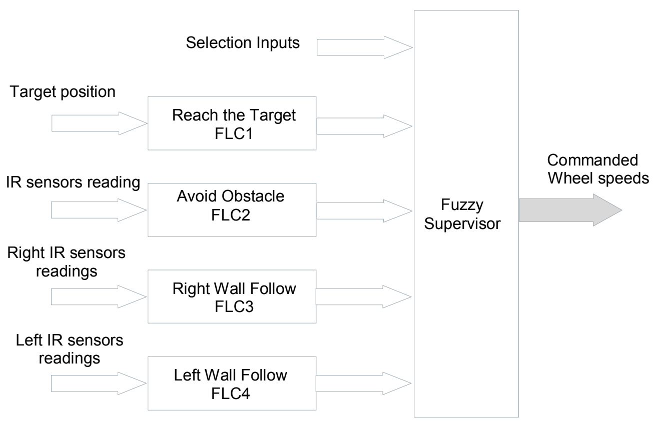


Figure 2. The proposed hierarchical behavior based control scheme

- Tuning and debugging operations are faster and easier.
- New simple behaviors can easily be added in order to expand robot skills.

A. *Reach the Target*

The user sets initial positions for robot and target, reach target behavior reacts on relative position between robot and target ignoring the presence and position of obstacles. In this behavior, the robot first turns until it is aligned to the target, and then moves straight forward. The inputs to this behavior are: the distance between the robot and target (Dist), and the alignment error of the robot (Dir) as shown in Fig. 3.

The fuzzy sets of the (Dist) input (expressed in millimeters) are "*zero*", "*near*" and "*far*", whereas (Dir) input is subdivided into "*L*" (Left), "*CL*" (Center-Left), "*C*" (Center), "*CR*" (Center-Right), and "*R*" (Right) fuzzy sets. The output variables are the speed command signals for the robot left and right wheels (V_L) and (V_R) respectively, and they are labeled "*NF*" (Negative Fast), "*NS*" (Negative Slow), "*Z*" (Zero), "*PS*" (Positive Slow), "*PF*" (Positive Fast).

All inputs have been represented by S and Z MFs for right and left boundary labels respectively and by triangular MFs for others as shown in Fig. 4 while the outputs have been represented by five uniformly spaced singletons. The rule base is shown in Table I. It can easily be interpreted using AND as a conjunction operator.

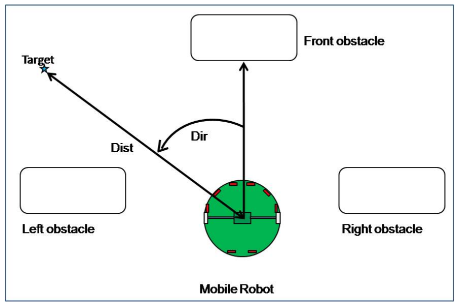


Figure 3. Mobile robot environment

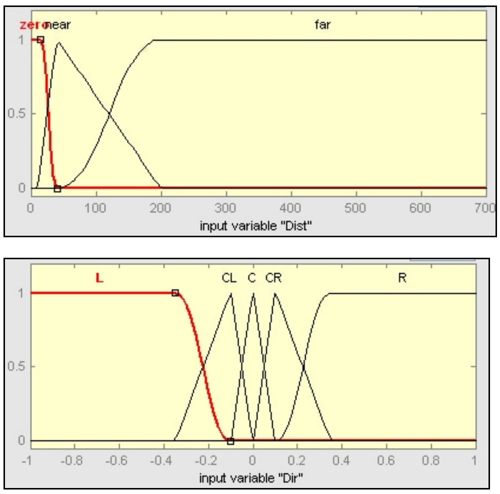


Figure 4. Input MFs of FLC1

TABLE I. RULE BASE OF FLC1

INPUTS (Logic AND)		OUTPUTS	
Dist	***Dir***	***VR***	***VL***
Zero	any	Z	Z
Near	C	PS	PS
Far	C	PF	PF
any	L	NF	PF
any	CL	NS	PS
any	R	PF	NF
any	CR	PS	NS

B. *Avoid Obstacle*

To guarantee a safe navigation of Khepera, FLC2 has been designed. It receives the signals coming from 8 IR proximity sensors and commands the speed of both wheels so that the robot moves in the opposite direction to close obstacles or goes straight ahead in the case of free path. The sensors are grouped into three groups named left sensors group consists of (S0, S1, S2), right sensors group consists of (S3, S4, S5), and front sensors group consists of (S2, S3). Three normalized values representing left, right, and front reading are entered to FLC2 as inputs. These inputs are calculated as follows:

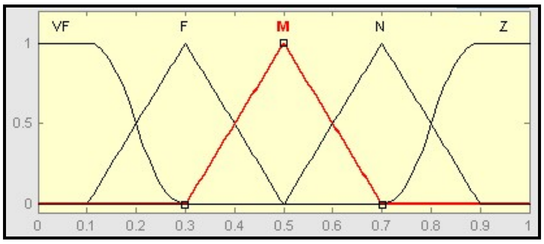
LS = Max (S0, S1, S2)/1023

FS = Max (S2, S3)/1023

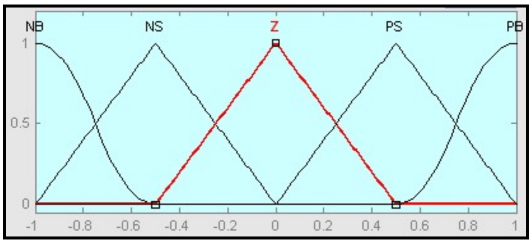
RS = Max (S3, S4, S5)/1023

(Hint: Maximum sensor reading is 1023)[12]

MFs of the three inputs are shown in Fig. 5.a, the labels for each input are "*VF*" (Very Far), "*F*" (Far), "*M*" (Medium), "*N*" (Near), and "*Z*" (Zero) depending on the distance between robot and obstacle (larger reading for smaller distance).



(a) RS, FS, and LS



(b) Angular velocity ω

Figure 5. Input MFs of FLC2

This behavior has three inputs; each has five fuzzy sets; so the total number of rules should be 125 rules [13]. In this work, AO behavior has been divided into two parts; AOpart1 and AOpart2; this would be come with reduction in rule base as will be shown. AOpart1 takes (RS, LS) as its inputs and gives the initially supposed angular velocity of the robot (ωi) as an output. This initial angular velocity does not take into consideration front sensor reading, so it's used only to give indication to the relation between left and right sensor reading. The second part AOpart2 takes ωi calculated in AOpart1 along with (FS) to give the required angular velocity (ω) and the linear speed of the robot (V).Then the left and right wheel speeds (V_L and V_R respectively) of the robot can be calculated [14]

$$V_R = (2V+ \omega L)/2 \quad (1)$$

$$V_L = (2V- \omega L)/2 \quad (2)$$

L= distance between robot wheels

The initial angular velocity takes five labels named "*NB*" (Negative Big), "*NS*" (Negative Small), "*Z*" (Zero), "*PS*" (Positive Small), and "*PB*" (Positive Big) and represented by triangular MFs as shown in Fig. 5.b. The linear speed is the output of AOpart2 and has five singleton values named "*Z*" (Zero), "*S*" (Small), "*M*" (Medium), "*B*" (Big), "*VB*" (Very Big).

Table II and III show the rule base of the two parts of this behavior. Each part consists of 25 rules (i.e 50 rules total). So, by such dividing for the rule base, the total number of rules has been greatly reduced.

TABLE II. RULE BASE OF AOPART1

LS / RS	Z	N	M	F	VF
Z	Z	PS	PS	PB	PB
N	NS	Z	PS	PS	PB
M	NS	NS	Z	PS	PS
F	NB	NS	NS	Z	PS
VF	NB	NB	NS	NS	Z

TABLE III. RULE BASE OF AOPART2

FS / ωi	Z	N	M	F	VF
NB	ω=ωi, V=Z	ω=ωi, V=Z	ω=ωi, V=S	ω=ωi, V=S	ω=ωi, V=S
NS	ω=ωi, V=Z	ω=ωi, V=S	ω=ωi, V=S	ω=ωi, V=B	ω=ωi, V=B
Z	ω =PB, V=Z	ω =PB, V=S	ω=ωi, V=M	ω=ωi, V=B	ω=ωi, V=VB
PS	ω=ωi, V=Z	ω=ωi, V=S	ω=ωi, V=S	ω=ωi, V=B	ω=ωi, V=B
PB	ω=ωi, V=Z	ω=ωi, V=Z	ω=ωi, V=S	ω=ωi, V=S	ω=ωi, V=S

For example the first rule:

IF (RS) is "Z" AND (LS) is "Z" AND (FS) is "Z" THEN (ω) is "PB" and (V) is "Z"

can be interpreted as: if there are obstacles surrounding left, front and right directions of the robot at very small distance, the robot should turn in place to find free path.

C. *Escaping from Trap*

1) *Trap-state detection:*

A method for trap-state detection [15] compares the instantaneous direction of travel with robot-to-target direction has been used. If the robot's direction of travel is more than 90^o off target,

$$|Dir| > 90^o \quad (3)$$

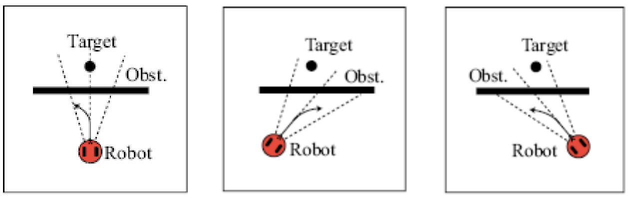
i.e the robot starts to move away from the target and is very likely about to get trapped; the system should carry out the recovery algorithm discussed below.

2) *Recovery Through Wall-following:*

Three states shown in Fig. 6 are designed for the robot to encounter this problem. NS (Normal State) for absolute alignment error smaller than 90^o ($|Dir| < 90^o$). The UNS right (Up-Normal State) for ($Dir <- 90^o$) , this means it needs to follow the wall on its right, while UNS left for ($Dir > 90^o$), this means it needs to follow the wall on its left.

a) *Left-wall Follow Behavior (LWF) (FLC3)*

This behavior tries to maintain the robot at constant distance from the left wall. It takes normalized readings of sensors S0, S1 as inputs and gives left and right wheel speeds as outputs. Each input has three linguistic labels "*Far*", "*Near*", and "*Close*" as shown in Fig. 7. The outputs are right and left commanded wheel speeds of the Khepera. Each output have five singleton labels named "*NF*", "*NS*", "*Z*", "*PS*", and "*PF*". Table IV shows the rule base of this behavior.



(a) NS (b) UNS left (c) UNS right

Figure 6. An illustration of robot states

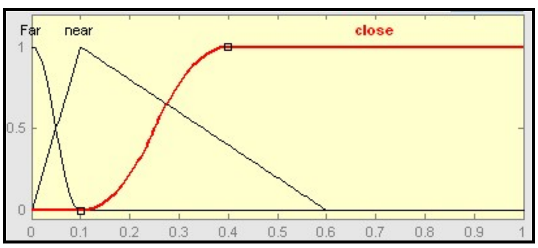


Figure 7. Input MFs of FLC3 and FLC4 (S0, S1 ,S4 , and S5)

TABLE IV. RULE BASE OF LEFT-WALL FOLLOW (FLC3)

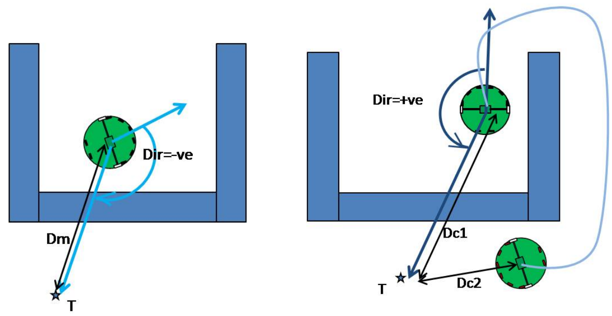
INPUTS (Logic AND)		OUTPUTS	
S0	***S1***	V_R	V_L
Close	Close	Z	PS
Close	Near	PS	PS
Close	Far	PS	Z
Near	Close	Z	PS
Near	Near	PS	PF
Near	Far	PF	PS
Far	Close	NF	Z
Far	Near	PS	PF
Far	Far	PF	PF

b) *Right-wall Follow Behavior (RWF) (FLC4)*

By the same way this behavior tries to maintain the robot at constant distance from the right wall. S5, S4 normalized signals are the inputs and left and right wheel speeds are the outputs. The inputs have the same MFs as those of FLC3 shown in Fig. 7. The rule base is same as that of FLC3 but with reversing right and left speeds.

B. *Fuzzy Supervisor*

In order to accomplish the task, a fuzzy supervisor determines the priority of execution for the four elementary behaviors. The fuzzy supervisor carries out this task according to number of selection inputs representing distance to the target, proximity of obstacles and the stored robot state. These selection inputs are Dist_sv (distance between robot and target), Lprox, Rprox (left and right proximity sensors respectively), AngState (the state of the robot) represents normalized absolute value of the alignment error, and iWFstate (initial wall follow state) represents the polarity of the angle at the moment of firing one of wall follow behaviors; this is required to maintain the wall follow state till exit trap even if the polarity of the angle is reversed during navigation as shown in Fig. 8. The wall follow state is maintained till the current distance between robot and target (Dc in the Figure) become smaller than distance measured at firing wall follow (Dm); at this point the robot ensures that it has been exit the trap.

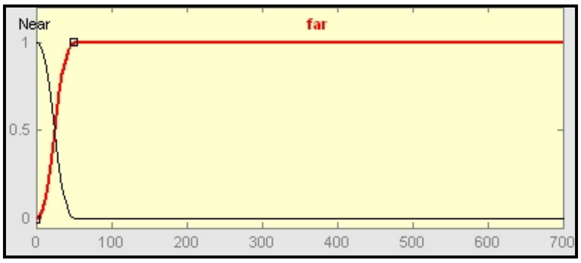


(a) (b)

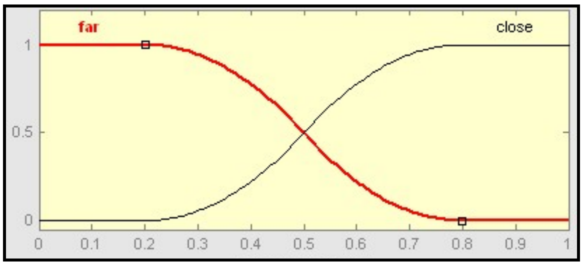
Figure 8. Reversing of angle polarity during robot navigation

The commanded wheel speeds of the four behaviors (FLC1, FLC2, FLC3 and FLC4) along with the selection inputs are the input signals of the supervisor FLC (Fig. 2). Whereas the outputs are the commanded wheel speeds sent to the Khepera.

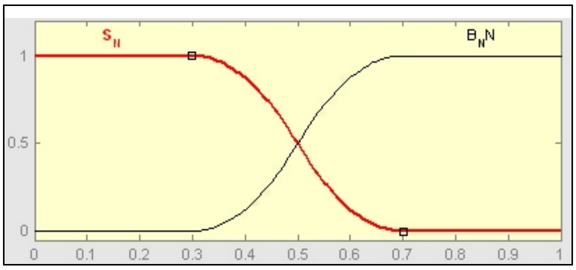
The labels for (Dist_sv) are "*Near*" and "*Far*", the labels for (Lprox), and (Rprox) are "*Close*" and "*Far*", the two labels for (AngState) are "*NS*", and "*UNS*", and labels for (iWFstate) are "*IRW*" (Initially Right wall) and "*ILW*"(Initially Left Wall). Fig. 9 shows MFs of the selection inputs. The outputs of supervisor are commanded right and left wheel speeds to be sent to Khepera robot.



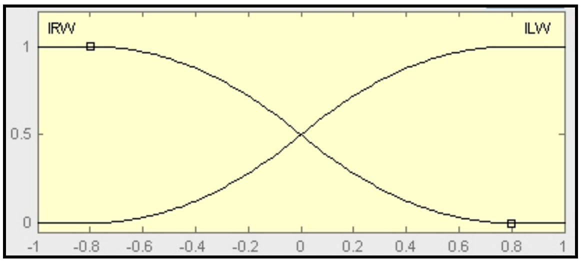
(a) (Dist_sv)



(b) (Lprox), and (Rprox)



(c) AngState



(d) iWFstate

Figure 9. MFs of the supervisor FLC selection inputs

TABLE V. RULE BASE OF SUPERVISOR FLC

INPUTS					OUTPUT
Dist_sv	***Lprox***	***Rprox***	***AngState***	***iWFstate***	***Behavior***
Near	any	any	any	any	RT
Any	Far	Far	Any	any	RT
Far	Close	any	NS	any	AO
Far	any	Close	NS	any	AO
Far	Close	any	UNS	ILW	LW
Far	any	Close	UNS	IRW	RW

The rule base for supervisory FLC is shown in Table V. It can be noted from this table that the priority is assigned to RT behavior in the following cases:

- When the distance between the robot and target is very near (the "*Near*" MF of Dist_sv or is fully activated). This rule solves the problem of unreachable target when obstacles are near the target.
- If both left and right sensors read low value (the "*Far*" MF of both Rprox and Lprox is fully activated) i.e. the robot's path is clear.

In these cases, independently of obstacles or state of the robot, the robot has to reach the target (the outputs of the supervisor coincide with those of FLC1).

While the priority is assigned to AO behavior when the robot is in normal state ("*NS*" MF of AngState is fully activated) and the maximum value of left or right proximity sensors is high (the "*Close*" MF of Rprox or Lprox is fully activated), in this case, independently of the actual location of the target, the robot has to avoid the collision (the outputs of the supervisor coincide with those of FLC2).

If the state of the robot is up-normal ("UNS' MF of AngState is fully activated), the robot expects to enter trap situation. If the value of left proximity sensors is high (the "*Close*" MF of Lprox is fully activated), and iWFstate is "*ILW*", it should follow the left wall until escaping from this trap (the outputs of the supervisor coincide with those of FLC3).While if the value of right proximity sensors is high, and iWFstate is "*IRW*", it should follow the right wall until escaping from this trap (the outputs of the supervisor coincide with those of FLC4).

In every intermediate condition the supervisor will perform a command fusion of the four FLCs blending their outputs to achieve a safe navigation toward the target.

V. SIMULATION RESULTS AND DISCUSSION

A sample experiment was chosen to show the effectiveness of the proposed fuzzy control strategy. Fig. 10 shows the performance of the robot during the navigation simulation, in which multiple traps were found in its way. Fig. 11 reports the linear wheel speeds provided by the AO, RT, LWF, the RWF behaviors, and by the fuzzy supervisor.

For illustration the path is divided to seven segments as shown in Fig. 10. The robot starts at start point S, needs to reach target point T. At first it senses no obstacles at beginning of Seg. 1, hence it firs RT behavior (supervisor output speeds coincides with FLC1 output speeds (see Fig. 11)). At the end of this segment, the robot senses an obstacle; it firs AO. Avoiding this obstacle results in alignment error smaller than - 90^o; hence the robot expects to enter trap situation. The robot selects to follow the right wall as long as absolute alignment error greater than 90^o and Dc > Dm1 (Seg. 2)(supervisor output speeds coincides with FLC4 output speeds) till absolute alignment error becomes smaller than 90^o (Seg. 3), it uses AO and RT. Through Seg. 2 and 3 the alignment error may take positive value greater than 90^o but the robot still storing that the right wall is the intended wall for escaping trap till Dc becomes smaller than Dm1. In Seg. 4 there is no stored wall follow state, and it uses AO and RT behaviors. Again at point 4 avoiding obstacles results in alignment error greater than 90^o, the robot expects to enter another trap situation. The robot selects to follow the left wall, and continue by the same way as before till reaching the target.

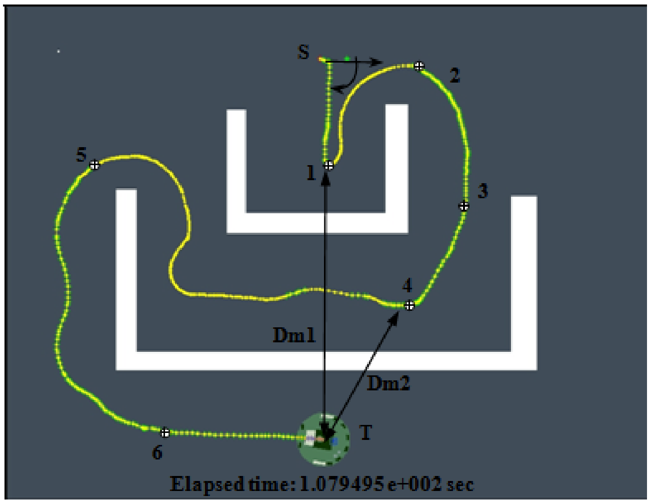


Figure 10. Measured path of Khepera robot in environment with multiple trap situations

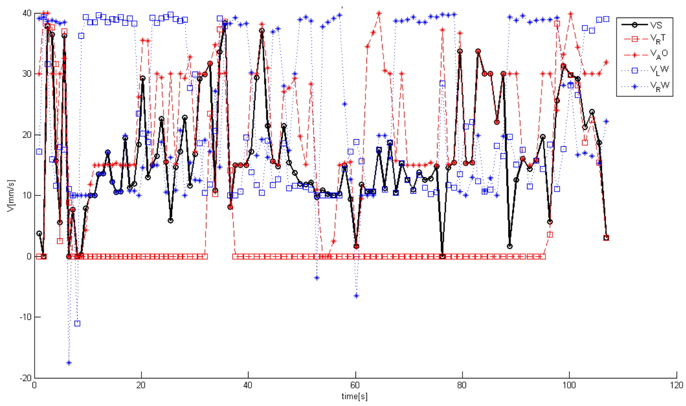


Figure 11. Linear speed of the robot in simulation of Fig. 10

VI. CONCLUSIONS AND FUTURE WORK

Fuzzy logic approach has been proved to be a simple and powerful technique for control problems. Appling it through behavior based modular architecture comes with great simplification in design process.

The proposed reactive fuzzy logic controller for Khepera has shown good performance in accomplishing its intended task in reaching its goal avoiding any collisions for most static and dynamic environments.

Controlling in a reactive manner, and having no map for the environment can lead to non optimized path; so future work on this research can apply this reactive controller with a shortest path algorithm to consider the optimized path. The proposed controller also can be applied for multiple robots environments to test its applicability. Searching and learning algorithms such as neural networks and genetic algorithms can be used to optimize the rules and membership functions.

REFERENCES

[1] V. M. Peri, "Fuzzy logic controller for an autonomous mobile robot," Celeveland State University,May 2005.

[2] M. Benreguieg, P. Hoppenot, H. Maaref, E. Colle, C. Barret, "Fuzzy navigation strategy: Application to two distinct autonomous mobile robots. Robotica, vol. 15, pp 609-615,1997.

[3] F. Cupertino, V. Giordano, D. Naso, L. Delfine , "Fuzzy control of a mobile robot,". Robotics & Automation Magazine, IEEE, vol. 13, pp 74-81, 2006.

[4] A. Zhu and S. X. Yang, "An adaptive neuro-fuzzy controller for robot navigation," Recent Advances in Intelligent Control Systems, Chapter 12. Yu, W, Springer-Verlag London Limited, London, pp 277-307, 2009.

[5] O. OBE, and I. DUMITRACHE, "Fuzzy control of autonomous mobile robot," U.P.B. Sci. Bull., Series C, Vol. 72, Iss. 3, 2010.

[6] H. Boubertakh, M. Tadjine, P.Y. Glorennec, and S. Labiod, "A simple goal seeking navigation method for a mobile robot using human sense, fuzzy logic and reinforcement learning," Journal of Automatic Control, University of Belgrade, Vol. 18,(1), pp 23-27, 2008.

[7] M. K. Singh, D.R.Parhi, S.Bhowmik, and S.K.Kashyap, "Intelligent controller for mobile robot: Fuzzy logic approach," The 12th International Conference of International Association for Computer Methods and Advances in Geomechanics (IACMAG)1-6 Goa, India, 2008.

[8] S. K. Harisha, Ramkanth Kumar P., M. Krishna, and S. C. Sharma, "Fuzzy logic reasoning to control mobile robot on pre-defined strip path," World Academy of Science, Engineering and Technology, No. 42, 2008.

[9] O. R. Motlagh, T. S. Hong, N. Ismail, "Development of a new minimum avoidance system for a behavior-based mobile robot," ScienceDirect, Fuzzy Sets and Systems, 160, pp1929-1946, 2009.

[10] M. Wang, and J. N. Liu," Fuzzy logic-based real-time robot navigation in unknown environment with dead ends,". ScienceDirect, Robotics and Autonomous Systems, 56, pp 625–643, 2008.

[11] Khepera I, K-team, http://www.k-team.com, 2002

[12] K-team, Khepera User Manual. Laussane, Switzarland,1999

[13] K.M., Passino, S.Yorkovich, Fuzzy Control, Addison Wesley Longman, Menlo Park, California, 1998.

[14] M. I., Ribeiro, , P. Lima, Kinematics Models of Mobile Robots, Instituto Superior Tecnico (IST), Portugal, 2002.

[15] J. Borenstein, and Y. Koren, "Real time obstacle avoidance for fast mobile robots," IEEE Transaction on Systems, Man, and Cypernetics, Vol. 19, No. 5, pp 1179-1187, 1989.