IJE TRANSACTIONS A: Basics Vol. 25, No. 4, (October 2012) 355-366

International Journal of Engineering

Journal Homepage: www.ije.ir

A Q-learning Based Continuous Tuning of Fuzzy Wall Tracking without Exploration

S. Valiollahi, R. Ghaderi*, A. Ebrahimzadeh

Department of Electrical and Computer Engineering, Babol University of Technology, Babol- 7414871167.

PAPER INFO

Paper history:
Received 19 Aprill 2012
Received in revised form 15 June 2012
Accepted 30 August 2012

Keywords:
Autonomous Navigation
Wall Tracking
Fuzzy Q-learning
Khepera Robot

ABSTRACT

A simple and easy to implement is proposed to address wall tracking task of an autonomous robot. The robot should navigate in unknown environments, find the nearest wall, and track it solely based on locally sensed data. The proposed method benefits from coupling fuzzy logic and Q-learning to meet requirements of autonomous navigations. The robot summerizes the obtained information from the world into a set of fuzzy states. For each fuzzy state, there are some suggested actions. States are related to their corresponding actions via simple fuzzy if-then rules, designed by human reasoning. The robot selects the most encouraged action for each state by Q-learning and through online experiences. The objective is to design a wall tracking algorithm which can efficiently adapt itself to different wall shapes in completely unknown environments. Q-learning is applied without any exploration phase, i.e. no training environment is considered. Experimental results on simulated Khepera robot validate that the proposed method efficiently deals with various wall contours from simple straight shape to complex concave, convex, or polygon shapes. The robot successfully keeps track of walls while staying within predefined margins..

doi: *10.5829/idosi.ije.2012.25.04a.07*

1. INTRODUCTION

Autonomy is a necessity of the robots that are increasingly replacing/cooperating human in homes and workspaces, or hazardous and unreachable environments. An autonomous robot should move purposefully and carry out specific tasks in unknown environments usually with unpredictable dynamics. A robot system comprises some main interacting units: sensors, preprocessing unit, decision making unit, controllers, motors, and actuators [1, 2]. Although, all these units affect the robot performance, decision making unit as the robot brain plays the most significant role. As the focus of this study is on decision making unit, a small wheeled robot with simple infrared sensors is selected so that other units do not require demanding processing [3]. A robust decision making algorithm is designed to equip the robot with the capability of tracking walls. The robot should find the nearest wall from any start points and continue tracking walls while staying in predefined margins. Imagine a robot working in a workspace crowded with people and furniture. Also, suppose the robot should move around a room, between rooms, or pass corridors to perform specific tasks e.g. delivering objects. In such cases, wall tracking provides a safe and easy way for the robot to move. Wall tracking is also applicable to multi task robots. For instance, if a robot tries to reach a goal behind an obstacle, tracking the obstacle walls is faster than avoiding it. Another application of wall tracking would be exploring unreachable and hazardous environments to gather information. A wall tracker robot moves in such environments, and provides valuable measurements of their circumferences. Most of researches [4-8] deal with relatively simple walls shapes and complicated algorithms. However, the proposed method is simple and at the same time efficient in face of complex wall shapes.

Major problems in autonomous robot navigation are as follow: (1) usually a mathematical model of the environments is not available; (2) sensed data is not reliable and precise; (3) real time response is necessary. Fuzzy logic is an adequate tool to address autonomous robot navigation due to its efficient properties. Generally, researchers mention three main advantages for fuzzy system. First, fuzzy logic is a flexible tool to model input-output relations. It allows incorporation of heuristic knowledge in form of if-then rules, and is an

* Corresponding Author Email: r_ghaderi@nit.ac.ir , (R. Ghaderi)

efficient alternative when the system cannot be exactly modeled. Fuzzy rules efficiently implement simple behaviors, independent of complicated mathematical models, to handle various complex tasks. Secondly, due to their qualitative descriptions, fuzzy behaviors are capable of being transferred to other environments without major changes. Finally, interpolative nature of fuzzy system leads to smooth movements of robots and an acceptable performance in face of noises and fluctuations in the sensed data [2].

Robot behaviors include reasoning, acting, and reacting according to the sensed information. Flexible behaviors require learning ability. Learning is to obtain information or improve current skills on the basis of experience, observation, and training. Q-learning is an efficient robust tool to tune the fuzzy systems due to its simple, model free, and dynamic structure [9].

A set of heuristic fuzzy if-then rules is designed by human reasoning and without any restrictive presumptions about the world. Incorporation of heuristic knowledge to design of fuzzy rules results in few number of rules, which in turn leads to a fast and easy implementation of planning process. The proposed method is independent of a global model or prior knowledge about the world. The only source of information comes from the locally sensed data by limited range infrared sensors. Fuzzy inputs are the reported data by infrared sensors while fuzzy outputs are the robot speed and steering angle. There are several suggested actions (steering angles) for each fuzzy state. The robot selects the most encouraged action for each state by Q-learning and through interactions with the world. Online tuning of the fuzzy inference provides a flexible decision making unit, which could adapt itself to unknown environments with various shapes of walls. Q-learning is applied without any exploration phase, i.e. no training environment is considered. The test environments are supposed to be completely unknown and may include variety of wall shapes to be good representatives of real world environments. For such environments, designing an appropriate training environment is usually very difficult or impossible.

Simulated experiments are conducted on sixty environments with various degrees of complexity, including concave, convex, and polygon walls shapes. Environments are designed by placing various shapes of obstacles in different sizes around walls. Considered obstacles include a wide range of geometrical shapes, which are reliable representatives of vertices and contours existing in real world environments. Experimental results on simulated Khepera robot validate that the proposed method efficiently deals with various wall contours from simple straight shape to complex concave, convex, or polygon shapes. The robot successfully keeps track of walls while staying within predefined margins.

The rest of paper is organized as follow: Section 2 deals with related literature. Section 3 provides a brief description about the Khepera robot. The proposed fuzzy Q-learning method is explained in section 4. Simulated results are depicted in section 5, section 6 includes a discussion about the proposed method, and finally the paper is concluded in section 7.

2. RELATED LITERATURE

Fuzzy logic has proven to be an efficient tool to autonomous navigation problems owing to its efficient properties such as independency of a precise model of the world, tolerance against uncertain or noisy information, fast response, and easy implementation [3, 10, 11]. Fatmi, A et al. and A. Karambakhsh et al. [10, 12], fuzzy logic is applied to design of robot behaviors. The parameters of applied fuzzy inferences are set offline. This is usually done based on expert knowledge or a trial and error process, which is not adequate in case of unexpected situations. Learning strategies are efficient alternatives to overcome this shortcoming.

Supervised learning algorithms usually require large amounts of training input/output data, which may be hard to obtain specially for autonomous navigations [13, 14]. Unsupervised and dynamic structure of reinforcement learning makes it a promising tool to online applications [4, 9, 15]. Q-learning is an efficient simple-structured model free reinforcement learning implementation, that is often used to adjust fuzzy inference systems; this coupling is also known as fuzzy Q-learning (FQL) [4-6]. FQL is applied, where a fuzzy inference of eight rules are designed offline and Q-learning is applied to tune fuzzy outputs memberships online by P.Y. Glorennec et al. [5]. A dynamic fuzzy Q-learning (DFQL) to fine tune both structure and parameters of fuzzy inference system for wall tracking task of an autonomous robot by M. J.Er and C. Deng [6]. However, sixteen initial fuzzy rules are designed offline base on a human driver's intuitive experience. A reinforcement ant optimized fuzzy controller (RAOFC) is proposed for wall tracking task of an autonomous robot by C. F. uang and C. H. Hsu, [7] , where no prior assignment of fuzzy rules is necessary. Eighteen fuzzy if-then rules are obtained after the learning process. The premise parts of fuzzy rules are obtained through a fuzzy clustering method while conclusion parts of rules are designed using Q-value aided ant colony optimization. This approach selects conclusion parts of rules from a set of suggested actions according to ant pheromone trails and Q-values, both of whose values are updated using reinforcement signals.

Both DFQL and RAOFC entail a complex time consuming and less intractable learning process in comparison with FQL. Since initially there is no rule in the fuzzy inference system, specially in RAOFC, these methods are not applicable without a training phase. Furthermore, the greater numbers of rules result in slower decision making processes which is not desirable for real time application. The proposed FQL approach in references [5, 8], fails to satisfy predefined margins successfully when applied to wall tracking task , as mentioned in referenc [7]. In this paper a FQL algorithm with higher accuracy in satisfying predefined margins is proposed for wall tracking task. The accuracy of FQL is improved by an efficient definition of reinforcement signal, which has a critical role in Q-learning performance. Experimental results demonstrate an acceptable tradeoff between simplicity and accuracy of the proposed wall tracking algorithm. Furthermore, applying the proposed method, the robot can find nearest walls from any start points or return to walls in case of failures. For this purpose, unlike DFQL and RAOFC, the outputs of all front sensors are considered in fuzzy states and not just half of them.

In FQL, DFQL, and RAOFC, after an exploration or a learning phase, one best action is chosen amongst some candidates as the output of each fuzzy rule. In such methods, the goal is to learn an optimal fuzzy rule base which is then exploited in experiments. The proposed method applies Q-learning without any exploration phase. No training environment is considered whose appropriate design is usually difficult or impossible specially for unknown environments. This outstanding point is the major difference between the proposed fuzzy Q-learning algorithm and the previously mentioned ones. The proposed method aims to produce an adaptive flexible wall tracking algorithm in face of various wall contours from simple to complex ones. Therefore, several options are considered as outputs of each fuzzy rule, any of which is probable to be selected depending on the encountered wall shape. The issue will be discussed more in section 6.

3. KHEPERA ROBOT

Khepera is a small mobile robot with a circular shape of 55 mm in diameter, 30 mm in height and 70 g in weight [16]. Using such a small robot, the navigation algorithm can be developed independent of a precise model of the robot. This in turn, allows an easy transfer of the designed navigation algorithm to other robots [17]. Khepera has eight infrared sensors, which are composed of an emitter and an independent receiver. These sensors (S_0, S_1,…, S_7) are arranged in a somewhat circular fashion around its body (see Figure 1) and measure distances in a short range from about 1 to 5 cm.

The sensor readings are integer values in the range of [0, 1023]. A sensor value of 1023 indicates that the robot is very close to the object, and a sensor value of 0 indicates that the robot does not receive any reflection of the infrared signal [3].

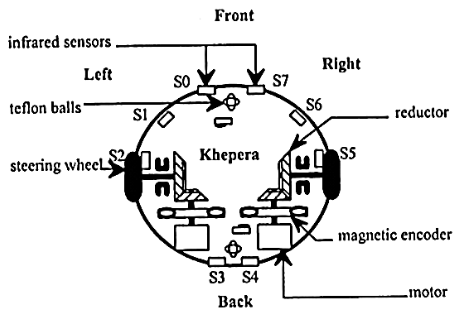


Figure 1. The Khepera Robot [3]

Three groups of sensors are considered as inputs to decision making unit i.e. left, front, and right sensors. The considered value for each group is the maximum output of each of the merged sensors.

Left sensor : $S_L = \max(S_1, S_2)$

Front sensor : $S_F = \max(S_0, S_7)$

Right sensor : $S_R = \max(S_6, S_5)$

Khepera has two wheels and two small Teflon balls. A DC motor moves each wheel. The robot's maximum linear and angular speeds are about 40 mm/s and 1.58 rd/s respectively [3]. Khepera robot is driven differentially, thus the controller has to give the angular velocities i.e. ω_L and ω_R, . As the human drivers reason more with linear velocity (v) and steering angle (Φ), the output of fuzzy rules are considered v and Φ. If the cornering force on wheels are neglected, the control commands are easily generated by simple geometrical equations [18] that make correspondence between (v,Φ) and (ω_R, ω_L) (Equations (1) and (2)):

$$v = r.(\omega_R + \omega_L)/2 \tag{1}$$

$$\frac{d\varphi}{dt} = r.(\omega_R + \omega_L)/2T \tag{2}$$

where, r is the radius of the wheels.

4. THE PROPOSED FUZZY Q-LEARNING METHOD

To get a better insight to the proposed method, brief descriptions of fuzzy logic and Q-learning is provided below:

4. 1. Fuzzy Logic Fuzzy logic maps an input space to an output space by a list of if-then statements called rules. The rules are structured in a human decision making format. A typical rule could be as follows:

IF Obstacle is Far THEN Speed is Low

where, "*Obstacle*" and "*Speed*" are input variable and output variable, respectively. Rule base is the key element in robot intelligence. A rule base should consider all possible cases of linguistic adjectives for input variables. Linguistic adjectives, like "*Far*" and "*Low*", are described by membership functions which map the value of the variables to membership degrees ([0,1]) in each fuzzy set (see Figure 2).

Fuzzy sets are often defined as piecewise linear shapes (triangular or trapezoidal) to reduce the computational complexity of acquiring membership degrees. Fuzzy logic is able to handle imprecise data efficiently. If input data is sensed imprecisely, the membership degree in each fuzzy set may change but the relative membership degrees over fuzzy sets remain the same qualitatively. Fuzzy operators are applied to combine input variables and manipulate the output fuzzy set for each rule. The outputs from all rules are then aggregated to form a single fuzzy set. To extract a crisp output from the aggregated fuzzy set, a defuzzification method is applied [19].

4. 2. Q-learning

Q-learning usually provides best fit to requirements of real world applications due to its unsupervised, dynamic, and model free structure. The robot observes the world as a set of state-action pairs. To each action a Q value is assigned. Transition from one state to another (via proper actions) may bring it punishment or reward, and accordingly Q values are updated as Equation (3):

$$Q(x_t,a_t) \leftarrow Q(x_t,a_t) + \beta\{r_t + \gamma V_t(x_{t+1}) - Q(x_t,a_t)\} \tag{3}$$

where, x_t is the current state, a_t is the action taken at state x_t, $Q(x_t,a_t)$ is the Q value of state x_t , $V(x_{t+1})$ is an estimated value of the new state (x_{t+1}). β and γ are the learning rate and forgetting factor respectively both in range of [0 1], and r_t is the immediate reinforcement scalar signal. Parameter β is redundant i.e. one can eliminate β (taking β = 1), and $0.9 < \gamma < 0.99$ [9].

4. 3. Fuzzy Q-learning

A heuristically designed fuzzy rule base with N=8 rules is applied. Input variables are the three groups of Khepera infrared sensors mentioned in Section 3 i.e. S_L, S_F, and S_R which approximately cover a 180 degrees view of the surroundings. To reduce the number of rules, just two linguistic adjectives are considered for inputs: Far (F) and Near (N) whose membership functions are depicted in Figure 2. The approach is to find the nearest wall and continue tracking it. The nearest wall (right or left) is determined by checking the value of parameter (P) as below:

$$\textit{If}\ \ S_L \succ S_R\ \ ;\ P=1\ ;\ \textit{Else}\ P=0\ ;\ \textit{End}$$

The outputs are the robot speed (S) and steering angle (Φ) as Figure 3. S_i ($1 \le i \le N$) is a fixed value for each rule which can be zero(Z), C_1*Vmax, C_2*Vmax. C_1 and C_2 are constants smaller than one and $C_1 < C_2$. Vmax is the maximum considered speed. S_i is described by linguistic adjectives of PB(Positive Big), PS(Positive Small), Z(Zero), NS(Negative Small), and NB(Negative Big) which are symmetric i.e. PB=-NB, PS=-NS. The rule base is as follow:

$$\begin{aligned}
&R_1: IF(S_L,S_F,S_R)=NNN\ \ THEN\ \ s_1=Z && \&\ \ \varphi_1=(2\times p-1)PB\\
&R_2: IF(S_L,S_F,S_R)=NNF\ \ THEN\ \ s_2=Z && \&\ \ \varphi_2=PB\\
&R_3: IF(S_L,S_F,S_R)=NFN\ \ THEN\ \ s_3=C_1\times \mathrm{V}_{\max} && \&\ \ \varphi_3=Z\\
&R_4: IF(S_L,S_F,S_R)=NFF\ \ THEN\ \ s_4=C_1\times \mathrm{V}_{\max} && \&\ \ \varphi_4=PS\\
&R_5: IF(S_L,S_F,S_R)=FNN\ \ THEN\ \ s_5=Z && \&\ \ \varphi_5=NB\\
&R_6: IF(S_L,S_F,S_R)=FNF\ \ THEN\ \ s_6=Z && \&\ \ \varphi_6=(2\times p-1)PB\\
&R_7: IF(S_L,S_F,S_R)=FFN\ \ THEN\ \ s_7=C_1\times \mathrm{V}_{\max} && \&\ \ \varphi_7=NS\\
&R_8: IF(S_L,S_F,S_R)=FFF\ \ THEN\ \ s_8=C_2\times \mathrm{V}_{\max} && \&\ \ \varphi_8=(2\times p-1)NB
\end{aligned}$$

The applied fuzzy model is the Sugeno, or Takagi-Sugeno-Kang method [20]. Equation (4) computes the robot speed:

$$S = \frac{\sum_i \alpha_i s_i}{\sum_i \alpha_i} \tag{4}$$

where, α_i is the truth value of the ith rule calculated by product method. For each rule, there are J=6 actions or suggested steering angles (φ[i,j], $1 \le j \le J$). the six suggested actions are distributed equally in predefined intervals. To each of these actions, a q value (q[i,j]) is assigned. The values of φ[i,j] are determined offline. However, deciding which of these steering angles are selected for rules is done online by Q-learning.

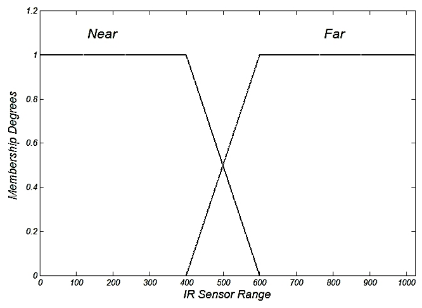


Figure 2. Fuzzy membership function

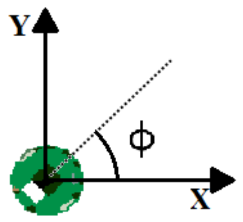


Figure 3. Demonstration of robot steering angle in Cartesian coordinates

TABLE 1. Ranges of Fuzzy Outputs

Z	PS	PB	NS	NB
0	[0 90]	[90 180]	[-90 0]	[-180 -90]

1. t=0, observe the state x_t.
2. For each rule i, choose φ[i,j] with maximum q value.
3. Compute (xt) and its corresponding Q(xt, (xt)).
4. Apply the action (x_t). Observe the new state xt+1.
5. Receive the reinforcement r_t.
6. Compute an estimate value of the new state xt+1 i.e. V_{xt+1} .
7. Update q[i,j].

Figure 4. Fuzzy Q-learning algorithm

To determine values of φ[i,j] for each rule, the total possible degree of rotation for robot (i.e. approximately 360 degrees) is divided to five intervals of as Table 1.

To obtain the robot steering angle a fuzzy Q-learning approach is applied as depicted in Figure 4. Q is a N*J matrix whose elements (q[i,j]) are initialized to zero. Among the J suggested actions for each rule, only the action associated with maximum q value (Equation (7)) is selected and takes part in computation of the overall action see Equations (5) and (6).

The explained method is depicted as a flowchart in Figure 5.

$$\phi = \frac{\sum_i \alpha_i \varphi_{im}}{\sum_i \alpha_i} ; 1 \le i \le N, 1 \le j \le J \tag{5}$$

$$\varphi_{im} = \varphi[i, j] \mid \max_j q[i, j] \tag{6}$$

x_t is the fuzzified state of sensors observed in the time t e.g. (S_L, S_F, S_R)=FFN. In each iteration of the algorithm, maximum q values are updated applying Equations (7), (8), (1), and (9) respectively.

$$q_{im} = \max_j q[i, j] \tag{7}$$

$$Q(x_t, a(x_t)) = \sum_i \alpha_i q_{im} \tag{8}$$

$$q_{im} = q_{im} + \alpha_i dQ \tag{9}$$

As demonstrated in Figure 5, the robot is an integrated system comprising three main units of sensors, decision making, and motors. The robot intracts with environment in a closed loop. In robot viewpoint, the sensed environment is abstracted into eight fuzzified states e.g. x_t=(S_L=F,S_F=F,S_R=N). A fuzzy if-then rule is considered for each state.

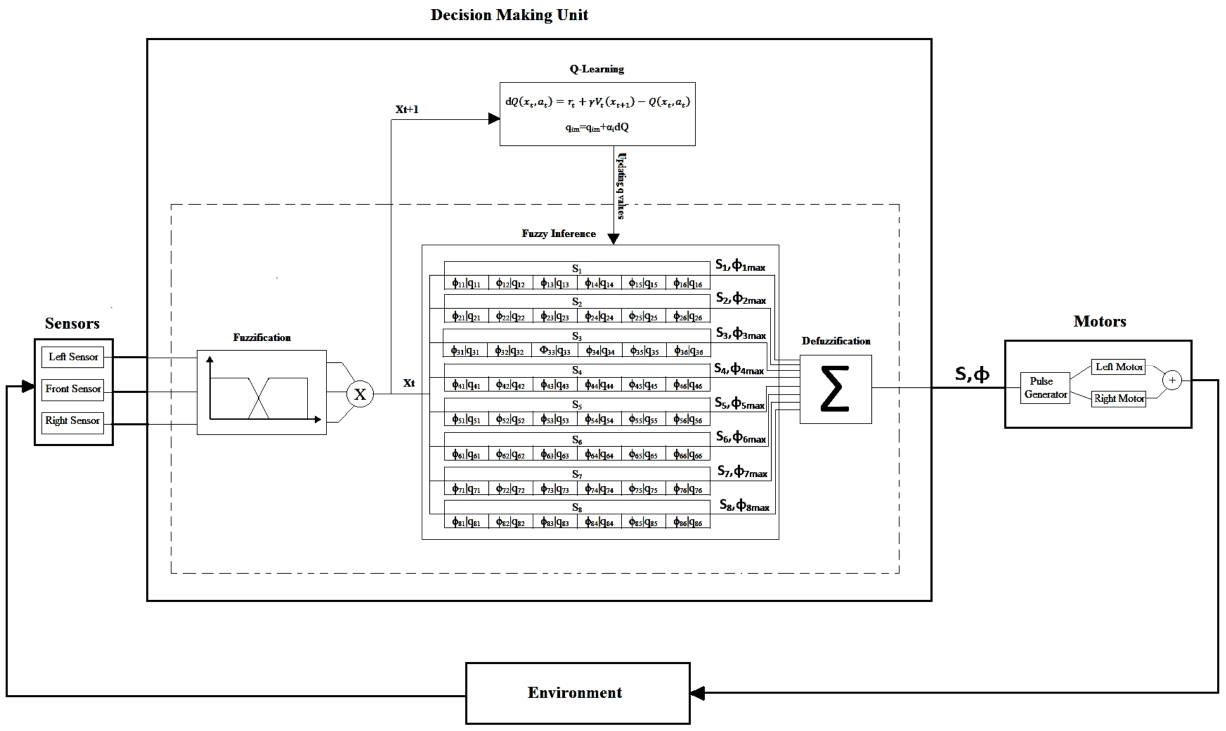


Figure 5. Flow chart of the proposed method

The outputs of each rule are a fixed value velocity and a steering angle selected by Q-learning from six suggested actions. In each step, the actions with maximum q value is selected and others are neglected. In each current state (x_t), the overall velocity and steering angle of the robot are computed by a defuzzification method and appropriate control commands are sent to motors. q values are updated after receiving the immediate reinforcement signal by observing the new state (x_{t+1}).

Choosing an appropriate reinforcement signal (r_t) is a deciding factor in the overall performance of Q-learning. r_t is usually defined with respect to the existing obejctives. In this work, the objective is to keep the robot within predifned margins [d_1 , d_2] from walls as shown in Figure 6; d_1=30mm is the minimum allowed distance from walls and d_2=60mm is the maximum allowed distance.

To improve the acuracy of wall tracking, the distance constraint (d_2-d_1) is splitted in five parts and the reinforcement signal (r_t) is allocated according to which part the robot center locates as Equation (10):

$$r_t = \begin{cases} 5; & d_1 < d < d_1 + \frac{d_2 - d_1}{5} \\ 4; & d_1 < d < d_1 + \frac{2(d_2 - d_1)}{5} \\ 3; & d_1 < d < d_1 + \frac{3(d_2 - d_1)}{5} \\ 2; & d_1 < d < d_1 + \frac{4(d_2 - d_1)}{5} \\ 1; & d_1 < d < d_2 \\ -4; & \text{otherwise} \end{cases} \tag{10}$$

5. SIMULATED EXPERIMENTS

Applying the proposed wall tracking algorithm, the robot is expected to find the nearest wall from any start points and continue tracking walls while staying in predefined margins. Sixty different simulated environments are designed in three levels of complexity i.e. simple, normal, and complex; each group contains twenty environments. Levels of complexity are determined according to the number and shapes of obstacles locating around walls. The robot is expected to track these obstacles which are parts of walls and not avoid them. The obstacles are placed randomly around walls. Some typical shapes of obstacles are depicted in Figure 7.

The obstacles in sixty simulated environments include a wide range of geometrical shapes, which are reliable representatives of vertices and contours existing in real world environments. The basic geometrical patterns such as rectangular, triangular, concave, and convex objects are considered as basic obstacles. More complex obstacles are designed by mixing these basic shapes, which do not necessarily have known geometrical shapes. Obstacles areas vary from 1-10 times of the robot area. Areas of all environments are the same so that the results could be comparable. Visual instances for simple, normal, and complex environments are depicted in Figures 10, 12, and 14 respectively. The degrees of complexity for the three groups of environments are defined quantitatively as Table 2.

Each expriment is repeated ten times to assure rialiblity of obtained results. The reported result for each experiment is the average of ten executions. According to statistical measures like variance and average, repeating the experiments for more than ten times does not make sensible changes in the averaged results. Simulated expriments are executed in KiKs

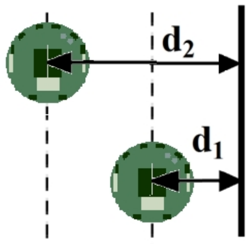


Figure 6. Predefined margins (d_1=30mm, d_2=60mm)

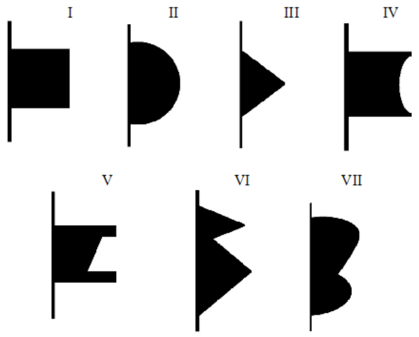
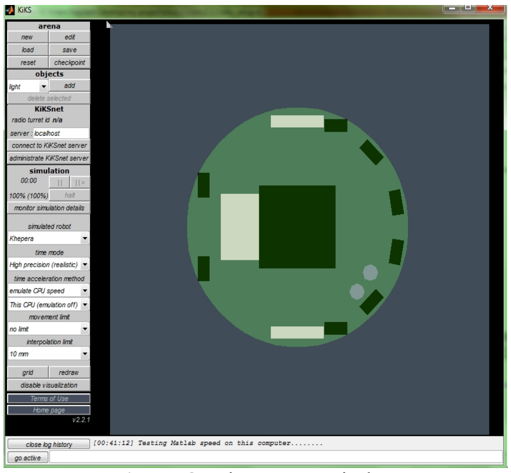


Figure 7. Typical shapes of obstacles. I: rectangular, II: convex, III: triangular, IV: concave, V & VI: polygon, VII: concave-convex

TABLE 2. Quantitative description of simple, normal, and complex environments

Environments Groups	Type of obstacles	Number of obstacles
Simple	I-II	2-4
Normal	I-IV	4-8
Complex	I-VII	8-12



objects: light
radio turret id n/a
server : localhost
00:00
100% (100%)
simulated robot: Khepera
time mode: High precision (realistic)
time acceleration method: emulate CPU speed
This CPU (emulation off)
movement limit: no limit
interpolation limit: 10 mm

```
[00:41:12] Testing Matlab speed on this computer........
```

Figure 8. KiKs setup window

TABLE 3. Success Percentage in Finding the Nearest Wall

	worst	average	best
Success Percentage in Finding the Nearest Wall:	85%	90%	100%

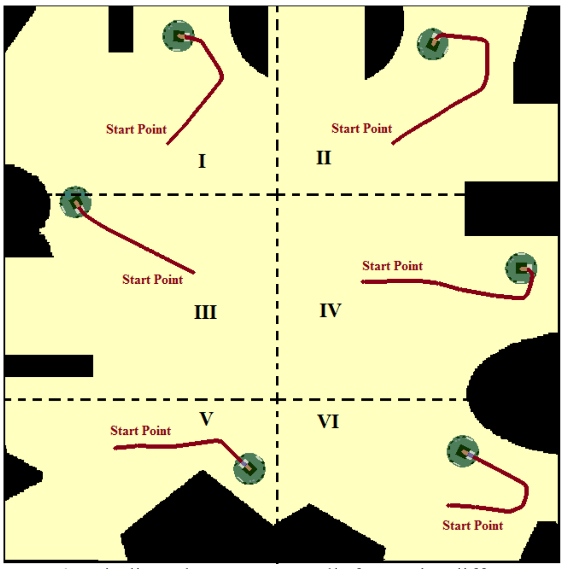


Figure 9. Finding the nearest wall from six different start points in a complex environments.
The average success percentage over sixty simulated environments is 90%.

software arena. KiKS is an abbreviation for "Kiks is a Khepera Simulator". The program is a Matlab application that simulates a Khepera robot connected to the computer in a very realistic way. The simulated Khepera is controlled from Matlab in the same way as real, physical Kheperas. The simulator can be downloaded from www.tstorm.se/projects/kiks. [21]. The figure below (Figure 8) shows the setup window of KiKs. Performance of the proposed method is evaluated according to three criteria as listed below:

❖ Finding the Nearest Wall from the Start Point

As claimed before, the robot can find the nearest wall and keep tracking it even if the start point is not next to a wall. To experiment this, six different start points are considered for each environment. The overall success percentage of finding nearest walls is 90% and obtained by averaging over the result of six start points, and then over sixty environments. Among these sixty environments the worst percentage is 85% while the best one is 100% (Table 3). To choose start points, environments are partitioned into six and a start point is selected randomly in each partition. Figure 9 shows considered partitions and the result of an experiment performed in a typical complex environment.

❖ Keeping the Predefined Margins from Walls

If the robot is in the wall tracking mode, which could be recognized through sensors states, the distance of the robot center to the wall is measured by sensors outputs in each step. If the measured distance is not in predefined margins of [d1 d2], the failure counter is increased by one. Finally, the failure percentage for each execution is obtained. The ultimate result is the average over twenty environments of a group (Table 4). As expected, the failure percentage (8%-12%) increases form simple to complex environment. As well as counting the robot failures in keeping predefined margins, the maximum, average and minimum steps needed to return within predefined margins are also computed. The steps depicted in Table 4 is obtained by averaging over twenty environments of a group.

❖ Speed

The total number of steps for completing a full loop around the environment is also reported in Table 2. The maximum allowed number of steps is 10000 for all three groups of environments that is quite enough, considering the environment with maximum circumference. To highlight the effective role of Q-learning in tuning fuzzy inference parameters, the same experiments were repeated for a fuzzy rule base whose parameters were set completely offline and without Q-learning (Table 5). The output membership functions in fuzzy method are singletons fixed at the mean value of intervals considered for fuzzy Q-learning method. Comparisons between two methods according to Tables 4 and 5 reveal superiority of fuzzy Q-learning over fuzzy method in terms of speed and accuracy. The failure percentage of fuzzy method varies from 41% to 74% which is much more greater than failure range of fuzzy Q-learning. Figures 10-17 provide visual comparisons, which clarify the smoother and far more accurate performance of fuzzy Q-learning over fuzzy method.

Figure 10 demonstrates a typical simple environment where the fuzzy Q-learning could successfully track walls with rectangular /convex shapes of different sizes, and stay in predefined margins. On the other hand, the fuzzy method demonstrates an inferior performance even in tracking straight walls (Figure 11). Furthermore, the fuzzy method fails to track vertices of rectangular obstacles. To robot, a vertex is a point between two different states. If the robot cannot adapt itself properly and quickly to the new state, it may fall in danger of losing track of walls. As can be seen, applying the proposed method, the robot can successfully continue tracking walls even when shapes of walls changes thanks to efficient selection of fuzzy outputs by Q-learning.

However, applying fuzzy method, the decision making unit cannot provide the adequate change of steering angle so that the robot can keep track of walls when states changes e.g. in case of a vertex. As can be seen from Figure 11, the robot gets out of margins while reaching vertex of the upper right square. For the upper left square, the robot completely loses track of the left wall, and finally comes within margins after a time consuming loop. The convex walls arise a similar challenge for the robot. However, the changes of states are not so sharp as of a vertex.

Figure 12 shows a typical normal environment, the proposed method can successfully track walls even with new added wall shapes i.e. traingular and concave walls. Furthermore, the more number of obstacles around walls than simple environement does not disturb the robot from tracking walls correctly. This spots adaptability of the proposed method to environmental changes. For the fuzzy method (Figure 13), similar failures with simple environment are repeated. The number of these failures increases, as more changes happen in wall shapes.

TABLE 4. Performance of fuzzy Q-learning method.

Environments Groups	**Failures**	**Steps needed to return within margins**			**Steps**
		Maximum	**Average**	**Minimum**	
Simple	8%	7	2	1	1775
Normal	10%	9	3	1	2385
Complex	12%	11	5	1	2892

The reported results for each group of environments are the average over twenty environments.

TABLE 5. performance of fuzzy method

Environments Groups	**Failures**	**Steps needed to return within margins**			**Steps**
		Maximum	**Average**	**Minimum**	
Simple	41%	581	176	1	2571
Normal	52%	693	207	1	3458
Complex	74%	710	271	1	4298

The reported results for each group of environments are the average over twenty environments.

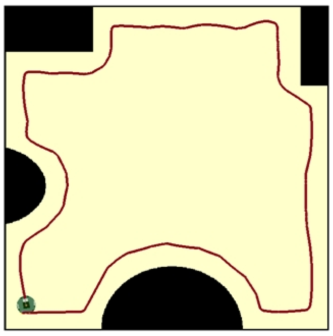


Figure 10. Performace of fuzzy Q-learning method in a simple environment

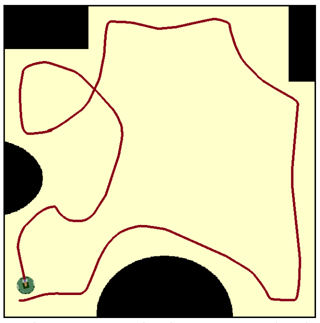


Figure 11. Performace of fuzzy method in a simple environment

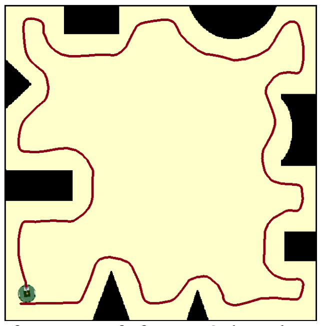


Figure 12. Performace of fuzzy Q-learning method in a normal environment

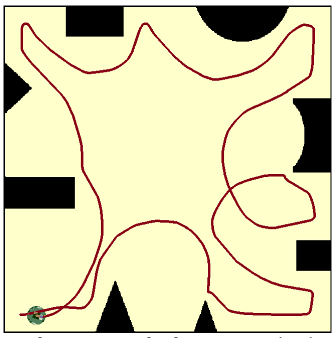


Figure 13. Performace of fuzzy method in a normal environment

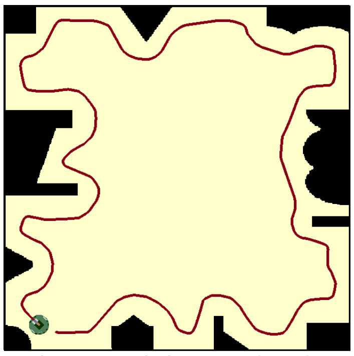


Figure 14. Performace of fuzzy Q-learning method in a complex environment

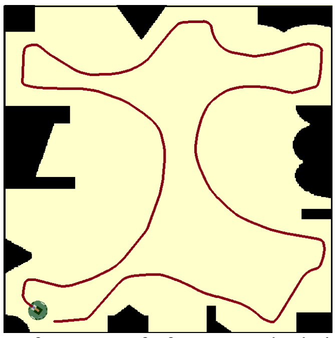


Figure 15. Performace of fuzzy method in a complex environment

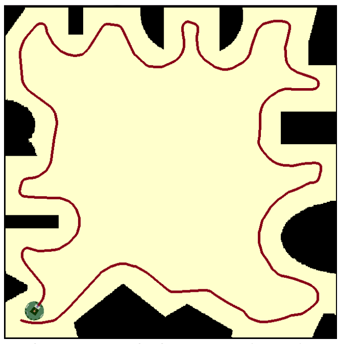


Figure 16. Performace of fuzzy Q-learning method in a complex environment

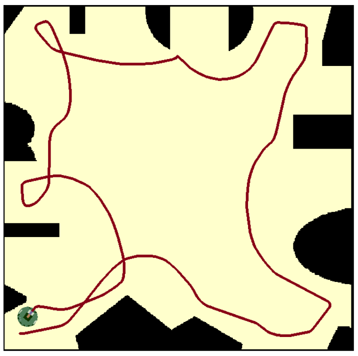


Figure 17. Performace of fuzzy method in a complex environment

Figures 14 and 16 represents two typical complex environments where the proposed method can successfully track walls with basic rectangular, traingular, convex, and concave shapes, and also mixtures of these basic shapes. As can be seen, the robot can quickly adapt itself to the changes of wall shapes. However, as environments get more complex the inefficiency of fuzzy method becomes more apparent (Figures 15 and 17).

6. DISCUSSION

As mentioned in Section 4.3, six suggested steering angles are considered for each rule instead of one fixed sigleton. This endows the decision making algorithm capability of adaptaion to variuos environmental changes. As demonstrated in previous section, fixed outputs for fuzzy rules cannot deal with environments with variuos wall contours. Even if, we assume that learning one best output for each fuzzy rule is responsive, desing of an appropriate training environment is not always possible specially for unknown environments. The proposed fuzzy Q-learning algorithm produces a dynamic rule base where the output of each rule may change during time intervals so that the entire decision making unit can adapt itself to various wall shapes, driven by the objective of staying in predefined margins.

Applying the proposed fuzzy Q-learning, a simulated expriment is started with q values initialized to zero. In other words, no exploration phase or training environment is considered for Q-learning. Any of the six suggested actions may be once the best action during some time intervals depending on the encountered wall shapes.

Figures 18-20 show the process of choosing actions for rules in typical simple, normal, and complex environements respectively for right wall tracking task during a complete loop around these environments.

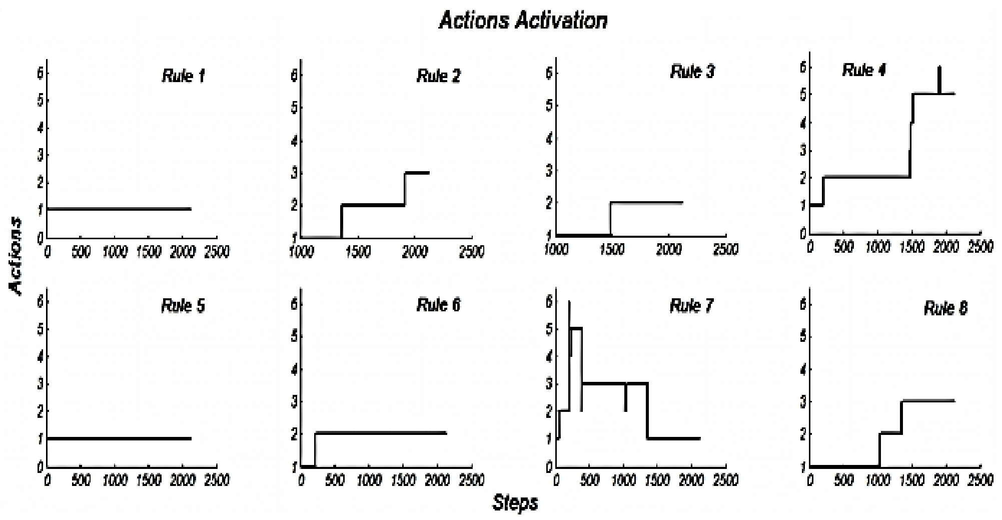


Figure 18. Choosing actions for fuzzy rules in a simple environment

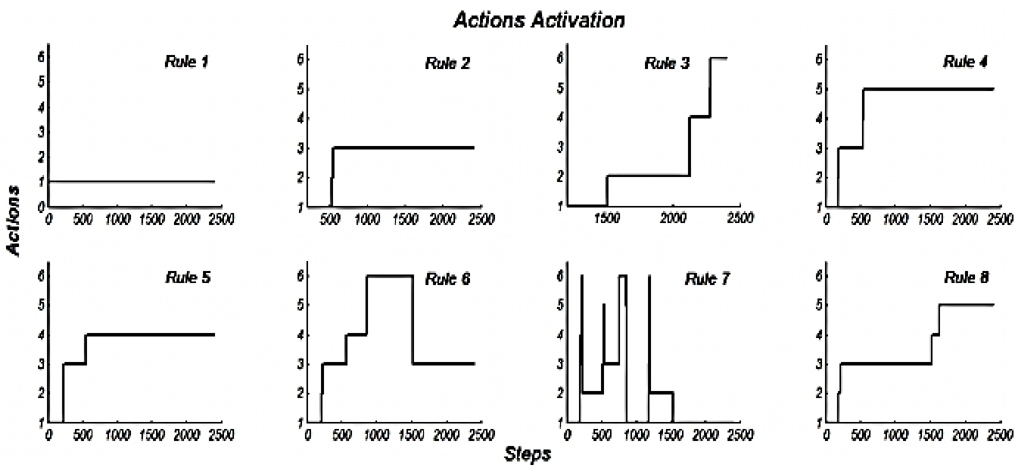


Figure 19. Choosing actions for fuzzy rules in a normal environment

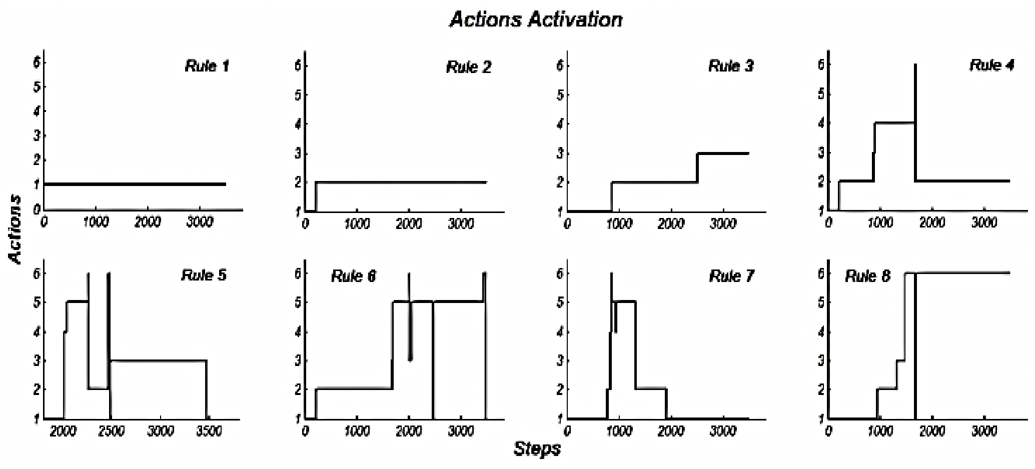


Figure 20. Choosing actions for fuzzy rules in a complex environment

In each step, the action with maximum q value is selected for each rule. The overall defuzzified steering angle is computed and the appropriate command is sent to motors, after the robot movement an immediate reinforcement signal is received. The maximum q value for each rule is updated with respect to the truth value of that rule. During a time interval, if the balance between immediately received punishments and rewards results in deterioration of a q value from its maximum position, another action is selected which has the current maximum q value. The maximum q value for a rule is updated according to the truth value of that rule. Therefore, rules with greater truth values are expected to experience more changes of action selection. Which rules are fired more depends on the environment and the encountered wall shapes. For instance, in sample environment referred in Figure 18, the first rule (fuzzy state) barely seems to be fired, since there observed no changes in action selection. On the other hand, the most changes of action selection happened for the seventh rule (fuzzy state). However, as expected, more fuzzy states get involved from simple to complex environments (Figures 18-20), because more types of wall shapes will be included.

7. CONCLUSION AND FUTURE WORK

A fuzzy Q-learning method was proposed to address autonomous wall tracking task of Khepera robot. Eight heuristic fuzzy if-then rules were designed based on human reasoning, and without any restrictive presumptions about the world. Incorporation of heuristic knowledge resulted in a few number of rules, which in turn led to a fast and easy implementation of planning process. Fuzzy inputs were outputs of Khepera infrared sensors while fuzzy outputs were the robot speed and steering angle. Applying Q-learning, the steering angle was tuned online and through interactions with the world. Simulated experiments were conducted on sixty environments with various degrees of complexity, including concave, convex, and polygon wall shapes. Simulated environments were completely unknown to the robot, and no training environment was considered. Each execution of the proposed algorithm was done with q values initialized to zero (unbiased q values), i.e. Q-learning was applied without any exploration phase. The obtained results presented an efficient performance of the robot in finding nearest walls from start points and staying in predefined margins while tracking walls. Simulated experiments confirm that the proposed method can efficiently operate and adapt itself to various wall contours without exploration phase. To highlight the effective role of Q-learning in online tuning of fuzzy inference system, the same experiments were repeated with a fuzzy rule base whose parameters were set completely offline. Comparisons between two decision making methods reveal the superiority of online tuned fuzzy inference system in terms of speed and accuracy.

Future work will be devoted to improving the autonomy of the proposed algorithm by making previously offline settings online, for instance online tuning of input membership functions, online selection of learning parameters through an optimization algorithm, and etc. The proposed algorithm will be experimented in simulated environments with more realistic features like three dimensional environments with: bumpy or slippery floors, obstacles with different reflection conditions, noisy sensed information, and etc.

8. AKNOWLEDGEMENTS

This work has been supported by Department of Electrical and Computer Engineering, Babol University of Technology. Special thanks go to founders of http://www.tstorm.se/projects/kiks and www.k-teams.com who have made an easy access for researchers to the open source KiKs software.

9. REFERENCES

1. Galindo, C., Fernández-Madrigal, J.-A. and González, J., "Improving Efficiency in Mobile Robot Task Planning Through World Abstraction", ***IEEE Transactions on Robotics and Automation***, Vol. 20, (2004).
2. Saffiotti, A., "The uses of fuzzy logic in autonomous robot navigation", ***Soft Computing***, Vol. 1, (1997), 180-197.
3. Maaref, H. and Barret, C., "Sensor-based fuzzy navigation of an autonomous mobile robot in an indoor environment", ***Control Engineering Practice,*** Vol. 8, (2000), 757-768.
4. Ming-liang, X. and Wen-bo, X., "Fuzzy Q-learning in continuous state and action space", ***The Journal of China Universities of Posts and Telecommunications***, Vol. 17, (2010), 100-109.
5. Glorennec, P.Y. and Jouffe, J., "A reinforcement learning method for an autonomous robot", ***Citeseer,*** Vol. 12, 1996.
6. Er, M.J. and Deng, C., "Online Tuning of Fuzzy Inference Systems Using Dynamic Fuzzy Q-Learning", ***IEEE Transactions on Systems, Man, and Cybernetics—Part B: Cybernetics***, Vol. 34, (2004).
7. Juang, C.F. and Hsu, C.H., "Reinforcement Ant Optimized Fuzzy Controller for Mobile-Robot Wall-Following Control", ***IEEE Transactions on Industrial Electronics***, Vol. 56, (2009), 3931-3940.
8. Glorennec, P.Y. and Jouffe, J., "Fuzzy Q-learning", ***6th IEEE Int. Conf. Fuzzy Systems,*** 1997.
9. Glorennec, P.Y., "Reinforcement Learning: an Overview", ESIT, Aachen, Germany, 2000.

10. Fatmi, A., Yahmadi, A. Al., Khriji, L. and Masmoudi, N., "A Fuzzy Logic Based Navigation of a Mobile Robot", ***World Academy of Science, Engineering and Technology,*** Vol. 22, (2006).
11. Seraji, H. and Howard, A., "Behavior-Based Robot Navigation on Challenging Terrain: A Fuzzy Logic Approach", ***IEEE Transactions on Robotics and Automation***, Vol. 18, (2002).
12. Karambakhsh, A., "Robot Navigation Algorithm to Wall Following Using Fuzzy Kalman Filter", ***9th IEEE International Conference on Control and Automation (ICCA)***, 2011.
13. Obayashi, M., Kuremoto, T. and Kobayashi, K., "A Self-Organized Fuzzy-Neuro Reinforcement Learning System for Continuous State Space for Autonomous Robots", ***CIMCA,*** 2008.
14. Li, C., Chen, P. and Li, Y., "Adaptive Behavior Design Based on FNN for the Mobile Robot", ***IEEE International Conference on Automation and Logistics***, 2009.
15. Kaelbling, L.P., Littman, M.L. and Moore, A.W., "Reinforcement Learning: A Survey", ***Journal of Artificial Intelligence Research,*** Vol. 4, (1996), 237-285.
16. Mondada, F., Franzi, E. and Lenne, P., "Mobile robot miniaturization: a tool for investigation in control algorithms", ***The International symposium on experimental robotics***, (1993), 336-341.
17. Benreguieg, M., Hoppenot, h., Maaref, H., Colle E. and Barret , C., "Fuzzy navigation strategy: application to two distinct autonomous mobile robots", ***Robotica***, Vol. 15, (1997), 609-615.
18. Kato, A. and Kamikawa, K., "Obstacle Avoidance Based on Approximate Reasoning for Mobile Robots", ***IEEE 'RSJ International Workshop on Intelligent Robots and Systems 'X9***, Vol. (1989).
19. Kruse, R., Gebhardt, J. and Klawonn, F., "Foundations of Fuzzy Systems", Wiley and Sons, (1994).
20. Sugeno, M., "Industrial applications of fuzzy control", Elsevier Scientific Publishing Company, (1985).
21. Storm, T., "KiKS is a Khepera Simulator user guide", Available from: http://www.tstorm.se/projects/kiks.

A Q-learning Based Continuous Tuning of Fuzzy Wall Tracking without Exploration

S. Valiollahi, R. Ghaderi, A. Ebrahimzadeh

Department of Electrical and Computer Engineering, Babol University of Technology, Babol- 7414871167.

P A P E R   I N F O

Paper history:
Received 19 Aprill 2012
Received in revised form 15 June 2012
Accepted 30 August 2012

Keywords:
Autonomous Navigation
Wall Tracking
Fuzzy Q-learning
Khepera Robot

چکیده

در این مقاله الگوریتمی ساده و کارآمد به منظور پیاده سازی رفتار تعقیب دیوار توسط ربات پیشنهاد شده است. ربات بایستی در محیط‌های ناشناخته حرکت کند، نزدیکترین دیوار را بیابد و آن را تنها بر پایه‌ی داده‌های حس شده‌ی محلی تعقیب کند. الگوریتم پیشنهادی از مزایای ترکیب منطق فازی و یادگیری Q برای تامین نیازهای هدایت خودمختار بهره می‌گیرد. ربات اطلاعات به دست آمده از جهان پیرامون را به مجموعه‌ای از حالت‌های فازی خلاصه می‌کند. برای هر حالت فازی تعدادی عمل پیشنهادی وجود دارند. حالت‌ها توسط قوانین اگر-آنگاه فازی، که با منطق انسان طرح شده‌اند، به عمل‌های نظیرشان مربوط می‌شوند. برای هر حالت، ربات بهترین عمل را با یادگیری Q و از طریق تجربیات برخط انتخاب می‌کند. هدف طراحی الگوریتم تعقیب دیواری است که بتواند خود را به طور کارآمد با شکل‌های مختلف دیوار در محیط‌های کاملا ناشناخته تطبیق دهد. یادگیری Q بدون مرحله اکتشافی به کار گرفته شده‌است، بدین معنی که محیط یادگیری وجود ندارد. نتایج آزمایشات روی ربات کپرا شبیه‌سازی شده کارآمد بودن روش پیشنهادی را در برخورد با شکل‌های مختلف دیوارها، از حالت ساده‌ی خط مستقیم گرفته تا حالات پیچیده‌ی مقعر، محدب و چندضلعی، تایید می‌نمایند. ربات به طور موفق دیوارها را تعقیب می‌نماید در حالیکه در محدوده تعیین شده باقی می‌ماند.

doi: *10.5829/idosi.ije.2012.25.04a.07*