An Intelligent System for Parking Trailer using Reinforcement Learning and Type 2 fuzzy Logic

[1]Morteza Sharafi,[2]Mohammad Reza Aalami,[3]Seyed Abolfazl Fakoorian

[1,2,3] Department of Electrical Engineering,Islamic Azad University Gonabad branch

(mortezasharafi@gmail.com,rezanis68@yahoo.com,a.fakoorian@yahoo.cm)

***Abstract*: In examples of reinforcement learning where state space is continuous, it seems impossible to use reference tables to store value-action .In these problems a method is required for value estimation for each state-action pair .The inputs to this estimation system are (characteristics of) state variables which reflect the status of agent in the environment .The system can be either linear of nonlinear .For each member in set of actions of an agent, there exists an estimation system which determines state value for the action .On the other hand, in most real world problems, just as the state space is continuous, so is the action space for an agent .In these cases, type 2 type 2 fuzzy systems may provide a useful solution in selection of final action from action space .In this paper we intend to combine reinforcement learning algorithm with fuzzified actions and state space along with a linear estimation system into an intelligent systems for parking Trailers in cases where both state and action spaces are continuous .Finally, the successful performance of the proposed algorithm is shown through simulations on trailer parking problem .**

Keyword: Reinforcement Learning; Type 2 type 2 fuzzy Systems; Trailer Parking Problem; SARSA Algorithm.

I. INTRODUCTION

Learning is the ability to improve behavior based on previous experiences and observations .Machine learning was proposed in artificial intelligence in order to create learner machines and therefore higher levels of flexibility and intelligence .If equipped with learning tools, a machine can constantly improve its performance and increase its efficiency .Reinforcement learning is a wide area of research in machine learning [1-22]. In reinforcement learning, agent attempts to improve its behavior based on trial and error and using its experiences .Agents can observe environment characteristics which form the state space for learner .Then, during each interval, agent influences the environment through some operations .Therefore, in the next interval, different inputs are fed to the agent based on their previous actions .In addition to these new inputs, each agent receives a reinforcement signal called "reward" which shows the appropriateness of the previous action .Reward can assume positive or negative values regarding the appropriateness of the previous action .In limited discrete spaces, value of each action for each state is stored in a reference table called Q Table [1, 2, 5]. Each row of this table represents a state while each column corresponds to an action .Agent makes decisions based on this table and its policies.

However, in larger and more complex environment where state space is continuous, it is virtually impossible to use reference tables .The problem becomes more serious when action state is also continuous and not contained to a limited number of actions. One way to generalize states in a continuous space and to produce continuous sets of actions is to employ type 2 type 2 fuzzy inference systems [23] In type 2 type 2 fuzzy environments, a number of membership functions or type 2 type 2 fuzzy sets are defined over the range of each variable; each variable is then described according to its membership to any of these type 2 type 2 fuzzy sets [6, 7, 18, 19] Type 2 type 2 fuzzy rules allow the system to perform human-like inference under uncertainty [23].

In this paper, we attempt to employ SARSA algorithm – a well-known algorithm in reinforcement learning [1] combined with type 2 fuzzy logic for cases where both state and action spaces are continuous .Then, we try to improve system efficiency using estimation systems for value-action functions .Finally, the system is applied to intelligent Trailer parking problem .The problem is defined as designing an automatic controller for parking a Trailer with forward and backward movements [24]. Here, we assume that Trailer driver is not an expert and his/her knowledge can not be used in control and simulations; rather, the intelligent agent should learn how to park the Trailer based on trial and error .

The paper is organized as follows. Section II describes the proposed algorithm; Trailer parking problem is introduced in Section III where the proposed algorithm is used to find a solution to the problem; Section IV provides simulation results; and finally, the paper concludes with Section V and some suggestions.

II. THE PROPOSED ALGORITHM

In this chapter we introduce type 2 fuzzy State-Action-Reward-State-Action (SARSA) algorithm with linear value estimation .First, we define type 2 fuzzy sets in the problem environment and over range of actions for agents .In this case, each rule R_i is related to a region in state space corresponding to an action in the action set .Each action in the rule has a value (Q) which is used by the agent to select a particular action .In other words, the table Q is extended in a way that each row points to a set of states instead of only one state; as well, each column represent as set of actions .These sets are type 2 fuzzy sets which cover the whole state and action space .This leads to definition of one element as $Q(\tilde{s}_i, \tilde{a}_j)$ where both $\tilde{s}_i$ and $\tilde{a}_j$ are type 2 fuzzy sets .

As mentioned earlier about reinforcement learning algorithms, this element represents a space-action value which is estimated by a linear estimator in the proposed algorithm using weighted combination of state variables . Each row of the table is in form of a type 2 fuzzy rule R_i .

if x_1 *is* A_1^i *and* x_2 *is* A_2^i *and ... and* x_n *is* A_n^i *then*
a_1 *is* B^1 *with* $w_0^i + w_1^i x_1 + ... + w_n^i x_n$ *or*
a_2 *is* B^2 *with* $w_0^i + w_1^i x_1 + ... + w_n^i x_n$ *or*
$\vdots$
a_m *is* B^m *with* $w_0^i + w_1^i x_1 + ... + w_n^i x_n$

And:

$$Q(S_i, B^j) = w_0^i + w_1^i x_1 + ... + w_n^i x_n,\ S_i = < A_1^i, A_2^i, ..., A_n^i >$$

Where <x_1, $x2$, ..., x_n> are state variables; A_j^i is membership function for the j^{th} variable in the state space in the i^{th} rule; and B^m is the membership function for the m^{th} action .Furthermore, the element $Q(S_i, B^j)$ corresponds to the state S_i and the action B_j which are both type 2 fuzzy sets .According to what we have so far, the number of rules equals the multiplication of number of membership functions for all states .For real values, and for the input vector $\vec{x} = (x_1, ..., x_n)^T$ the output *y* at the Takagi-Sugeno type 2 fuzzy system using Mamdani multiplication inference engine for k rules will be as follows :

$$y = \frac{\sum_{i=1}^{k} \Phi_i(x).\bar{y}_i(x)}{\sum_{i=1}^{k} \Phi_i(x)}$$

Where Φ_i is

$$\Phi_i(x) = \prod_{j=1}^{n} \mu_{ij}(x_j)$$

Where μ_{ij} refers to the j^{th} membership function in the i^{th} rule. $\bar{y}_i(x)$ is the center of the selected set of actions in the i^{th} rule sometimes referred to as i^{th} local action .

In the proposed algorithm state-action value is determined as follows :

$$Q(S_i, B^j) = w_0^i + w_1^i x_1 + ... + w_n^i x_n$$

As mentioned before, B^j represents the membership function for the action selected based on the agent's policies in the i^{th} rule .Initially, all w_i's are zero .Suppose that the agent performs and action and takes the state x_t .According to x_i's membership function and according to the action selected based on the rule and policies (local action), (1) determines the basic action selected by the agent .Equation (4) presents how the basic action is chosen:

$$a_t^G = \sum a_t^i \phi_i(x_t)$$

Where a_t^G is the basic action at the step *t* and a_t^i represents the local action at the step *t* chosen according to the agent's policy in the rule R_i .In addition, $\phi_i(x_t)$ is a basic type 2 fuzzy function defined as below:

$$\phi_i(x) = \frac{\Phi_i(x)}{\sum_{i=1}^{k} \Phi_i(x)}$$

$Q(x_t, a_t^G)$ Which shows the value of the state x_i for the action a_t^G is calculated as follows :

$$Q(x_t, a_t^G) = \sum Q(s_i, a_t^i) \phi_i(x_t)$$

According to SARSA, the value of each state must be updated now .The update equations are

$$T\arg et\, Q(s_i, a_t^i) = Q(s_i, a_t^i) + \alpha \tilde{\varepsilon}_{t+1} \phi_i(x_t)$$
$$\varepsilon_{t+1} = r_{t+1} + \gamma Q(x_{t+1}, a_{t+1}^G) - Q(x_t, a_t^G)$$

Now $Q(s_i, a_t^i)$ must be updated for *i=1,2,...,k* .For doing so, an MSE error is defined as

$$e(s_i, a_t^i) = t\arg et\ Q(s_i, a_t^i) - Q(s_i, a_t^i)$$
$$E(s_i, a_t^i) = 0.5 e(s_i, a_t^i)^2$$

Now, we can update $Q(s_i, a_t^i)$ parameters based on this error and using optimization methods .Such optimization algorithm may be of steepest descent type, LRSE, or intelligent optimization methods .The algorithm will be described in details in the following sections .For each variable involved in creating a state, several type 2 fuzzy sets are defined over its range .Then, premises for type 2 fuzzy rules are created based on these type 2 fuzzy sets .The weight vector $w_{k\times(n+1)}$ is then given initial value . *k* is the number of rules and *n* represents the number of state variable .It is clear that for *m* actions there will be *m* value of $w_{k\times(n+1)}$. The following steps are repeated for each episode :

t=0 and the process begins with the initial state x_0 . According to the agent's policies, a local action is selected at each rule .Then, the basic action a_0^G is selected according to (4).

For the step *t* in the episode do the following:

2-1 Perform the action a_t^G ; go to the state x_{t+1};and receive the reward r_{t+1} .

2-2 -Use (6) to determine the state-action values $Q(x_t, a_t^G)$ and $Q(x_{t+1}, a_{t+1}^G)$ for doing so, you need a_{t+1}^i which is determined according to the agent's policies .

2-3 Use (7) to determine the target for each action selected at each rule .

2-4 derive an error function in form of (8) for each action selected at each rule and update *w* using an optimization algorithm .

2-5 $x_t = x_{t+1}$, $a_t^i = a_{t+1}^i$

End, if x_i is a final state; otherwise $t = t + 1$ and go to 2 [25].

Fuzzy Sarsa Learning

Fuzzy Sarsa learning (FSL) is a Fuzzy method based on Sarsa learning [11], for environments with continuous operation and state.

Sarsa methods show the value of the act ofa in state of s as $Q(s,a)$ and updated formula is as (1).

$$Q(s_t,a_t) \leftarrow Q(s_t,a_t) + \alpha_t [r_{t+1} + \gamma Q(s_{t+1},a_{t+1}) - Q(s_t,a_t)] \quad (1)$$

Where α is the learning rate γ forgetting rate and r_{t+1} is the award in which the environment after the act of a_t give to factor in state of s_t.

A zero-order TSK Fuzzy system with R rules have the following form [3]:

if x_1 is L_{i1} and …and x_n is L_{in} R_i :

Then (a_{i1} with value w^{i1}) or … or ((a_{in} with value w^{in})

s= $x_1 * .. * x_n$ is n-dimensional input vector.

$L_i = L_{i1} * ... * L_{in}$ are set of Input Fuzzy membership functions , m is a discrete set of actions for each rule, a_{ij} and w^{ij} are the j-th Function candidate and Approximate values for the i-th rule, respectively.FSL aims to regulate w^{ij} online to get the best policy.

The possibility of choosing is obtained the j-th action in i-th rule in state of s_t based on the policy of soft max [3].

$$p(a_{ij}) = \frac{\exp(\mu_i(s_t) w^{ij}/T)}{\sum_{k=1}^{m} \exp(\mu_i(s_t) w^{ik}/T)} \quad (2)$$

$\mu_i(s_t)$ Is Fire intensity normalized of i-th rule in state of s_t and T> 0 is the temperature factor.

The actions chosen by each rule and its value with a_{ii^+} and w^{ii^+} are shown, respectively, and their values in (2) and (3) is calculated as [1, 9]:

$$a_t(s_t) = \sum_{i=1}^{R} \mu_i(s_t) a_{ii^+} \quad (3)$$

$$\hat{Q}_t(s_t,a_t) = \sum_{i=1}^{R} \mu_i(s_t) w_t^{ii^+} \quad (4)$$

Therefore the total weighted discrete function is selected by the rules.

Performing the act of a_t goes the environment to next state of s_{t+1} and the factor take the award signal r_{t+1}. The next action a_{t+1} calculated based on current weighted w_t.

Thus The i-th Rule weights updated by following formula [3]:

$$\Delta w_{t+1}^{ij} = \begin{cases} \alpha_{t+1} \times \Delta\hat{Q}_t(s_t,a_t) \times \mu_i(s_t) & if \quad j = i^+ \\ 0 & \text{otherwise} \end{cases} \quad (5)$$

which $\Delta\hat{Q}$ Is the error function that obtained by (6):

$$\Delta\hat{Q}_t(s_t,a_t) = r_{t+1} + \gamma \hat{Q}_t(s_{t+1},a_{t+1}) - \hat{Q}_t(s_t,a_t) \quad (6)$$

2 - FSL multi-agent (MAFSL)

In this paper we follow the common practice for operating multi-agent systems. The main ideas taken from [3] and [4].

Considering that the maximum value of the function mode in case is the same ($V(s) = \max_i Q(s,a_i)$) the following updated formula for V and Q states as follow:

$$Q^k(s^k,a_j^k) = \sum_{i=1}^{R} \mu_i(s^k) * wq_{ii^+}^k$$
$$\Delta Wq_{i^+jt}^k = \Delta Q^k(s^k,a_{i^+}^k)_{t+1} = \alpha * (r_{t+1} + \delta V^k(s_{t+1}) + Q_t^k(s_t,a_{i^+}^k)) * \mu_i(s^k) \quad (7)$$
$$V^k(s^j) = \sum_{i=1}^{R} \mu_i * Wv_i^k$$
$$\Delta Wv_{it}^k = \Delta V_{it}^k = \alpha * (r_{t+1} + \delta V^k(s_{t+1}) - V^k(s_t))) * \mu_i(s^k) \quad (8)$$

(I Rule Number, j Number of operating, k One of the possible acts , t represents the time step , Wv Inferior to the rules of system related to V and Wq Inferior to the rules of system related to Q, r amount of award and a_{i^+} are chosen act of rulei-th.)

Lower values of V and Q each have a separate phase. Input system is as follows.

Input of fuzzy systems are state-space dimensions. In the training phase, the rules are set lower. rules areZero-order TSK-type.

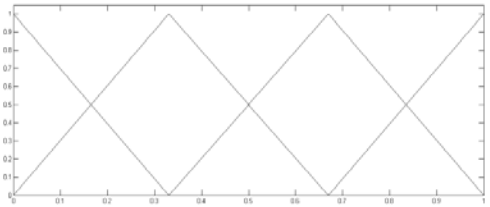

Figure 1: fuzzy Membership functions

The rules of the system multiplied by the number of input membership functions. Each represents a fuzzy rule.

Q inferior to the rules relating to systems containing the integer values corresponding to each fuzzy state implies that

the weight of each operation. Each set of operating rules and is independent of other factors.

Inferior to the rules of the system V of integers, each of which corresponds to the values of the fuzzy rule is expressed.

By partitioning Mode and calculating the fuzzy mode, fuzzy Q and V for the same position entirely solve the problem of curse of dimension and the number of dimensions even Compared to [4] are also significantly.

One of the issues that have a significant impact in decreasing sizes, the proper definition of the state. In the next section we will elaborate on this.

The following table shows the pseudo-code description of the algorithm is given.

This table shows Pseudo-code algorithm MAFSL

For any agent k we have j same action:
1.Initialize Qij and Vi by Zero
2.loop untile Q values converge
3.observe sate S1
4.select action from each rule by using (2)
5.choose final action by (6) equation
6.observe S2, reward and other agents actions
7.update Q and V use 7 and 8 equations
8.end loop

III. TRAILER PARKING PROBLEM

A. Problem Definition

Designing an optimal pathway for backward movement of a trailer through a number of fixed and moving obstacles is among most complicated problems in engineering. Factors such as type, shape, and rate of movement as well as time limitations for achieving the target dock may introduce further complication into the problem.

Backward movement of a trailer on a dock is a nonlinear control problem. Using the conventional control methods, a mathematical model for the system can be obtained, and then nonlinear control theory may be employed to design of the controller. An alternative to this is to design a controller which simulates human behavior. The latter is used in the present paper. We assume that an experience trailer driver is available; in addition, we can measure different positions of the trailer and corresponding driver's actions to move the trailer backward. Figure 1 shows the trailer and the loading (parking) dock.

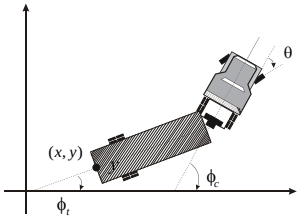

Figure 1. state variables involved in intelligent backward movement of the trailer

The trailer is controlled by changing θ. Only backward movement is allowed here. In each step, the trailer moves We assume that a sufficient space is present between the trailer and parking spot, and therefore, the vertical position y is not required as a state variable for our purpose.

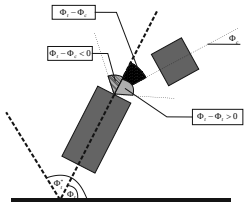

Figure 2. Relationship between truck and trailer angles

Problem constraints:

- The trailer has a constant velocity of V.
- The length of the trailer is $L_c + L_s$.

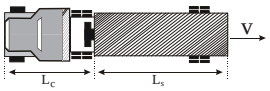

Figure 3. constraints of the trailer parking problem

IV. DEFINING THE TYPE 2 FUZZY SETS

TABLE I. TYPE 2 FUZZY RULES DEFINED FOR THE CONTROL PROBLEM

Details of Plant			
State	x, y		Coordinates of the center rear of the trailer
	Φ_t		Angle of trailer with *x*-axis
	Φ_c		Angle of the cab with *x*-axis
Control	θ		Steering angle of the front wheels relative to cab orientation
Constraints	$x > 0$		The loading dock is at $x = 0$
	$\lvert\Phi_t - \Phi_s\rvert \le 90$		The angle between the cab and trailer can't exceed 90°
	$-70 \le \theta \le 70$		Limit of steering of the front wheel
Equations of Motion	$A = r \times \cos(\theta)$ $B = A \times \cos(\Phi_c[t] - \Phi_t[t])$ $x[t+1] = x[t] - B \times \cos(\Phi_t[t])$ $y[t+1] = y[t] - B \times \sin(\Phi_t[t])$ $\Phi_c[t+1] = \Phi_c[t] + \arcsin(\frac{r \times \sin(\theta)}{L_s + L_c})$ $\Phi_t[t+1] = \Phi_t[t] - \arcsin(\frac{A \times \sin(\Phi_c[t] - \Phi_t[t])}{L_s})$		
	$\Phi_c[t+1]$		Is then adjusted to respect the constraint on $\Phi_t - \Phi_s$
Parameters	r	3m	distance front wheel moves per time step
	L_s	14m	length of the trailer, from rear to pivot
	L_c	6m	length of the cab, from pivot to axle

Figures 5 through 8 show the membership functions defined in MATLAB:

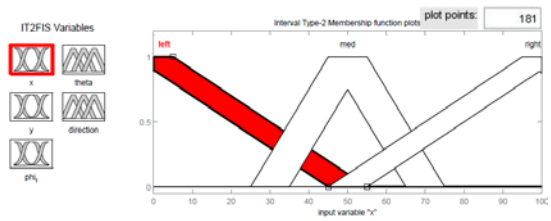

Figure 4. membership functions for position of the trailer

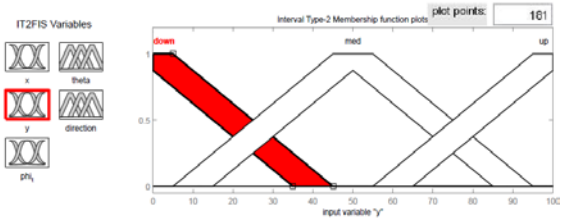

Figure 5. membership functions for the angle $\alpha = \Phi_t$ (degrees)

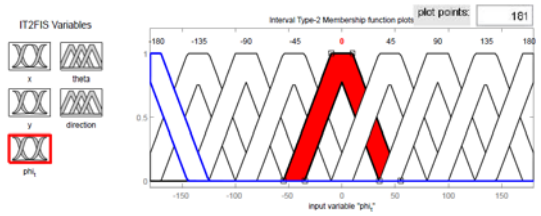

Figure 6. membership functions for the angle $\beta = \Phi_c$

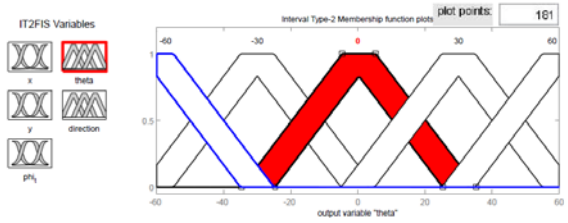

Figure 7. membership functions for the angle θ

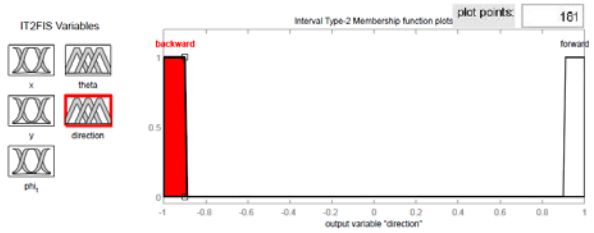

Figure 8. membership functions for the direction

type 2 fuzzy sets for states and actions; variable x in state space; variable α in state space; variable β in state space and steer wheel angle θ action .

3-2 Trailer parking problem as a reinforcement learning problem

The followings are the important notes to consider in the proposed algorithm

Fuzzification of actions and states

In this problem, each state is represented by $< x, \alpha, \beta >$ .To fuzzify the states, five type 2 fuzzy sets are defined for x while seven type 2 fuzzy sets are defined for α and three type 2 fuzzy sets are defined for β . Seven type 2 fuzzy sets are considered for steer wheel angle as the action .

A. *Determining start points for episodes*

Start points must be uniformly distributed over state space .To achieve this, x is varied between 0 and 100 with the step 1; this results in 1000 values for x .For each value of x, α varies between -90 and 270 with the step 10; this produces 36 values for α and β varies between -90 and 90 with the step 10; this produces 18 values for β . Therefore, there are 64800 start points for episodes .

B. *Assignment of rewards*

For rewarding, two ranges are considered for the variables x and *angle* .The allowable range initially includes the whole range i.e.[0, 100] for x , [-90, 270]and [-90,90] for *angles* .Then, the ranges are reduced toward the target proportional to the increase in number of episodes .If the agent goes to a predetermined range after performing an action, it will receive a positive reward which is proportional to the size of this range or the episode number; if the agent exit the range, however, receives the same reward but as a negative amount .The maximum reward (1) is that of the final episode, while the minimum reward is -1 .There are two exceptions in rewarding at each episode .Reaching the target results in the grand prize (i.e.100) .Exiting the allowed range brings a -100 punishment .If the agent goes to the determined range as a result of an action, the episode will end .The procedure continues until the agent goes to the determined range or exit the allowable range.

V. DETERMINING THE PARAMETERS AND SIMULATION RESULTS

In simulations, selected values for α, γ, and training rate in steepest descent are 0.5, 0.9, and 0.01 respectively . Maximum number of actions in each episode is 100 .Table I shows mean values for 15 runs with different number of episodes .The variance values are almost zero. The first column shows the number of episodes, while the second column lists number of states which resulted into final state as a percentage of 720 states .For testing, we created 100 random states in the allowable range and test them using the Q table .The results are shown in the third column .There are number of states in the state space which do not result in target state and produce results outside the allowable range through a number of actions .The forth and the fifth column show the simulation results after elimination of these states (a small tolerance is acceptable in reaching the target; $\Delta\varphi=5^\circ$, $\Delta x=0.5$) .

TABLE II. PERCENTAGE OF STATES, OUT OF TOTAL TEST STATES, RESULTING IN TARGET STATE.

Number of episode	1	2	3	4
360000	80	75	93	90
720000	90	83	99	98
1440000	98	96	99	99
2880000	98	98	100	100

VI. CONCLUSION

To test the proposed navigation method, simulation is performed using MATLAB in different conditions for the position of the trailer and moving and fixed obstacles. We tried to improve the trailer movement through changing or weighting the type 2 fuzzy rules. Due to the low noise in the installed sensors in comparison to the minimum range of control inputs (maximum of 1 to 2 cm compared to 1 m), there is no need to consider the measurement noise while modeling the controller. In the following lines, the movement of the trailer while facing obstacles moving with constant velocity is reviewed. The sign (*) is used to denote the movement and speed of the trailer. The space between the stars represents the changes in the trailer speed. The proposed algorithm is applied for a situation consisting of two fixed obstacles and one moving with constant speed (however, there is no limitation on the number of fixed and moving obstacles). The algorithm may be applied for a greater number of obstacles moving with different velocities. As it can be seen, the type 2 fuzzy controller operates well in variety of conditions. The following shows the program results for different conditions. Number of moving and fixed can be increased in the simulation. Figure 9 shows the results obtained from MATLAB simulation. In this paper, we proposed a method for learning in complex environments where both states and action spaces are continuous .The algorithm first fuzzifies the value-action table; then the desired action is determined based on the state of the input and type 2 fuzzy inference; finally, type 2 fuzzy SARA and steepest descent are used to update weights in the table .An important feature of this algorithm is adaptability of its type 2 fuzzy table which results in high efficiency as seen in simulation results .In addition, the algorithm does not need to learn which decision in the decision making process is the optimum decision; rather it optimizes the decision making through trail and error .This makes the system independent of expert knowledge for decision making in complex systems .

Finally, the proposed algorithm was used for finding a solution to Trailer parking problem using reinforcement learning for the first time .An important issue in solving the problem is how to reward actions which was addressed by a heuristic process. In the final section, we presented simulation results to prove the proposed method successful.

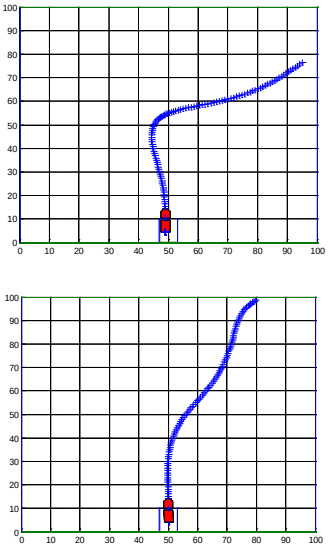

Figure 9. Trajectories determined for different initial states

REFERENCES

[1] R. Sutton and A. G. Barto, *Reinforcement Learning: An Introduction*. Cambridge, MA: MIT Press, 1998.

[2] L. P. Kaelbling, M. L. Littman and A. W. Moore, *Reinforcement learning: A survey*, J. Artif. Intell. Res., vol. 4, pp. 237–287, 1996.

[3] D. P. Bertsekas and J. N. Tsitsiklis, *Neuro-Dynamic Programming*. Belmont, MA: Athena Scientific, 1996.

[4] R. Sutton, *Learning to predict by the methods of temporal difference*, Mach. Learn., vol. 3, no. 1, pp. 9–44, Aug. 1988.

[5] C. Watkins and P. Dayan, *Q-learning*, Mach. Learn, vol. 8, no. 3/4, pp. 279–292, 1992.

[6] D. Vengerov, N. Bambos, and H. Berenji, *A type 2 fuzzy reinforcement learning approach to power control in wireless transmitters*, IEEE Trans.Syst., Man, Cybern. B, Cybern., vol. 35, no. 4, pp. 768–778, Aug. 2005.

[7] H. R. Beom and H. S. Cho, *A sensor-based navigation for a mobile robot using type 2 fuzzy logic and reinforcement learning*, IEEE Trans. Syst.,Man, Cybern., vol. 25, no. 3, pp. 464–477, Mar. 1995.

[8] C. I. Connolly, *Harmonic functions and collision probabilities*, Int. J.Rob. Res., vol. 16, no. 4, pp. 497–507, Aug. 1997.

[9] W. D. Smart and L. P. Kaelbling, *Effective reinforcement learning for mobile robots*, in Proc. IEEE Int. Conf. Robot. Autom., 2002,pp. 3404–3410.

[10] T. Kondo and K. Ito, *A reinforcement learning with evolutionary state recruitment strategy for autonomous mobile robots control*, Robot.Auton. Syst., vol. 46, no. 2, pp. 111–124, Feb. 2004.

[11] M. Wiering and J. Schmidhuber, *HQ-learning*, Adapt. Behav, vol. 6, no. 2, pp. 219–246, 1997.

[12] A. G. Barto and S. Mahanevan, *Recent advances in hierarchical reinforcement learning*, Discret. Event Dyn. Syst.: Theory Appl., vol. 13, no.4, pp. 41–77, Oct. 2003.

[13] R. Sutton, D. Precup, and S. Singh, *Between MDPs and semi-MDPs: A framework for temporal abstraction in reinforcement learning*, Artif.Intell., vol. 112, no. 1, pp. 181–211, Aug. 1999.

[14] T. G. Dietterich, *Hierarchical reinforcement learning with the MAXQ value function decomposition*, J. Artif. Intell. Res., vol. 13, pp. 227–303, 2000.

[15] G. Theocharous, *Hierarchical learning and planning in partially observable Markov decision processes*, Ph.D. dissertation, Michigan State Univ., East Lansing, MI, 2002.

[16] A. J. Smith, *Applications of the self-organising map to reinforcement learning*, Neural Netw., vol. 15, no. 8/9, pp. 1107–1124, Oct. 2002.

[17] P. Y. Glorennec and L. Jouffe, *Type 2 fuzzy Q-learning*, in Proc. 6th IEEE Int. Conf. Type 2 fuzzy Syst., 1997, pp. 659–662.

[18] S. G. Tzafestas and G. G. Rigatos, *Type 2 fuzzy reinforcement learning control for compliance tasks of robotic manipulators*, IEEE Trans. Syst.,Man, Cybern. B, Cybern., vol. 32, no. 1, pp. 107–113, Feb. 2002.

[19] M. J. Er and C. Deng, *Online tuning of type 2 fuzzy inference systems using dynamic type 2 fuzzy Q-learning*, IEEE Trans. Syst., Man, Cybern. B,Cybern., vol. 34, no. 3, pp. 1478–1489, Jun. 2004.

[20] C. L. Chen, H. X. Li, and D. Y. Dong, *Hybrid control for Robot Navigation: A hierarchical Q-learning algorithm*, IEEE Robot. Autom. Mag., vol. 15, no. 2, pp. 37–47, Jun. 2008.

[21] S. Whiteson and P. Stone, *Evolutionary function approximation for reinforcement learning*, J. Mach. Learn. Res., vol. 7, pp. 877–917, Dec.2006.

[22] M. Kaya and R. Alhajj, *A novel approach to multiagent reinforcement learning: Utilizing OLAP mining in the learning process*, IEEE Trans.Syst., Man, Cybern. C, Appl. Rev., vol. 35, no. 4, pp. 582–590, Nov.2005.

[23] J. S. R. Jang, C. T. Sun and E. Mizutani, *Neuro-Type 2 fuzzy and Soft Computing*, Englewood Cliffs, NJ, Prentice-Hall, 1997.

[24] Wang, Lie-Xin, A Course in type 2 fuzzy system and control, Prentice Hall PTR,pp157-160,1997

[25]Zahra Moeini, Vahid seyyedi ghomshe, Mohammad Teshne lab, *10th Iranian Type 2 fuzzy System Conference* pp.221-226 July