International Journal of Soft Computing 2 (1): 138-145, 2007
© Medwell Journals, 2007

A Learning Unidirectional Linear Response Fuzzy Controller with Golden Section Search

[1]Guofeng Tang, [1]Catherine Vairappan,[2]Qiping Cao and [1]Zheng Tang
[1]Faculty of Engineering, Toyama University, Gofuku 3190, Toyama shi, 930-8555, Japan
[2]Tateyama Institute of System, Toyama-shi, 930-0016, Japan

Abstract: This study proposes a learning Undirectional Linear Response (ULR) fuzzy controller using a golden section search. This fuzzy controller can be realized with ULR elements which can be implemented simply by a diode or a drain gate connected MOS transistor. This understanding of ULR system is further extended by incorporating the Golden Section Search (GSS) technique of finding the minimum point using the bracket method. We define GSS by evaluating the function at a chosen point in the larger of the two intervals. The midpoint is then replaced and the previous is now chosen to be the new endpoint. This is repeated until the width achieves the desired tolerance and the new minimum point is found. The simulation is carried out on the inverted pendulum problem based on the basic functions of a fuzzy controller that includes membership function, minimum and defuzzification functions of the ULR elements.

Key words: Golden section search, undirectional linear response, fuzzy controller

INTRODUCTION

As reported by the one of the founders of "Fuzzy Set Theory" in 1965, Prof. L. A. Zadeh of University of California, Berkeley, fuzzy theory is all about vagueness and uncertainty. Based on this fundamental theory, we are able to use nonprecise, ill-defined concepts and yet to work with these in a mathematically strict sense[1]. Various researches carried out by Kickert[2], Rutherford[3], King[4] and Li[5] all suggest in the instance of when the strategy applied by the human operator is vague and qualitatively described then the use of fuzzy theory is rather preferred.

It was the development of Fuzzy Controllers(FC) by Mamdani[6] that boosted the utilization of FC in various fields and encouraged many researchers to further experiment it. Since then FC has evolved from non learning, adaptive or non adaptive to learning FC whereby the weakness of non learning systems in response to disturbances or inability to provide a feedback control has since been improved with the learning FC. Some of these learning algorithms which have been used include Evolutionary Algorithm[7], Reinforcement Learning[8] and Supervised Learning[9].

However some of these proposed learning algorithms have proved to be not so efficient such as in the case of evolutionary method where learning process is rather long or heavy learning phase in reinforcement method since gradient information is not provided explicitly. Other problem often related to this also includes ill behavior which circumnavigate around an obstacle closely and slowly, premature convergence or trap situation[10] which continues to undermine the credibility of these algorithms and also difficulty in hardware realization. Although supervised learning as so far proven to have the advantage of faster convergence but it has the problem of insufficient training data which could result in incomplete fuzzy rules.

Unidirectional linear response is another learning algorithm and it trains many typical digital and analog as well as continuous problems[11]. ULR as previously been reported has being a powerful computing properties in close correspondence with earlier stochastic model based on McCulloh-Pitts neuron and graded neuron model based on sigmoid input-output relation[11]. This work was further continued with reinforcement learning to the adaptive fuzzy controller but with a simpler hardware implementation as we have proposed in ULR fuzzy controller[12]. When tested on cart-pole balancing problem it showed good control performance. On the contrary, the disadvantage of this system was that learning algorithm based on the back propagation need to have a derivative and the derivative has to be a zero.

We therefore tried to overcome the above mentioned problem by introducing a new learning method of Golden Section Search (GSS). This is a robust linear search method of locating an interval where the minimum point

Corresponding Author: Guofeng Tang, Faculty of Engineering, Toyama University, Gofuku 3190, Toyama shi, 930-8555, Japan

occurs between two points. In this routine the size of the interval containing the minimum reduces as the search takes places. For subsequent new step, two new points located in the interval are found whereby one section of the interval is discarded while a new interior point is placed within the new interval. This procedure is repeated until the width achieves a desired tolerance specified by the user.

LEARNING ALGORITHM

Unidirectional Linear Response (ULR) fuzzy controller: Unlike the typical nodes in neural network which are primarily sigmoid or hard limiters, ULR works as well as the traditional nodes with some extra features. The advantage of ULR is that it is able to not only retain the significant behaviors of the traditional models, but gives powerful capability of the analog and continuous signal processing[11].

The above figure can be defined as:

$$f(u) = \begin{cases} u & u > 0 \\ 0 & u \le 0 \end{cases} \tag{1}$$

$$u = \sum_{i=1}^{n} w_i x_i - \theta \tag{2}$$

Figure1 shows inputs to node N when x_1, x_2 .... x_n is defined as the input with weight of w_1, w_2 .... w_n and threshold is set at θ. This representation of unidirectional can be achieved either in a single bipolar or a diode connected MOS transistor. However when one or more layers are present connecting hidden units or nodes, the ULR multilayer network can be represented as in Fig. 2. The figure below shows a ULR multilayer fuzzy controller network with two inputs of x_1 and x_2, four fuzzy if-then rules, local mean-of-maximum(LMOM) and an output of F[13]. It consists of four layers:

Layer 1: The nodes in this layer share the membership function of $\mu_v(x)$ of V and V can be defined as linguistic label for the node function. V also defines the degree to which the given *x* satisfies it. The membership function used here is the triangular function Fig. 3 and 4.

$$U_{C_V S_{VL} S_{VR}}(x) = \begin{cases} 1 - |x - C_V| / S_{VR} & x \in [C_V, C_V + S_{VR}] \\ 1 - |x - C_V| / S_{VL} & x \in [C_V - S_{VL}, C_V] \\ 0 & \text{otherwise} \end{cases} \tag{3}$$

Where: C_V: center of triangle S_{VR}: right of the center S_{VL}: left of the center

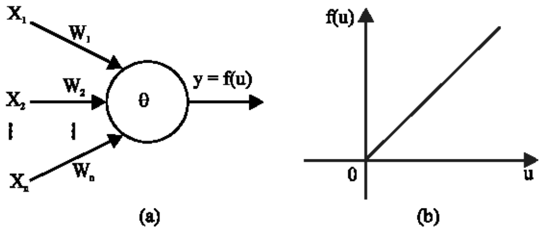


Fig. 1: Unidirectional linear response model

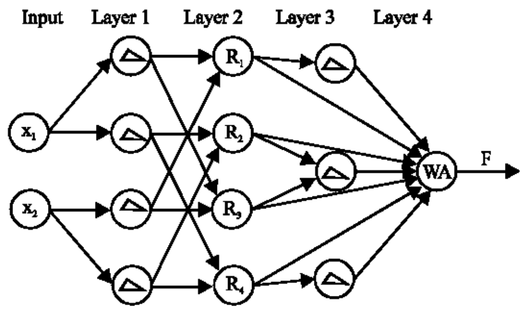


Fig. 2: A ULR multilayer network

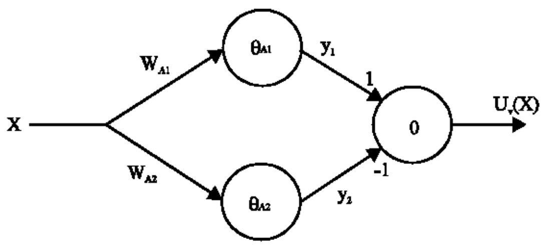


Fig. 3: Unidirectional linear response model

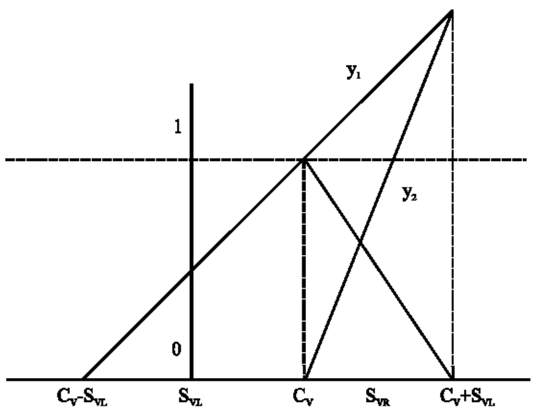


Fig. 4: Membership function

Weights are in the center where else the threshold of the network is spread about URL triangular. As both values of this systems change, the membership function too varies accordingly resulting in different forms of membership function. The triangular membership function used in the two-layer ULR network can berepresented as below:

$$w_{A1} = \frac{1}{S_{VL}} \tag{4}$$

$$w_{A2} = \frac{S_{VL} + S_{VR}}{S_{VL} \cdot S_{VR}} \tag{5}$$

$$\theta_{A1} = -\frac{S_{VL} - C_V}{S_{VL}} \tag{6}$$

$$\theta_{A2} = \frac{C_V \cdot (S_{VL} + S_{VR})}{S_{VL} \cdot S_{VR}} \tag{7}$$

Layer 2: Inputs for this layer originate from the previous layer. Besides performing a minimum operation that has been proposed to have a continuous differentiable softmin operation, it is also involved in the if part of the rule[13]. Figure 5a shows how the network achieves the minimum operation in the case of $u_{V1} = u_{V2}$ when the threshold for the network is zero and the outputs from layer 1 are:

$$f_1(u_{V1} - u_{V2}) = 0 \tag{8}$$

$$f_2(u_{V1}) = u_{V1} \tag{9}$$

$$\text{output } Z_r = f(0 + u_{V1}) = u_{V1} \tag{10}$$

For the case of $u_{V1} = u_{V2}$:

$$f_1(u_{V1} - u_{V2}) = u_{V1} - u_{V2} \tag{11}$$

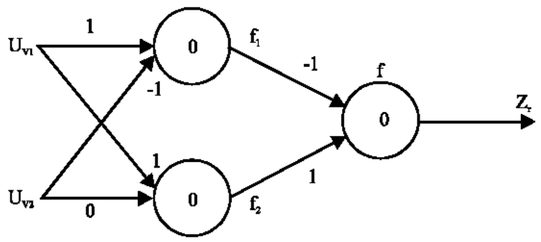
$$f_2(u_{V1}) = u_{V1} \tag{12}$$

$$\text{output } Z_r = f(-(u_{V1} - u_{V2}) + u_{V1}) = u_{V2} \tag{13}$$

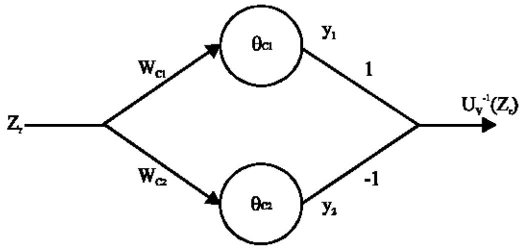
Layer 3: This layer uses a triangular membership function. The specified LMOM method is as such:

$$U^{-1}_{C_V,S_{VL},S_{VR}}(Z_r) = C_V + \frac{1}{2}(S_{VR} - S_{VL})(1 - Z_r) \tag{14}$$

The output is a limited value of $\mu_v^{-1}(z_r \to 0^*)$ and the membership function is monotic with $\mu_v^{-1}(z_r)$ as the



(a) Minimum operations



(b) Defuzzification

Fig. 5: ULR network

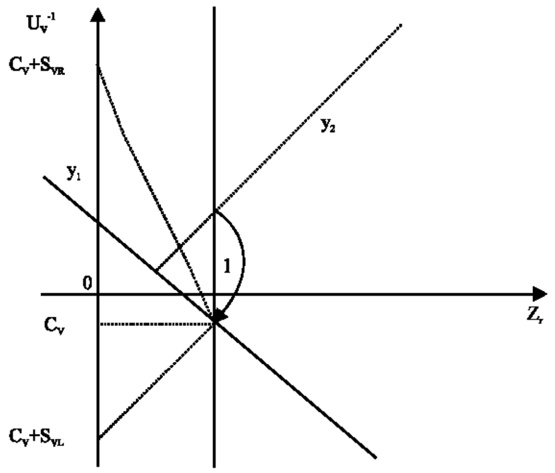


Fig. 6: Defuzzification network response

mathematical inverse. When the membership function of defuzzification used is the triangular function, the input output relationship can be represented as y = ax + b. Therefore it can be implemented by the network as shown in Fig. 5b.

Figure 6 shows the input-output relation of the defuzzification is the inverse of the membership function whereby the output can either be positive or negative. The output will always remain positive as for the URL. However the output of the defuzzification will be positive if $y_1 > 0$ and $y_2 = 0$ and negative if $y_1 = 0$ and $y_2 > 0$ by multiplying y_2 with -1. The response of the defuzzification network is expressed by pulsing y_1 with $-y_2$ while the parameters of the system is determined with $Z_r = 0$ and 1 Eq. 13.

The following conditions apply when determining the values of w_{c1}, θ_{c1} for y_1:

$$Z_r = 0U_V^{-1}(0) = \frac{1}{2}((C_V + S_{VR})(C_V - S_{VL})) \quad (15)$$

$$Z_r = 1 \quad U_V^{-1}(1) = C_V \quad (16)$$

w_{c2} and θ_{c2} for y_2 are inverse of w_{c1} and θ_{c1}:

$$w_{C1} = -\frac{S_{VR} - S_{VL}}{2} \quad (17)$$

$$\theta_{C1} = -\left(C_V + \frac{S_{VR} - S_{VL}}{2}\right) \quad (18)$$

In order to obtain the value for w_{c2} and θ_{c2} for y_2:

$$w_{C2} = \frac{S_{VR} - S_{VL}}{2} \quad (19)$$

$$\theta_{C2} = C_V + \frac{S_{VR} - S_{VL}}{2} \quad (20)$$

Layer 4: This is where all the signals coming from layers 3 and 4 are summed as following and rule is represented as r.

$$F = \frac{\sum_r Z_r U_V^{-1}(Z_r)}{\sum_r Z_r} \quad (21)$$

Learning in ULR fuzzy controller using Golden Section Search (GSS): As mentioned above, a fuzzy controller can be realized in a network as shown in Fig. 2 with appropriate membership function paramaters such as center parameters (C_V) , right (S_{VR}) and left (S_{VL}) spreads parameters of each of the triangular's membership functions. Therefore, learning of the fuzzy controller can be performed by adjusting the parameters. The learning in ULR fuzzy controller is based on reinforcement learning which is achieved when an optimal policy is found by maximizing the payoffs (or rewards) received over time. In this reinforcement learning model, the system interacts with its environment and as a result of this interaction; it then decides the output on which an evaluation is carried out. Learning is implemented during for this real time evaluation in order to maximize the expected result/output. In this study, we trained the fuzzy controller using the golden section search which is a linear search that does not require the calculation of the slope or derivative. The search begins by locating an interval in which the minimum of the performance occurs.

Golden section search is also known as golden ratio, mean, extreme ratio, golden proportion, golden mean, golden number or divine proportion. GSS has long existed for centuries. In fact many artists have used the mathematical principle to enhance the visual impact of their study. Example of such work includes Pyramids at Giza, The Parthenon, Michelangelo's Holy Family and many others. GSS is a linear approach used in finding where the local minimum occurs in a given interval. It starts with two initial guesses and proceeding two other interior points are calculated based on the golden ratio. Based on this understanding, for our proposed method, GSS is to train the vertex values given as C_V, S_{VL} and S_{VR} of each triangle in the system. During the training of the triangle, the interval of the three vertexes are selected as following: C of [C_V-S_{VL}, C_V+S_{VL}], C_V-S_{VL} of [C_V-S_{VL},C_V] and CV+S_{VL} of [C_V,C_V-S_{VR}].

While keeping the other parameters constant, we select one parameter from the parameter vector and the function becomes as a one variable function. A function f(x) is a unimodal on [a,b] if there exist a unique number c in [a,b] that is able replace the interval with a subinterval on which f(x) takes a minimum value. This is accomplished by evaluating the root of the function which has opposite sign at both ends by a bracketed points of a and b. Two interior points c and d are selected with *r* as the golden ratio $r = \left(\frac{\sqrt{5}-1}{2}\right)$ and is constant on each subinterval.

$$c = a + (1 - r)(b - a) \quad (22)$$

$$d = a + r(b - a) \quad (23)$$

Thus resulting in $a < c < d < b$. When $f(c) \le f(d)$, the minimum local would occur in the subinterval [a,d] and b is replaced with d Fig. 7a. The value of r remains constant on each of this subinterval search. The lengths of the two possible bracketing interval are specified to be the same such as (d-a)=(c-b). One of the old interior points will be used reused for the new subinterval while the other interior point will become an endpoint of the new subinterval. Then the search continues for the new interval.

But in the case of $f(d) < f(c)$, the minimum would occur in [c,b] and a is replaced with c Fig. 7b. As mentioned earlier, while one interior point is retained as a endpoint, the other point will become the new subinterval. On each iteration, only one new point will be found and only one new function evaluation will be made. The search continues and this process of bracketing is repeated until the distance between the two outer points is relatively small as specified by user.

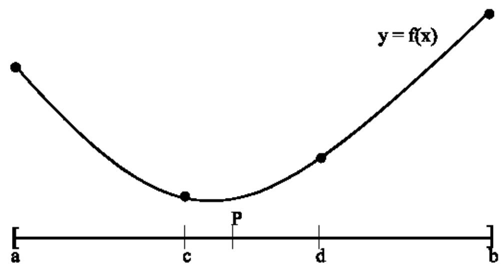


Fig. 7a: When f(c) < f (d), interval is squeezed from the right and the new interval is now a and d

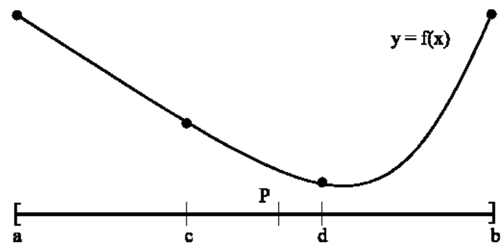


Fig. 7b: When f(d) < f(c), interval is squeezed from the left and the new interval is c and b

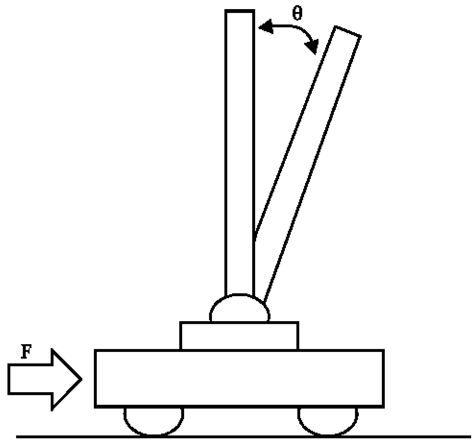


Fig. 8: Inverted pendulum

SIMULATION RESULTS

The proposed learning algorithm is applied to a classical fuzzy controller problem of inverted pendulum Fig. 8 in order to test the effectiveness of the proposed method. An inverted pendulum is a non linear and unstable system with one input and several output signals. The aim is to balance the pendulum vertically on a cart which moves in an uncontrolled state when an initial force is applied. The variables are set at zero when the cart at rest initially. The pendulum is set straight up and has only one degree of freedom. The primary task is to balance the pole and to keep the cart within boundaries by supplying the appropriate force to the cart.

The aim is to move the wagon along the x direction to a desired point without the pendulum falling. At a pendulum of x_1 from vertical, gravity produces an angular acceleration equal to x_2 and cart acceleration.

$$\ddot{\theta} = \frac{g\,sin\theta + cos\theta \dfrac{-F - ml\dot{\theta}^2 sin\theta}{m_c + m}}{l\left(\dfrac{4}{3} - \dfrac{m\,cos^2\theta}{m_c + m}\right)} \quad (24)$$

$$\ddot{x} = \frac{F + ml\left(\dot{\theta}^2 sin\theta - \ddot{\theta} cos\theta\right)}{m_c + m} \quad (25)$$

where

$\dot{\theta}$	:	angle of the pole with respect to the vertical line
$\ddot{\theta}$	:	angular velocity of the pole θ
F	:	force applied to the cart
g	:	gravity acceleration
m_c	:	mass of the cart, 1.0 kg
m	:	mass of the pole, 0.1 kg
l	:	length of the half-pole, 0.5 m

As mentioned previously, only the parameters of the first and third layers are trained. The parameters include all the weights and thresholds in these two layers which are altered accordingly when the relevant membership functions are changed. There are 13 triangles in the two layers with three vertexes C_V, C_V-S_{VL}, C_V+S_{VL} for each traingle. The to be set of parameters are defined as p_i and i^{th} time of learning as the completed time taken for the training. The algorithm used for the simulation can be outlined as such:

Step 1: Initialize the value of p_1, i=1.

Step 2: For the first triangle, train the vertexes C_V, C_V-S_{VL}, C_V+S_{VL} by GSS method in turn. The intervals needed in the algorithm are defined as following: C_V is trained in the interval of [C_V-S_{VL}, C_V+S_{VL}], C_V-S_{VL} in [C_V-S_{VL},C_V] and C_V+S_{VL} in [C_V,C_V-S_{VR}]. The values of the vertexes changes based on the direction of the minimization of the function f(x).

Step 3: Repeat step 2 for the other 12 triangles to complete the i^{th} time of learning and obtain p_{i+1}. i=i+1.

Step 4: Repeat step 2 and step3 until f(x) is sufficiently minimum.

This fuzzy controller is constructed with nine if-then rules as shown in Table 1, modeling after a skilled human sTable 1: If-then rules

			?	
		PO2	ZE2	NE2
	PO1	PL	PM	ZE
θ	ZE1	PS	ZE	NS
	NE1	ZE	NM	NL

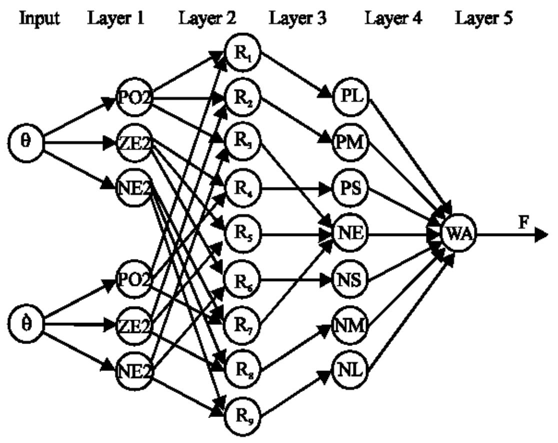


Fig. 9: Fuzzy controller

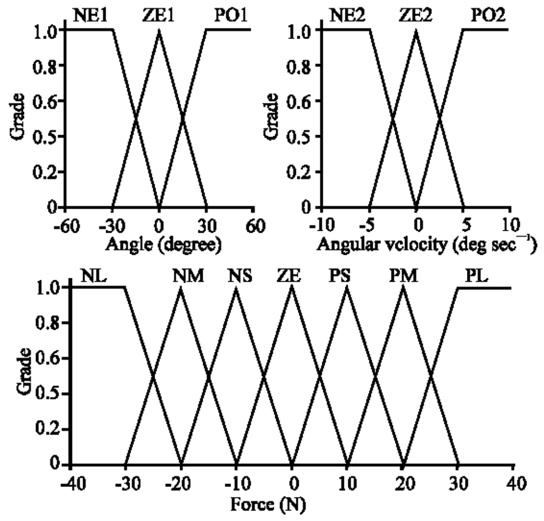
operator's knowledge of handling the system. The horizontal axis represents the angle of the pole where else the vertical is the pole velocity with representations such as Positive Large (PL), Positive Medium (PM), Positive Small (PS), Negative Large (NL), Negative Medium (NM), Negative Small (NS) and Zero (ZE). Figure 9 is the fuzzy controller for the specified system of inverted pendulum.

Figure 9 illustrates the fuzzy controller network used in this simulation. Only the second and fourth layer parameters are used in the training process. The simulation is carried out with and without the learning for a specified period of time with the angle set at 10 degrees. Results obtained from the 2 methods are compared to find the most optimal performance of the system. Fig.10 clearly demonstrates how the membership functions have shifted after learning was applied.

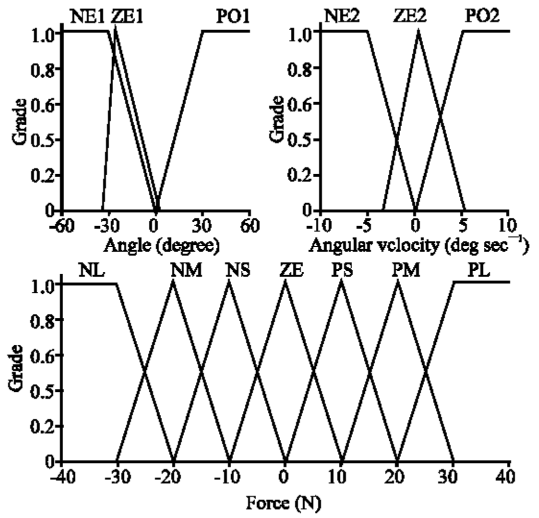
Based on the data obtained, the following graphs were plotted. Fig. 11 shows the controlled performance of the system with different initial angle of 10 degrees. When compared to the one before learnning, it shows that with the proposed algorithm, the fuzzy controller is capable of attaining the balance much faster as compared to the normal method without the learning.

As for the error, it clearly for angle of 10 degrees, the error rate shows an improment with the proposed method Fig. 12.

The simulation was repeated with various angles of the pole and the stopping parameter was set at -0.1°≤ θ ≤ 0.1° for the time duration of 2000 ms. The ranges of the



(a) Before learning



(b) After learning

Fig. 10: Membership functions before and after learning

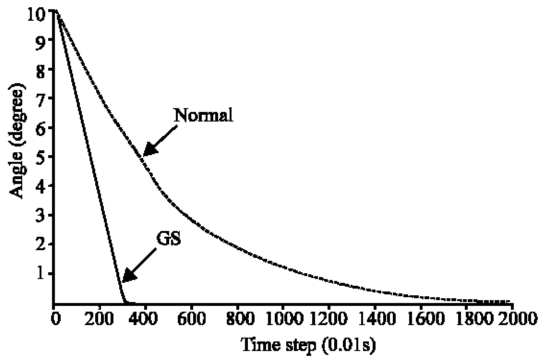


Fig. 11: The shape of the pole before learning and after learning when the angle is 10°.

Table 2: Parameters of membership functions (before learning)

Label	Center	Left spread	Right spread
PO1	30.0	30.0	5000.0
ZE1	0.0	30.0	30.0
NE1	-30.0	5000.0	30.0
PO2	5.0	5.0	5000.0
ZE2	0.0	5.0	5.0
NE2	-5.0	5000.0	5.0
PL	30.0	10.0	5000.0
PM	20.0	10.0	10.0
PS	10.0	10.0	10.0
ZE	0.0	10.0	10.0
NS	-10.0	10.0	10.0
NM	-20.0	10.0	10.0
NL	-30.0	10.0	10.0

Table 3: Parameters of membership functions (after learning)

Label	Center	Left spread	Right spread
PO1	30.0	30.0	5000.0
ZE1	-26.7	4.5	30.0
NE1	-30.0	4270.6	30.0
PO2	5.0	5.0	5000.0
ZE2	0.3	3.8	5.0
NE2	-5.0	1233.1	5.0
PL	30.0	10.0	5000.0
PM	20.0	10.0	10.0
PS	10.0	10.0	10.0
ZE	0.0	10.0	10.0
NS	-10.0	10.0	10.0
NM	-20.0	10.0	10.0
NL	-30.0	5000.0	10.0

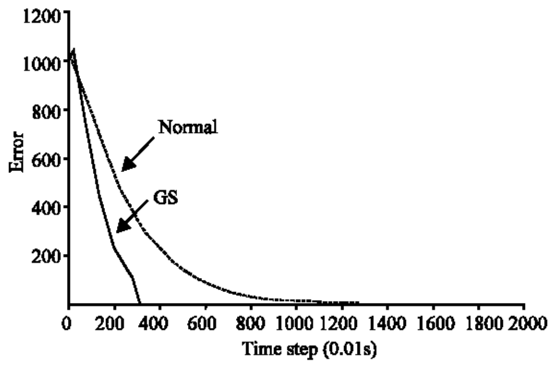


Fig. 12: The shape of the error before learning and after learning when the angle is 10°

tested angles were divided into group of small range (-12, -8, -4, 4, 8 and 12°) and larger angles of 40, 60 and 70° Fig. 13.

When tested for the smaller ranges of angles, it showed good control ability of returning to the initial state of the pole. Again this was repeated for angles of 40, 60 and 70° Fig. 14.

For the angle of 40°, with the proposed method, it managed to attain the balance state much earlier, though at the initial state there was a slight drop in the before it gained balance Fig. 15 and 16.

As for the angles of 60 and 70°, again the proposed method attained the balance state the faster where else

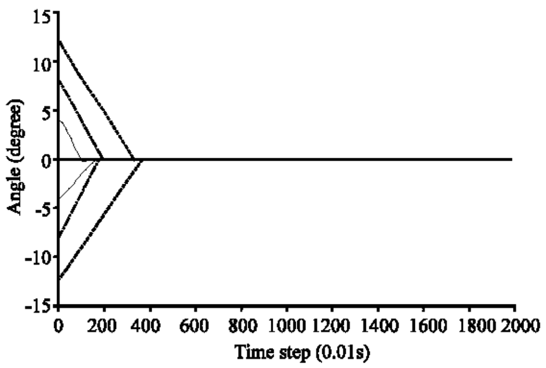


Fig. 13: Smaller ranges of angle

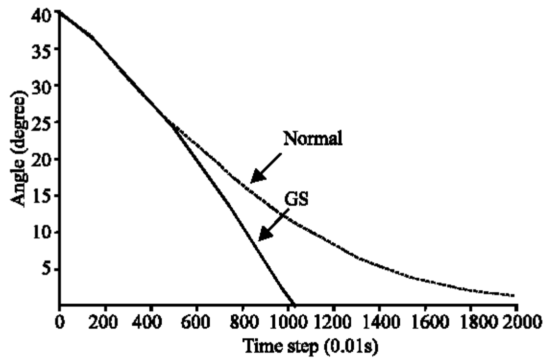


Fig. 14: Simulation angle of 40°

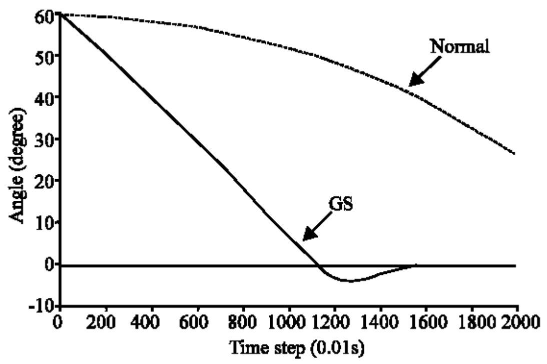


Fig. 15: Simulation angle of 60°

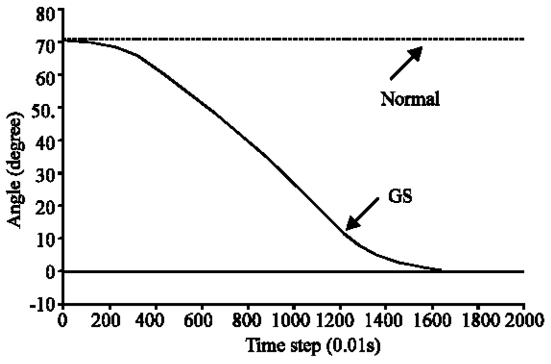


Fig. 16: Simulation angle of 70°

Table 4: The result of the control for angles

Angle(degree)	Before learning (NORMAL)	After learning (GS)
10	success	success
20	fault	success
30	fault	success
40	fault	success
50	fault	success
60	fault	success
70	fault	success
80	fault	fault

with the normal method; it failed to attain a balance state. The summary for the angle variation are captured in Table 4. The pole shape improved for the successful angle variations when plotted from the obtained results.

CONCLUSION

This study has described a ULR fuzzy controller with a new learning algorithm of golden section search. As previously reported in study, the hardware implementation is simply and consisting of only a single diode and a MOS transistor. By introducing this proposed algorithm, we do not need to find a derivative of the function since golden section search uses the linear approach in finding the local minimum. In this new approach, performance at a sequence of points is evaluated, starting at a distance of delta and as the performances increases between two successive iterations, a minimum will be bracketed. This process continues until the bracketing interval is reasonably small. We have applied this understanding on a common fuzzy controller application of the inverted pendulum. Results obtained from the varying inputs angles ranging from small to big angles showed an improvement in terms of the control performance of the system and also the error rate when compared to the one without learning. Thus the system has proved to be more efficient with the proposed learning of URL fuzzy controller with golden sections search.

REFERENCES

1. Zadeh, L.A., 1965. Fuzzy sets, Information and Control, 8: 338-353.
2. Kickert, W.J.M. and H.R. Van Nauta Lemke, 1976. Application of a Fuzzy Control in a warm plant, Automatica, 12: 301-308.
3. Rutherford, D. and G.A. Carter, 1976. A Heuristic Adaptive Controller for a Sinter Plant, Proc. 2nd IFAC Symp., Johannesburg.
4. King, P.J. and E.H. Mamdani, 1977. The application of fuzzy control systems to industrial processes, Automatica, 13: 235-242.
5. Li, Y.F. and C.C. Lau, 1988. Application of Fuzzy Control for Servo Systems, IEEE Intl. Conference on Robotics and Automation, pp:1511-1519.
6. Mamdani, E.H. and S. Asssilian, 1975. An experiment in linguistic synthesis with a fuzzy logic controller. Intl. J. Man-Machine Studies, 7: 11-13.
7. Homaifar, A. and Ed McCormick, 1995. Simultaneous design of membership functions and rule sets for fuzzy controllers using genetic algorithms, IEEE Trans. Fuzzy Systems, 3: 464-477.
8. Beom, H.R. and H.S. Cho, 1995. A sensor based navigation for a mobile robot using fuzzy logic and reinforcement learning, IEEE Trans. Syst. Man and Cybern., 25: 464-477.
9. Lin, C.T. and C.S.G. Lee, 1991. Neural-Network-based fuzzy logic control and decision systems, IEEE Trans. Fuzzy Systems, 2: 46-63.
10. Yung, N.H.C. and C. Ye, 1999. An intelligent mobile vehicle navigator based on fuzzy logic and reinforcement learning, IEEE Trans. Syst. Man, Cybern. B., 29: 314-321.
11. Tang, Z., O. Ishizuka and H. Matsumoto, 1993. A Model of Neurons with Unidirectional Linear Response, IEICE Trans. Fundamentals, 9: 1537-1540.
12. Tang, Z., M. Komori, O. Ishizuka and K. Tanno, 1996. An Adaptive Unidirectional Linear Response Fuzzy Controller Based on Reinforcement Learning, Electronics and Communications in Japan, Part 3, 2: 22-29.
13. Berenji, H.R. and P. Khedker, 1992. Learning and tunning fuzzy logic controllers through reinforcement, IEEE Trans. Neural Networks, pp: 724-740.