Adaptation of Fuzzy Cognitive Maps – a Comparison Study

Ján Vaščák, Ladislav Madarász

Department of Cybernetics and Artificial Intelligence, Faculty of Electrical Engineering and Informatics, Technical University of Košice
Letná 9, 042 00 Košice, Slovakia
jan.vascak@tuke.sk, ladislav.madarasz@tuke.sk

Abstract: This paper deals with the experimental study and comparison of various adaptation methods for setting-up parameters of fuzzy cognitive maps (FCMs). A survey is given of the best known methods, which are mostly based on unsupervised learning. The authors show better performance using supervised learning, namely least mean square approaches. Experiments were done on a simulation example of autonomous vehicle navigation. The paper is concluded by comparing their efficacy and properties.

Keywords: adaptation methods; fuzzy cognitive learning; Hebbian learning; supervised learning

1 Introduction

A navigation process of a vehicle requires solving a number of problems of different matter, i.e. finding not only a path to a goal but also regarding various constraints such as obstacle avoidance, limitations resulting from cooperation tasks with other vehicles, minimizing fuel consumption, etc. Such a process can be described by a set of conditions, which are at the same time mutually interconnected, creating complex decision chains and recurrences and causing very complicated mutual influences.

Means of artificial intelligence are based mostly on production rules, mainly because of their human-friendly knowledge representation, which will be the case if the rules are mutually independent; i.e. outputs of any rule do not enter antecedents of any other rule and so decision chains and closed loops are not created [10, 17]. Such systems will be named 'simple' rule-based systems for the following. However, a complex system is characterized by just such chains and loops. In that case the rule-based knowledge representation loses its main advantage and becomes 'unreadable'.

Therefore, for overcoming the limits of simple rule bases, fuzzy cognitive maps (FCMs) seem to be very suitable means, and these are able very clearly to represent graphically to a human notions and relations among them as seen in Fig. 1. A simple rule base (a rule set) lacking any chains or loops is the simplest or, in other words, the most degenerated form of an FCM. Hence, all operations and properties of production rules are valid for FCMs, too. Further, FCMs possess other additional properties and abilities (described in the next section) that are also convenient for the analysis and modelling of dynamic systems.

On the other hand, FCMs have the same basic drawbacks as other fuzzy systems: they are not able to self-learn. The design of adaptation approaches is much more difficult because of their complex structure and its variability [16, 18]. Therefore, at least the definition of notions, which are represented by nodes (see Fig. 1), is done manually by an expert and adaptation is limited to setting-up relations, i.e. graph edges. Most adaptation approaches are based on unsupervised learning, mainly Hebbian learning, e.g. [1, 3, 4, 13] but there are also approaches utilizing evolutionary computing [12].

Unsupervised learning is convenient for tasks such as clustering if we need, for instance, to separate data into a few groups. It is significant that elements from a given group have stronger relations among them and usually they react or are activated at the same time, which corresponds to the Hebbian paradigm of learning. However, navigation is a specific kind of activity somewhere between decision and control. It is a set of several processes with different dynamics, and any trial for separation could fail [14]. From this point of view, supervised learning seems to be more convenient. To verify this hypothesis as well, the well-known least mean square (LMS) approach, with some modifications, was used to show its suitability in this area.

In this paper, after introducing some basic notions regarding FCMs and their properties in Section 2, we will concentrate on variations of Hebbian learning and LMS in Section 3 and show properties of these learning approaches on a navigation simulation of a vehicle with their mutual comparison in Section 4. Finally, we will conclude with some remarks regarding the utilization of adaptation methods for navigation.

2 Fuzzy Cognitive Maps

In general a Cognitive Map (CM) is an oriented graph where its nodes represent notions and its edges causal relations (see Fig. 1). Mostly, notions are states or conditions and edges are actions or transfer functions, which transform a state in a node to another one in another node.

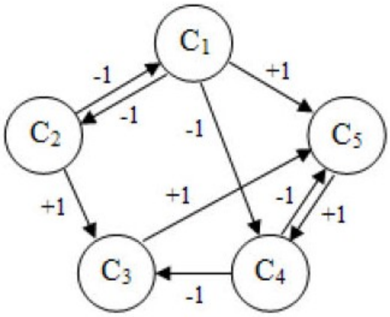


Figure 1
An example of FCM with crisp edges

CM is able to describe complex dynamic systems. It is possible to investigate e.g. limit cycles, collisions, etc. In addition, it is possible to define the strengths of relations, too. Originally they were represented by three values *-1*, *0* and *1*. Perhaps the main advantage of CM is its human-friendly knowledge representation in graphical form.

FCM represents an extension of CM and was proposed by Kosko in 1986 [6]. The extension is based on the strength values that are from an interval *[-1; 1]* as well as the fact that the nodes can be represented by membership functions. Strengths, or rather weights, correspond to rule weights in rule-based systems.

There are several possible formal definitions of FCM, but still the most commonly used one is in form given by Chen [2], which respects the original numerical matrix representation proposed by Kosko, where FCM is defined as a 4-tuple:

$$FCM = (C, E, \alpha, \beta) \tag{1}$$

where:

C – finite set of cognitive units described by their states $C = \{C_1, C_2, \ldots, C_n\}$,

E – finite set of oriented connections between nodes $E = \{e_{11}, e_{12}, \ldots, e_{nn}\}$,

α – mapping $\alpha \rightarrow [-1;1]$,

β – mapping $\beta \rightarrow [-1;1]$.

In other words, C represents nodes, E is for edges, α is a membership function placed in a node and the result is a grade of membership for values entering such a node. Similarly, β has the same meaning as α but for edges. The only difference compared to the definition of a fuzzy set is that the original interval of real values determined for grades of membership *[0; 1]* was extended to *[-1; 1]* in order to define as well negative connections, which refer to negations in logic. Similarly, β does the same but it is placed on a connection, i.e. it is its weight represented as a membership function. Further, we will use β only in the form of a singleton (a crisp value).

For computational representation of FCM a transition matrix is used. For the example in Fig. 1 it will look like:

$$E = \begin{bmatrix} 0 & -1 & 0 & -1 & 1 \\ -1 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & -1 & 0 & -1 \\ 0 & 0 & 0 & 1 & 0 \end{bmatrix}. \tag{2}$$

Cognitive units are in each time step in a certain state. Using a transition matrix we can compute their states for next time step and thus repeatedly for further steps. Similarly, as for differential equations we can draw phase portraits. To preserve values in prescribed limits a boundary function *L* is used as well. So we can compute the states for *t+1* as follows [6]:

$$C(t+1) = L(C(t) \cdot E). \tag{3}$$

Comparing Fig. 1 and 2 we can see FCMs are an extension of simple fuzzy rule-based systems. Fuzzy rules are totally independent because their consequents do not have any mutual influence, which is possible only in simpler decision cases. Simple rule-based systems do not enable any decision chains or representation of temporal information. From this point of view they are only a very special and restrained case of FCM, which can be depicted as an example in Fig. 2, where a set of *m* rules with inputs LX_i and outputs LU_i ($i=1,\ldots m$) is figured in the form of an FCM. There is depicted the evaluation process resulting in an accumulated (aggregated) value LU_c being defuzzified into a crisp form LU_c^*, too.

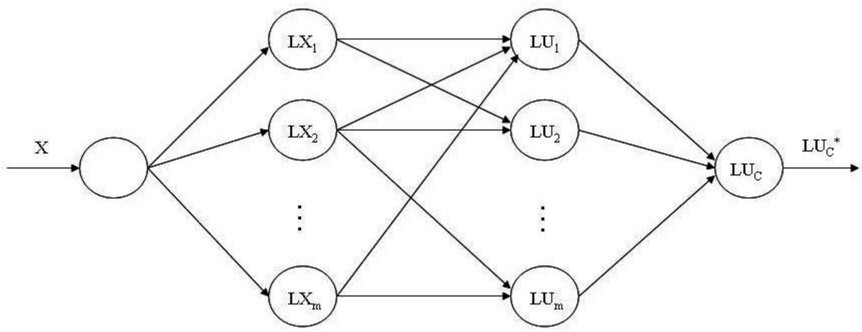


Figure 2
An example of FCM representing a simple rule set

3 Adaptation Approaches to FCM in Navigation

There are two basic approaches for setting-up the parameters of FCM. The first approach is based on expert skills where we need several experts who are able to set-up their own FCM. The weights of edges are then averaged. It seems to be very simple, but firstly we need to have such experts, and secondly the skills of these experts will necessarily be of different quality, which means we should rather do a weighted average, but there does not exist any hint how to determine these weights.

The second approach is based on unsupervised learning, which is well known mainly from neural networks. The creator of FCM, Bart Kosko, proposed using Hebbian learning [7], which is unsupervised. FCM was originally proposed only for causal relations, and looking at (3) we can see the state of a node and its change in next time $t+1$, i.e. $C(t+1)$ depends on $E(t)$ and $C(t)$, so we can write for the change of an element $e_{ij}(t) \in E(t)$ generally:

$$\dot{e}_{ij}(t) = f_{ij}(E(t), C(t)) + g_{ij}(t) \tag{4}$$

where $g_{ij}(t)$ is the so-called *forcing function*. In other words, there should be found a certain mutual correlation among nodes in individual time steps because they are more or less interconnected. So for n nodes we can get

$$\begin{aligned} E(t+n+1) = q_1 C(t+1)^T \cdot C(t+2) + q_2 C(t+2)^T \cdot C(t+3) + \ldots + \\ q_{n-1} C(t+n-1)^T \cdot C(t+n) + \\ q_n C(t+n)^T \cdot C(t+1), \end{aligned} \tag{5}$$

which is the sum of correlation matrices with the same dimensions like E and q_i are properly chosen weights.

Further, we can see that FCM, with its topology, resembles a neural network, too. Correlation learning (5) is a form of Hebbian learning and for neural networks it is known in the form:

$$\dot{e}_{ij}(t) = -e_{ij}(t) + C_i(t) \cdot C_j(t) \tag{6}$$

where $C_i(t), C_j(t) \in C(t)$ are directly interconnected nodes – i as output and j as input node. The formula (6) is named *Hebbian correlation learning* (HCL). The principle of Hebbian learning is based on the synchronous (at the same time) activation of both nodes. In such a case, the connection between them will be strengthened. Otherwise, if the nodes do not activate at the same moment, the connection will be weakened. However, in control generally due to various dynamic dependences, it can happen that the nodes will be activated at different time steps although there is a connection between them. Already this fact puts the idea of using Hebbian learning into doubt. Several experiments such as e.g. in [3, 7] as well as our own experience confirm poor results using the original form of Hebbian learning - HCL (6). There are problems with the selection of training patterns. If their activations are small, then probably only a few connections will arise and FCM will be poorly structured. In the opposite case many so-called false connections will remain, which should be removed.

Hebbian learning with damping / decay (HLD) is a modification of HCL using forgetting (decay) factor, which depends on the forgetting parameter α, and it is able to control the learning speed by learning parameter γ:

$$\dot{e}_{ij}(t) = \gamma \cdot C_i(t) \cdot C_j(t) - \alpha \cdot C_i(t) \cdot e_{ij}(t). \tag{7}$$

Another modification is *differential Hebbian learning* (DHL), which takes into consideration changes of node activations, i.e. their derivatives. If the derivatives of $C_i(t)$, $C_j(t)$ are nonzero then

$$\dot{e}_{ij}(t) = -e_{ij}(t) + \dot{C}_i(t) \cdot \dot{C}_j(t). \tag{8}$$

Nonlinear Hebbian learning (NHL) [13] is a more sophisticated method using stopping criteria as well. It uses its own calculations of activation values for nodes $C_j(t)$:

$$C_j(t+1) = f(C_j(t) + \sum_{\substack{i \neq i \\ i=1}}^{n} C_i(t) \cdot e_{ij}(t)). \tag{9}$$

The weight e_{ij} for the next time step $t+1$ is then calculated as

$$e_{ij}(t+1) = \eta \cdot e_{ij}(t) + \gamma \cdot C_j(t) \cdot (C_i(t) - sgn(e_{ij}(t)) \cdot e_{ij}(t) \cdot C_j(t)) \tag{10}$$

where η is the weight decay learning coefficient and γ is the own learning parameter. However, formula (10) is used only for weights that were initially nonzero, which requires knowledge from an expert, and this method is convenient first of all for final fine-tuning FCM.

There are two stopping criteria for learning: criterion of minimum square error F_1 of nodes, which represent outputs of FCM (denoted as *OC*) and criterion F_2 indicating whether there are still some significant changes of node activations during learning:

$$F_1 = \sqrt{\sum_{k=1}^{p} (OC_k - T_k)^2}, \tag{11}$$

$$F_2 = \left| OC_k(t+1) - OC_k(t) \right| < \varepsilon. \tag{12}$$

F_1 gives us information about the performance quality of p outputs from FCM (e.g. action variables of a controller in the form of FCM) where T_k is an average value from the interval of allowed values for a given *OC* and it should be known by experts (e.g. the allowed range of accelerations for a vehicle). The goal is to minimize F_1 under a certain value. F_2 tells us about the stability of the designed FCM, which is characterized by the stabilizing activation values of output nodes. The size of ∂ has been chosen after a series of experiments to be *0,002* [13]. When both criteria are fulfilled then the learning will be stopped.

The modified version, the so-called *improved nonlinear Hebbian learning* (INHL) introduced in [9], is based on the following weight change adaptation:

$$\Delta e_{ij}(t) = \eta \cdot \Delta e_{ij}(t-1) + \gamma \cdot z(t)^2 \cdot (1 - z(t)) \cdot (C_j(t) - e_{ij}(t-1) \cdot C_j(t)) \quad (13)$$

where $z(t) = 1/\left(1 + \exp(-C_i(t))\right)$ and the weight in the next time step $e_{ij}(t+1)$ is calculated as $e_{ij}(t)+\Delta e_{ij}(t)$. There is modified also the criterion F_1 as a sum of OC_k^2.

Because our training data also contained the desired values of output variables we tried to use a supervised learning method, in this case a well known LMS method defined as

$$e_{ij}(t+1) = e_{ij}(t) + \gamma \cdot (y_d(t) - \sum_{i=1}^{n} e_{ij}(t) \cdot C_i(t)) \cdot C_i(t) \quad (14)$$

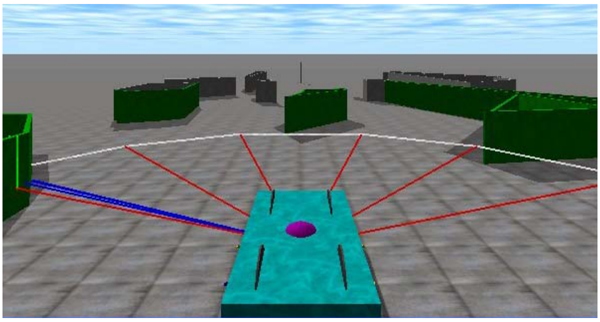
where $\sum_{i=1}^{n} e_{ij}(t) \cdot C_i(t)$ is the activation value of the input node $C_j(t)$ and $y_d(t)$ is the desired activation value of $C_j(t)$.

4 Modifications and Experimental Comparison

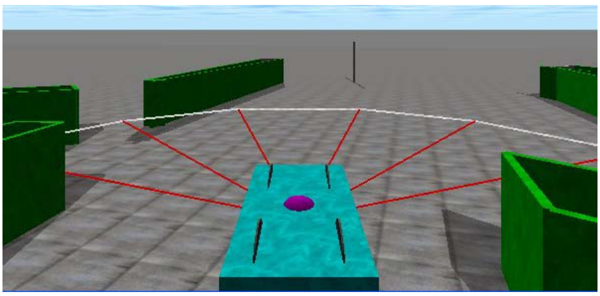
For automatic navigation of a vehicle, a simulator based on ODE library (http://www.ode.org/) was proposed [15] as being able to respond to real physical conditions. The main interface consists of an area with various obstacles and a vehicle model that can be controlled manually as well as automatically implementing source code of a given method. The simulator is able to collect data during the movement using 5 sensors that divide the surroundings into the same number of sectors, and from these data it can compute the current position of the vehicle. The goal is depicted as a vertical pole and its position is given in advance, see Fig. 3.

For all methods three basic experiments were performed using different starting positions, as depicted in Fig. 3. For comparison purposes, a manually designed

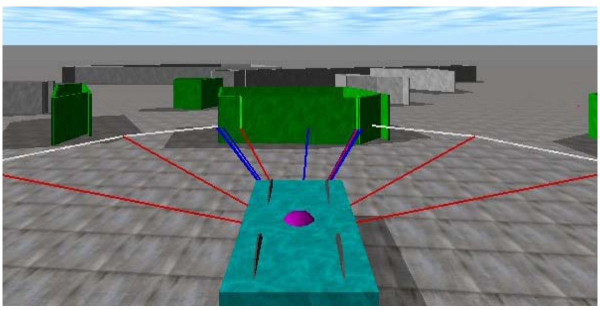
FCM was constructed, which is depicted in Fig. 4. There are in total 16 nodes, where 4 of them are output nodes (*OC*) C_0 – C_3 with bold capitals – turn to left (L), go forward (F), turn to right (R), stop (S). Other nodes represent calculated positions of the goal in relation to the vehicle C_4 – C_7, with symbols G_L – goal left, G_S – goal straight, G_R – goal right, G_C – goal close, and signals from sensors C_8 – C_{15} where S_i means the number of a sensor, c – close and cc – critically close. Positive connections are depicted as solid lines and negative connections as dashed ones.



(a)



(b)



(c)

Figure 3

Starting positions of the vehicle simulator

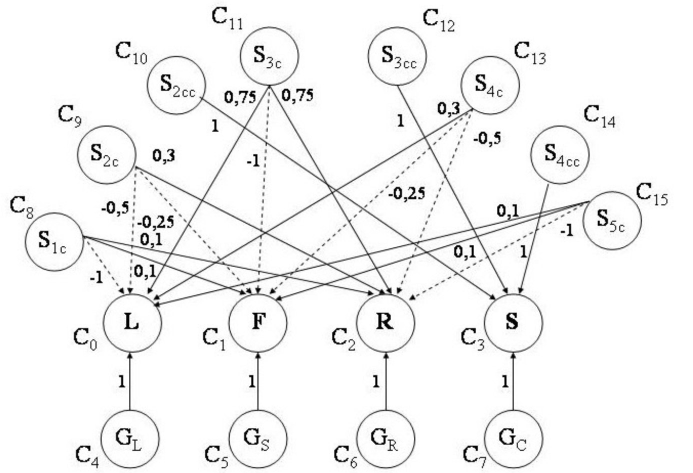


Figure 4

Manually defined reference FCM for the navigation of a simulated vehicle

Concerning the functions, which were implemented in the nodes, they are membership functions (Fig. 5) for evaluating signals from the sensors as *obstacle closeness* – nodes C_8 – C_{15}, calculated position of the goal like *goal position* – nodes C_4 – C_6 and *goal closeness* – node C_7 and finally, for evaluating action values of output nodes *angle of turning* – nodes C_0 – C_2. When the activation value in the node C_3 (stopping) achieves the value *0,95* then the vehicle will stop.

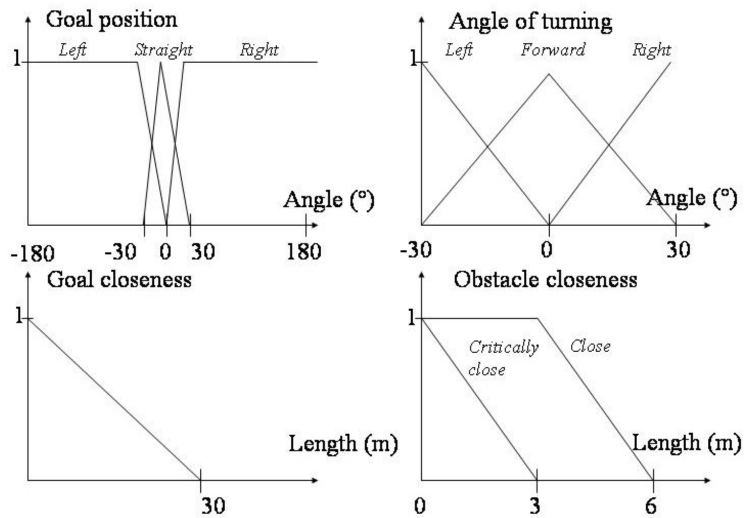


Figure 5

Membership functions of manually designed FCM nodes

In the case of a manual design it is necessary to mention we can consider only suboptimal solutions because we can never state that there is no better solution. Therefore, we suppose after a satisfactory number of cycles in the manual setting-up process that we have potentially the ‘best’ possible solution.

Concerning the evaluation criteria, we chose a subjective classification from 1 – the best to 5 – the worst (unsatisfactory). Generally such aspects were observed and evaluated as the behaviour of the vehicle in the goal area, the quality of obstacle avoidance, the smoothness of the wheel traces (whether the wheels did not turn too chaotically as a beginner), the choice of possible paths and of course the number of collisions with obstacles. The subjectivity of our evaluation is based on these three points: criteria weights, comparison to manually designed FCM as well as the calculation of some criteria. Firstly, the mentioned criteria have different weights of importance, with in this case the most important criteria being the number of collisions and the number of successfully found paths to the goal, which in turn are related to the total number of experiments. These weights were determined manually by a human reflecting his/her opinions. Secondly, resulting behaviour of a vehicle based on the adapted FCM is compared to the behaviour resulting from the manually designed FCM, which again reflects the properties of a given human. Finally, a criterion such as the choice of a possible path is evaluated purely subjectively, where another user may have a different opinion, because we consider uncertain notions such as the complexity of a path.

It seems to be a very ‘non-scientific’ approach but considering some objective criteria such as the shortest trajectory, the minimum number of turnings, the minimum number of collisions with obstacles, the minimum time, etc. there will be many situations when some criteria are evaluated better and some worse, and calculating the total measure of fulfilment is a nondeterministic task dependent on a given application. (This task leads again to weighted averaging, and simply setting-up weights for given criteria is a nondeterministic and very subjective task.) So we can get several solutions with the same or approximate total evaluation value. If we compare subjective and objective evaluation approaches the first approach seems to be more similar to the intuitive human reasoning and this was the most important element for choosing the subjective approach.

As already mentioned above, the task of adaptation methods is only to set-up the weights of connections e_{ij}. The structure (topology) of nodes and their activation (in our case membership) functions are given by an expert in advance. Further, the behaviour description of FCM designed by individual adaptation methods follows.

HCL by (6) showed really very poor results. Only two connections were created and the vehicle stopped very quickly. The reason for this is that in the training data there were many patterns which did not activate any membership function in the nodes, and thus in such a case already created connections were zeroed. Therefore, such patterns were removed before learning started to prevent clearing connections. The number of connections increased considerably but weights were very small so the result was almost identical with the previous case. Only through using a very large training set and after many learning cycles was this approach able to bring some results, which is unacceptable.

Similarly, HLD by (7) behaved in a manner similar to HCL – without filtering inactive patterns, only a few connections were established; and omitting them, too many connections were established. Either there were strong connections to the stopping node C_3, which caused premature finish, or too many, and weak connections are in fact false and they should be removed. In any event, such a structure of connections needs manual corrections done by an expert, which is not welcome.

In contrast, concerning Hebbian learning, NHL and INHL represented excellent results [8] after only a few learning cycles. Although NHL needs setting-up of nonzero connections in the matrix *E* by an expert, it produces very stable solutions without any false or weak connections behaving like noise. There are very small differences between a manually designed matrix *E* and matrices created by NHL and INHL. As the INHL method also enables the updating of zero initial weights, it tends to create almost a full set of all possible connections but with very small weight values which do not affect the performance of such an FCM. Filtering such connections could be useful because INHL shows slightly smaller stability of learning than NHL, and it needs a few more learning cycles.

Using supervised learning also brought very good results. At first, LMS by (14) was tried on a zero initial weight matrix. The results were only a little worse than those obtained by the manually designed FCM. However, the best results of all were achieved using LMS with unity initial matrix. The smoothness of wheel traces was even better than with an FCM designed by an expert, but it is necessary to do many experiments setting-up the learning parameter to find the best one, which is time consuming work. In Fig. 6 there are depicted for comparison the weight matrices of LMS with unity initialization and the FCM manually designed by an expert. We can see significant differences in matrices although they are functionally similar.

Table I

Successfulness of FCM Learning Methods

Order	**Method**	**Note**
1	LMS with unity initialization	1
2	Expert	2
3	Nonlinear Hebbian learning	2
4	Improved nonlinear Hebbian learning	2
5	LMS with zero initialization	3
6	*Hebbian learning with damping*	*5*
7	*Hebbian correlation learning*	*5*

In the Table I are the mentioned methods ordered according to their degree of success, and total evaluations are assigned to them based on experiments done with the simulation of vehicle navigation in the area with obstacles. HLD and HCL are denoted in italics because they seem to be fully inconvenient for FCM.

Conclusions

Although the proposed experiment does not require the creation of a complex form of FCM containing decision chains or loops, and therefore it was possible to use a simple LMS method, the utilization of supervised learning is a challenge for research in spite of many theoretical difficulties. Unsupervised learning is not a nostrum for solving problems of machinery learning. The Hebbian law is very general as it does not go into the depth of learning principles, and therefore enhanced methods derived from Hebbian learning such as NHL and INHL are perhaps also efficient only for simpler low-order problems such as two-tank systems, etc., as is often presented in literature. Therefore it could also be very useful to focus interest on various interpolation and nonlinear methods already used in conventional rule-based fuzzy systems [5, 11]. There is still one more aspect which should be taken into consideration. Using learning methods we can get indeed two functionally identical systems – FCM but their structure is quite different, as we can see also in Fig. 6. This is a problem because especially FCM should reflect just human representation of knowledge.

$$
\begin{bmatrix}
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-1 & 0{,}1 & 0{,}1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
-0{,}5 & -0{,}25 & 0{,}3 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0{,}25 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0{,}75 & -1 & 0{,}75 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0{,}30 & -0{,}25 & -0{,}5 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0{,}25 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0{,}1 & 0{,}1 & -1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}
$$

(a)

$$\begin{bmatrix}
0 & 0 & 0 & 0 & -0{,}06 & -0{,}57 & -0{,}2 & -0{,}52 & -0{,}63 & -0{,}63 & -0{,}03 & 1 & -0{,}08 & -0{,}63 & -0{,}3 & -0{,}63 \\
0 & 0 & 0 & 0 & 0{,}18 & 0{,}54 & 0{,}13 & 0{,}7 & 0{,}5 & 0{,}25 & 0 & 0{,}45 & -0{,}07 & 0{,}25 & 0 & 0{,}5 \\
0 & 0 & 0 & 0 & -0{,}2 & -0{,}57 & -0{,}06 & -0{,}52 & -0{,}63 & -0{,}63 & -0{,}3 & 1 & -0{,}08 & -0{,}63 & -0{,}03 & -0{,}63 \\
0 & 0 & 0 & 0 & 0{,}48 & 0{,}19 & 0{,}48 & 0{,}48 & 0{,}28 & 0{,}43 & 1 & 1 & 1 & 0{,}43 & 1 & 0{,}28 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\
0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0 & 0
\end{bmatrix}$$

(b)

Figure 6

Weight matrices of FCM designed manually (a) and using LMS with unity initialization (b)

Acknowledgement

This work is the result of the project implementation: Center of Information and Communication Technologies for Knowledge Systems (ITMS project code: 26220120020) supported by the Research & Development Operational Program funded by the ERDF.

References

[1] Blanco A., Delgado M., Pegalajar M. C., Fuzzy Automaton Induction Using Neural Network, International Journal of Approximate Reasoning, Elsevier, Vol. 27, pp. 1-26, 2001

[2] Chen S. M., Cognitive-Map-based Decision Analysis Based on NPN Logics, Fuzzy Sets and Systems, Elsevier, Vol. 71, No. 2, pp. 155-163, 1995

[3] Gavalec M., Mls K., Fuzzy Cognitive Maps and Decision Making Support, in: Proc. of the 21th Int. Conf. Mathematical Methods in Economics, Prague, pp. 87-93, 2003

[4] Huerga A. V, A Balanced Differential Learning Algorithm in Fuzzy Cognitive Maps, in: Proc. of the Sixteenth International Workshop on Qualitative Reasoning, Barcelona, Spain, 2002

[5] Johanyák Zs. Cs., Kovács Sz., A Brief Survey and Comparison on Various Interpolation-based Fuzzy Reasoning Methods, Acta Polytechnica Hungarica, Vol. 3, No. 1, pp. 91-105, 2006

[6] Kosko B., Fuzzy Cognitive Maps, International Journal of Man-Machine Studies, Elsevier, Vol. 24, No. 1, pp. 65-75, 1986

[7] Kosko, B., Fuzzy Engineering, Prentice-Hall, 1997

[8] Kotras M., Adaptation of Fuzzy Cognitive Maps for Needs of Navigation, Bc. thesis in Slovak, Technical University in Košice, Slovakia, 2009

[9] Li S. J., Shen R. M., Fuzzy Cognitive Map Learning Based on Improved Nonlinear Hebbian Rule, in: Proc. of the Third Int. Conference on Machine Learning and Cybernetics, Shanghai, China, pp. 2301-2306, 2004

[10] Madarász L., Intelligent Technologies and their Applications in Complex Systems (in Slovak), Technical University in Košice, Slovakia, p. 348

[11] Oblak S., Škrjanc I., Blažič S., If Approximating Nonlinear Areas, then Consider Fuzzy Systems, IEEE Potentials, Vol. 25, No. 6, pp. 18-23, 2006

[12] Papageorgiou E. I., Parsopoulos K. E., Stylios C. D., Groumpos P. P., Vrahatis M. N., Fuzzy Cognitive Maps Learning Using Particle Swarm Optimization, International Journal of Intelligent Information Systems, Springer, Vol. 25, No. 1, pp. 95-121, 2005

[13] Papageorgiou E. I., Stylios C. D., Groumpos P. P., Unsupervised Learning Techniques for Fine-Tuning Fuzzy Cognitive Map Causal Links, Int. Journal of Human-Computer Studies, Elsevier, Vol. 64, pp. 727-743, 2006

[14] Pozna C., Troester F., Precup R.-E., Tar, J. K., Preitl S., On the Design of an Obstacle Avoiding Trajectory: Method and Simulation, Mathematics and Computers in Simulation, Elsevier Science, Vol. 79, No. 7, pp. 2211-2226, 2009

[15] Rutrich M., Adaptation of Fuzzy Cognitive Maps for Navigation Needs, MSc. thesis in Slovak, Technical University in Košice, Slovakia, 2009

[16] Tar, J. K. et al., Centralized and Decentralized Applications of a Novel Adaptive Control, INES 2005, Budapest, Budapest Tech, 2005, pp. 87-92

[17] Vaščák J., Fuzzy Cognitive Maps in Path Planning, Acta Technica Jaurinensis, Vol. 1, No. 3, 2008, pp. 467-479

[18] Vaščák J., Madarász L., Automatic Adaptation of Fuzzy Controllers, Acta Polytechnica Hungarica, Budapest, Hungary, Vol. 2, No. 2, 2005, pp. 5-18