Associative Model for Solving the Wall-Following Problem

Rodolfo Navarro, Elena Acevedo*, Antonio Acevedo, and Fabiola Martínez

Escuela Superior de Ingeniería Mecánica y Eléctrica, IPN, Mexico City, Mexico
rufles248@hotmail.com,
{eacevedo,macevedo,fmartinezzu}@ipn.mx

Abstract. A navigation system for a robot is presented in this work. The Wall-Following problem has become a classic problem of Robotics due to robots have to be able to move through a particular stage. This problem is proposed as a classifying task and it is solved using an associative approach. In particular, we used Morphological Associative Memories as classifier. Three testing methods were applied to validate the performance of our proposal: Leave-One-Out, Hold-Out and K-fold Cross-Validation and the average obtained was of 91.57%, overcoming the neural approach.

Keywords: Classification, Associative Models, Morphological models, Wall-Following.

1 Introduction

It is a fact of life that technology makes progress by leaps and bounds, insomuch that robots are more common day by day in our environment. It is possible that in the near future they could be in our homes performing daily homework which is done by us.

Nowadays, most of the robots are operated by human beings. Some others can operate under an autonomous hand such as Asimo [1], the Murata Boys [2], the participants of RoboCup league [3], Surena 2 [4] or the HRP-4 [5] among others. We have senses which give us freely movement without colliding against people or obstacles, however, robots need sensors to simulate our senses and a navigation system to be able to move through a particular stage. This navigation system is called Wall-Following which has become a classic problem of the Robotics. Some researchers have proposed solutions using diverse computational tools for improving the performance, these tools are: Genetic programming [6], Fuzzy logic [7], Computer vision [8], Vector Field Histogram [9], a system based on sonar and odometric sensorial information [10], Chaos theory [11][12], Hopfield neural network, [13], Topological mapping [14], systems implemented in hardware together with Lyapunov functions [15], Proportional Derivative (PD) controllers [16], IR Sensors [17][18] and Distance sensors [19].

* Corresponding author.

J.A. Carrasco-Ochoa et al. (Eds.): MCPR 2012, LNCS 7329, pp. 176–186, 2012.

In this work, we applied the associative approach to solve the Wall-Following problem. We used the morphological-based associative memory as classifier and the data was obtained from the UC Irvine Machine Learning Repository [20]. In [21] other results are reported from works which used the same data.

In section 2, memory associative concept is presented together with the theory of morphological associative memories. The description of data and the model for resolving the Wall-Following problem is presented in Section 3.

Section 4 shows the results from our proposal and the comparisons with other approaches. Finally, Conclusions are presented.

2 Morphological Associative Memories

An Associative Memory (AM) [22] **M** is a system that relates input patterns, and outputs patterns. Two phases comprise the design of an AM: learning phase and recalling phase. In the learning phase, the memory is trained by associating input patterns **x** and output patterns **y** (see figure 1). Both input and output patterns can represent any association, for example: fingerprints with faces, names with telephone numbers, DNA sequences with names, etc.

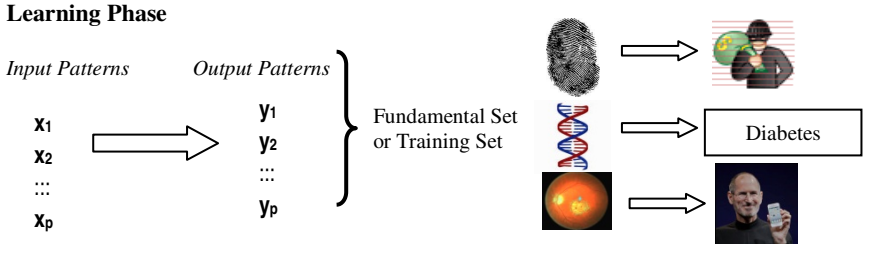

Fig. 1. The learning phase for an Associative Memory

After the associative memory was trained, output patterns can be recalled by presenting the input patterns to the memory. This task is performed by the recalling phase (see figure 2).

In figure 2, one can observe that when an input pattern x^k is presented to the AM its corresponding pattern y^k must be recalled. Moreover, if a noisy version of an input pattern x^k represented by $\widetilde{x^k}$ is presented to the associative memory, the corresponding pattern y^k should be recalled, if this happens then AM has a correct recall.

Formally, we can say that for a k integer and positive, the corresponding association will be denoted as (x^k, y^k) . The associative memory **M** is represented by a matrix whose ij-th component is m_{ij}. Memory **M** is generated from an a priori finite set of known associations, known as the fundamental or training set of associations.

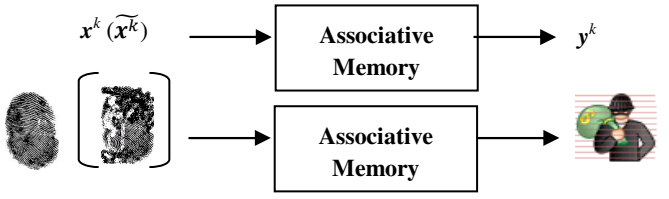

Fig. 2. The recalling phase for an Associative Memory

If μ is an index, the fundamental set is represented as: $\{(x^\mu, y^\mu) | \mu = 1, 2, \ldots, p\}$ with p the cardinality of the set. The patterns that form the fundamental set are called fundamental patterns. If it holds that $x^\mu = y^\mu$, $\forall \mu \in \{1, 2, \ldots, p\}$, **M** is *autoassociative*, otherwise it is *heteroassociative*; in this case it is possible to establish that $\exists \mu \in \{1, 2, \ldots, p\}$ for which $x^\mu \neq y^\mu$.

2.1 Morphological Associative Memories

The fundamental difference between classic associative memories (*Lernmatrix* [23], *Correlograph* [24], *Linear Associator* [25] and Hopfield [26]) and Morphological associative memories [27] lies in the operational bases of the latter, which are the morphological operations: dilation and erosion. This model broke out of the traditional mould of classic memories which use conventional operations for vectors and matrices in learning phase and sum of multiplications for recovering patterns. Morphological associative memories change products to sums and sums to maximum or minimum in both phases.

The basic computations occurring in the proposed morphological network are based on the algebraic lattice structure (**R**, $\vee$, $\wedge$, +), where the symbols $\vee$ and $\wedge$ denote the binary operations of maximum and minimum, respectively. Using the lattice structure (**R**, $\vee$, $\wedge$, +), for an m x n matrix A and a p x n matrix B with entries from **R**, the matrix product $C = A \nabla B$, also called the *max* product of A and B, is defined by equation (1).

$$c_{ij} = \bigvee_{k=1}^{p} a_{ik} + b_{kj} = (a_{i1} + b_{1j}) \vee (a_{i2} + b_{2j}) \vee \cdots \vee (a_{ip} + b_{pj}) \quad (1)$$

The *min product* of A and B induced by the lattice structure is defined in a similar fashion. Specifically, the *i,j*th entry of $C = A \Delta B$ is given by equation (2).

$$c_{ij} = \bigwedge_{k=1}^{p} a_{ik} + b_{kj} = (a_{i1} + b_{1j}) \wedge (a_{i2} + b_{2j}) \wedge \cdots \wedge (a_{ip} + b_{pj}) \quad (2)$$

Suppose we are given a vector pair $\mathbf{x}= (x_1, x_2, \ldots, x_n)^t$ and $\mathbf{y}= (y_1, y_2, \ldots, y_n)^t \in \mathbf{R}^m$. An associative morphological memory that will recall the vector when presented the vector is showed in equation (3)

$$W = y\nabla(-x)^{\mathrm{t}} = \begin{bmatrix} y_1 - x_1 & \cdots & y_1 - x_n \\ \vdots & \ddots & \vdots \\ y_m - x_1 & \cdots & y_m - x_n \end{bmatrix} \tag{3}$$

Since W satisfies the equation $W \Delta \mathbf{x} = \mathbf{y}$ as can be verified by the simple computation in equation (4)

$$W\nabla x = \begin{bmatrix} \bigvee_{i=1}^{n} (y_1 - x_i + x_i) \\ \vdots \\ \bigvee_{i=1}^{n} (y_m - x_i + x_i) \end{bmatrix} = y \tag{4}$$

Henceforth, let $(\mathbf{x}^1, \mathbf{y}^1)$, $(\mathbf{x}^2, \mathbf{y}^2)$, …, $(\mathbf{x}^p, \mathbf{y}^p)$ be p vector pairs with $x^k = (x_1^k, x_2^k, \ldots, x_n^k)^t \in \mathbf{R}^n$ and $y^k = (y_1^k, y_2^k, \ldots, y_m^k)^t \in \mathbf{R}^m$ for $k = 1, 2, \ldots, p$. For a given set of pattern associations $\{(\mathbf{x}^k, \mathbf{y}^k) \mid k= 1, 2, \ldots, p\}$ we define a pair of associated pattern matrices (X, Y), where $X=(\mathbf{x}^1, \mathbf{x}^2, \ldots, \mathbf{x}^p)$ and $Y=(\mathbf{y}^1, \mathbf{y}^2, \ldots, \mathbf{y}^p)$. Thus, X is of dimension n x p with i,jth entry x_i^j and Y is of dimension m x p with i,jth entry y_i^j. Since $\mathbf{y}^k \nabla (-\mathbf{x}^k)^t = \mathbf{y}^k \Delta (-\mathbf{x}^k)^t$, the notational burden is reduced by denoting these identical morphological outer vector products by $\mathbf{y}^k \times (-\mathbf{x}^k)^t$. With each pair of matrices (X, Y) we associate two natural morphological m x n memories M and W defined by

$$M = \bigvee_{k=1}^{p} (y^k \times (-x^k)^t) \tag{5}$$

$$W = \bigwedge_{k=1}^{p} (y^k \times (-x^k)^t) \tag{6}$$

With these definitions, we present the algorithms for the learning and recalling phase.

Learning Phase

1. For each p association $(\mathbf{x}^\mu, \mathbf{y}^\mu)$, the minimum product is used to build the matrix $\mathbf{y}^\mu \Delta (-\mathbf{x}^\mu)^t$ of dimensions m x n, where the input transposed negative pattern $\mathbf{x}^\mu$ is defined as $(-\mathbf{x}^\mu)^t = \left(-x_1^\mu, -x_2^\mu, \ldots, -x_n^\mu\right)$.
2. The maximum and minimum operators ($\vee$ and $\wedge$) are applied to the p matrices to obtain M and W memories as equations (5) and (6) show.

Recalling phase

In this phase, the minimum and maximum product, Δ and ∇, are applied between memories M or W and input pattern $\mathbf{x}^{\omega}$, where $\omega \in \{1, 2, ..., p\}$, to obtain the column vector $\mathbf{y}$ of dimension m as equations (7) and (8) shows:

$$\mathbf{y} = M \Delta \mathbf{x}^{\omega} \tag{7}$$

$$\mathbf{y} = W \nabla \mathbf{x}^{\omega} \tag{8}$$

3 Methods and Materials

Dataset Information

The dataset is named: "Wall-Following navigation task with mobile robot SCITOS-G5" and is available at [28].

Data was obtained by using the robot SCITOS G5, which navigated through the room following the wall in a clockwise direction, for 4 rounds.

The dataset contains four sensor readings named 'simplified distances' and the corresponding class label. These simplified distances are referred to as the 'front distance', 'left distance', 'right distance' and 'back distance'. They consist, respectively, of the minimum sensor readings among those within 60 degree arcs located at the front, left, right and back parts of the robot.

The original data consisted of 5456 records, however, there were repeated records or same data that was assigned to a different class, therefore, the dataset was depurated and finally we had 5406 records.

Specifically, the dataset has four real-valued attributes and four classes: Move-Forward, Sharp-Right, Slight-Left and Slight-Right.

Proposed Morphological Classifier

The first step to describe the classifier is to present two definitions [29].

Positive-Hot Vector. Let i and $j \in Z^+$ and $a \in \mathbf{R}^+$ and $p \in Z^+$, where p is the number of associations. The i-th *positive-hot* vector with dimension p is defined as $Ph^i \in \mathbf{R}^p$ which i-th component $Ph_i^i = a$ and the remaining components are $Ph_j^i = 0$, $\forall j \neq i$, $1 \leq j \leq p$.

Negative-Hot Vector. Let i and $j \in Z^+$ and $b \in \mathbf{R}^-$ and $p \in Z^+$, where p is the number of associations. The i-th *positive-hot* vector with dimension p is defined as $Nh^i \in \mathbf{R}^p$ which i-th component $Nh_i^i = b$ and the remaining components are $Nh_j^i = 0$, $\forall j \neq i$, $1 \leq j \leq p$.

The number b is the symmetric of a then, $b = -a$.

Positive-hot and Negative-hot vectors are used together with *max* and *min* memories, respectively. The given name to these vectors is an analogy with the one-hot and zero-hot vectors used in Alpha-Beta Bidirectional Associative Memories [30], they have the form: [1 0 0 0] and [0 1 1 1], respectively. The latter vectors are utilized to identify the corresponding out pattern associated with input pattern. In our case, positive and negative-hot vectors are use to give a higher weight for each class, that is why we assigned the higher or lower (for *max* and *min* memory, respectively) value to the first element in each vector representing class 1, then, we assigned the lower and higher values to the second element when the pattern corresponds to class 2 and so on. This way to create out patterns allowed us to remark differences between classes.

We assigned the positive-hot and negative-hot vector to each class as Table 1 shows.

Table 1. Positive-Hot and Negative-Hot vector assigned to each class

Movement	Positive-Hot vector	Negative-Hot vector
Move-Forward	$\begin{vmatrix} a \\ 0 \\ 0 \\ 0 \end{vmatrix}$	$\begin{vmatrix} b \\ 0 \\ 0 \\ 0 \end{vmatrix}$
Sharp-Right	$\begin{vmatrix} 0 \\ a \\ 0 \\ 0 \end{vmatrix}$	$\begin{vmatrix} 0 \\ b \\ 0 \\ 0 \end{vmatrix}$
Slight-Left	$\begin{vmatrix} 0 \\ 0 \\ a \\ 0 \end{vmatrix}$	$\begin{vmatrix} 0 \\ 0 \\ b \\ 0 \end{vmatrix}$
Slight-Right	$\begin{vmatrix} 0 \\ 0 \\ 0 \\ a \end{vmatrix}$	$\begin{vmatrix} 0 \\ 0 \\ 0 \\ b \end{vmatrix}$

The selection of the value of a and b was done empirically. Several experiments were performed using different values of a and b. For testing the performance of every experiment, Leave-One-Out validation algorithm was applied. Figure 3 shows the results of these experiments. From this figure, we can observe that the value of 3 gave the best performance. When the value is higher than 3 the performance decreased.

We present an illustrative example of the process for building *max* and *min* morphological associative memories.

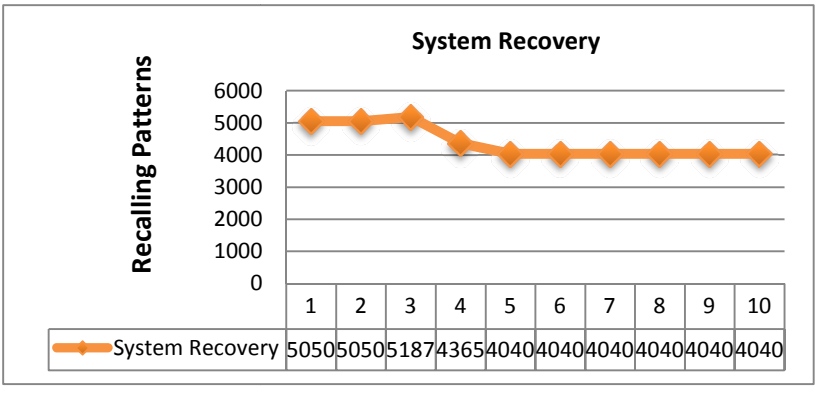

Fig. 3. Results of the performance of the classifier using different values for *a* and *b*

Example: Let be the four following associations using original values from dataset. We build *max* morphological associative memory using equation (5).

$$x^1 = \begin{bmatrix} 1.344 \\ 0.496 \\ 2.843 \\ 0.692 \end{bmatrix} \rightarrow y^1 = \begin{bmatrix} 3 \\ 0 \\ 0 \\ 0 \end{bmatrix}, x^2 = \begin{bmatrix} 0.753 \\ 0.457 \\ 2.323 \\ 0.442 \end{bmatrix} \rightarrow y^2 = \begin{bmatrix} 0 \\ 3 \\ 0 \\ 0 \end{bmatrix},$$

$$x^3 = \begin{bmatrix} 1.581 \\ 1.387 \\ 1.815 \\ 2.064 \end{bmatrix} \rightarrow y^3 = \begin{bmatrix} 0 \\ 0 \\ 3 \\ 0 \end{bmatrix}, x^4 = \begin{bmatrix} 1.687 \\ 0.445 \\ 2.332 \\ 0.429 \end{bmatrix} \rightarrow y^4 = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 3 \end{bmatrix}$$

$$M = \begin{bmatrix} 1.656 & 2.504 & 0.157 & 2.308 \\ 2.247 & 2.543 & 0.677 & 2.558 \\ 1.419 & 1.613 & 1.185 & 0.936 \\ 1.313 & 2.555 & 0.668 & 2.571 \end{bmatrix}$$

We recalled pattern $\mathbf{x}^1$ as follows.

$$M\Delta x^1 = \begin{bmatrix} 1.656 & 2.504 & 0.157 & 2.308 \\ 2.247 & 2.543 & 0.677 & 2.558 \\ 1.419 & 1.613 & 1.185 & 0.936 \\ 1.313 & 2.555 & 0.668 & 2.571 \end{bmatrix} \Delta \begin{bmatrix} 1.344 \\ 0.496 \\ 2.843 \\ 0.692 \end{bmatrix} = \begin{bmatrix} 3 \\ \mathbf{3.039} \\ 1.628 \\ 2.657 \end{bmatrix}$$

The results for the remaining patterns are

$$M\Delta x^2 = \begin{bmatrix} 2.409 \\ \mathbf{3} \\ 1.378 \\ 2.066 \end{bmatrix}, \quad M\Delta x^3 = \begin{bmatrix} 1.972 \\ 2.492 \\ \mathbf{3} \\ 2.483 \end{bmatrix} \text{ and } M\Delta x^4 = \begin{bmatrix} 2.489 \\ 2.987 \\ 1.365 \\ \mathbf{3} \end{bmatrix}$$

From each vector, we looked for the maximum value, in the first case, this value corresponds to 3.039 then we wrote a 3 in that component and we wrote 0 in the remaining components. This process is accomplished for the other vectors. The results are shown as follows.

$$\begin{bmatrix}0\\3\\0\\0\end{bmatrix} = y^1, \quad \begin{bmatrix}0\\3\\0\\0\end{bmatrix} = y^2, \quad \begin{bmatrix}0\\0\\3\\0\end{bmatrix} = y^3, \quad \begin{bmatrix}0\\0\\0\\3\end{bmatrix} = y^4$$

We can observe that our proposal classified the latter 3 patterns correctly but it could not classify the first pattern.

A similar process is performed for *min* memories, but in that case, the minimum value is obtained and a -3 is written in that position and in the others we wrote a 0.

Some of the patterns that *max* memory cannot classify then *min* memory classify them, but there are patterns that none of them can do it.

4 Experiments and Results

The algorithm was implemented with the programming language Microsoft Visual C# 2010 Express Edition ® and was tested on a PC with CORE i5® processor and 4 GB of RAM memory, the operating system was Microsoft Windows 7®.

Using the algorithm for classifying described in the latter section, we carried out several experiments for testing the performance of our proposal. Three methods were used: Leave One Out, Hold-Out, and K-Fold Cross-Validation.

In the three cases, two sets had to be arranged: one set for training and the other set for testing. For Leave-One-Out, we took 5405 elements for training the morphological associative memory and one element for testing. The first step was to take the first element for testing and the remaining records for training, then, we took the second element as the testing set and the remaining elements as training set, and so on. This process was performed 5406 times.

For Hold-Out test, we selected randomly the elements contained in each set. The size of the sets was varying from 10% for training and 90% for testing to 90% for training and 10% for testing. For each size we performed 15 calculations.

Finally, we used K=10 for K-Fold Cross-Validation, then the whole database was divided into 10 sets with the same number of elements. We took the first set K1 for testing and the remaining sets were for training, afterwards, the set K2 was used for testing and the others for training and so on, the elements for every set were selected randomly. We performed 20 calculations for every test. Table 2 shows the summary of the results from every test.

From Table 2 we can observe that the average of the best results from the three test (using 70-30 for Hold Out) was 96.38% of effectiveness and for the worst results the average was 93.24%.

But we cannot assure this result is good or bad without comparing it with other classification algorithms.

Table 2. Results of the performance of our proposal using three methods: Leave-One-Out, Hold-Out and 10-Fold Cross-Validation

Test	Best Result	Worst result	Average
Leave One Out(5405-1)	95.94894562	95.94894562	95.94894562
Hold out 10-90 (540-4866)	94.6568023	65.92683929	80.2918208
Hold out 20-80 (1081-4325)	95.69942197	64.11560694	79.90751446
Hold out 30-70 (1802-3604)	96.00443951	80.13318535	88.06881243
Hold out 40-60 (2160-3246)	96.30314233	75.35428219	85.82871226
Hold out 50-50 (2703-2703)	96.55937847	81.94598594	89.25268221
Hold out 60-40 (3240-2166)	95.9833795	79.45521699	87.71929825
Hold out 70-30 (3780-1626)	**96.37146371**	**86.77736777**	**91.57441574**
Hold out 80-20 (4320-1086)	96.40883978	80.84714549	88.62799264
Hold out 90-10 (4860-546)	96.88644689	93.58974359	95.23809524
K Fold Cross (70%-30%)	**96.85185185**	**87.5925926**	**92.22222223**

Table 3 shows the results from four works [21] which utilized the same database that we used, in their experiments. The algorithms applied were: Perceptron, Mixture of experts, multilayer perceptron, and Elman network.

Table 3. Results of effectiveness from four methods and the morphological associative memory

	Logic Perceptron	Mixture Of Experts	Multilayer Perceptron	Recurrent Elman Network	Morphological Associative Memory
Best result	14.79%	67.85%	**97.96**%	96.42%	96.37%
Worst result	13.55%	67.77 %	82.07%	76.52%	**86.77**%
Average	21.56%	67.81%	90.01%	86.47%	**91.57**%

From Table 3, it can be observed that the best result is showed by the Multilayer perceptron method with 97.96% of effectiveness and it is better than our proposal with 1.59 of effectiveness. However, our worst result is higher that multilayer perceptron in a 4.7%. Besides, the average of morphological associative memory is the best with 91.57% surpassing the second best algorithm by 1.56%.

5 Conclusions

Wall-Following problem is a current problem due to the continuous development of walking robots interacting with us in our daily life. Therefore, it is important to have a

reliable navigation system for robots in order to be able to coexist with them avoiding collisions.

Wall-Following is a non-linear problem and it can be seen as a classification problem and it can be solved by the use of some methods such as: *K*-Nearest Neighbors, Rule-based systems, Bayesian models, and any other method which can deal with non-linear problems.

Morphological Associative models have shown to be an option as a tool for many applications, one of them is classification. The latter experiments demonstrated that the associative approach shows competitive results in comparison with neural network approach.

Our proposal is an algorithm that is easy to understand and to implement. It is based on the fundamentals of morphological mathematics and associative memory approach. The value of a and b for positive-hot and negative-hot vectors have to be obtained empirically. As a future work, we pretend to applied our proposal to other data in order to find a formal way to assign the values to a and b.

Acknowledgments. The authors would like to thank the Instituto Politécnico Nacional (COFAA and SIP), and SNI for their economical support to develop this work.

References

1. ASIMO The World's Most Advanced Humanoid Robot, http://asimo.honda.com/asimo-specs/
2. Murata Manufacturing Co., Ltd., http://www.murata.com/corporate/boy_girl/index.html
3. RoboCup, http://www.robocup.org/robocup-home/
4. IEEE Spectrum, Inside Technology, http://spectrum.ieee.org/automaton/robotics/humanoids/iran-humanoid-robot-surena-2-walks-stands-on-one-leg
5. CNET News, http://news.cnet.com/8301-17938_105-20016709-1.html
6. Ross, J.S., Daida, J.M., Doan, C.M., Bersano-Begey, T.F., McClain, J.J.: Variations in Evolution of Subsumption Architectures Using Genetic Programming: The Wall Following Robot Revisited. In: Genetic Programming: Proceedings of the First Annual Conference, July 28 (1996)
7. Braunsting, R., Sanz, P., Ezkerra, J.M.: Fuzzy Logic Wall Following of a Mobile Robot Based on the Concept of General Perception. In: 7th International Conference on Advanced Robotics, ICAR 1995, pp. 367–376 (1995)
8. DeSouza, G.N., Kak, A.C.: Vision for Mobile Robot Navigation: A Survey. IEEE Transactions on Pattern Analysis and Machine Intelligence 24(2), 237–267 (2002)
9. Borenstein, J., Koren, Y.: Real-time Obstacle Avoidance for Fast Mobile Robots in Cluttered Environments. Reprint of Proceedings of the 1990 IEEE International Conference on Robotics and Automation, pp. 572–577 (1990)
10. Carelli, R., Oliveira, F.E.: Corridor navigation and wall-following stable control,for sonar-based mobile robots. Robotics and Autonomous Systems 45, 235–247 (2003)
11. Meisburger, S., Hubler, A.: Chaos in Wall Following Robots (September 2006)

12. Bullen IV, H.W., Ranjan, P.: Chaotic transitions in Wall-Following Robots (August 2009)
13. Sadati, N., Taheri, J.: Solving Robot Motion Planning Problem Using Hopfield Neural Network In A Fuzzified Environment. In: Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, pp. 1144–1149 (2002)
14. Huang, W.H., Beevers, K.R.: Topological Mapping with Sensing-limited Robots. In: Sixth International Workshop on the Algorithmic Foundations of Robotics (WAFR 2004), pp. 1–16 (2004)
15. Chung, T.L., Bui, T.H., Kim, S.B., Oh, M.S.: Wall-Following Control of a Two-Wheeled Mobile Robot. KSME International Journal 18(8), 1288–1296 (2004)
16. Mehta, S.: An Autonomous Wall Following Robot, Department of Electrical and Computer Engineering Cleveland State University Cleveland, Ohio 44115 (2008)
17. Gavrilut, I., Tiponut, V., Gacsadi, A., Tepelea, L.: Wall-following Method for an Autonomous Mobile Robot using Two IR Sensors. In: 12th WSEAS International Conference on Systems, Heraklion, Greece, July 22-24 (2008)
18. Huang, L.: Wall-following control of an infrared sensors guided wheeled mobile robot. International Journal of Intelligent Systems Technologies and Applications 7(1), 106–117 (2009)
19. Lamperski, A.G., Loh, O.Y., Kutscher, B.L., Cowan, N.J.: Dynamical Wall-Following for a Wheeled Robot, using a Passive Tactile Sensor. In: Proceedings of the 2005 IEEE International Conference on Robotics and Automation, ICRA 2005, pp. 3838–3843 (2005)
20. UC Irvine Machine Learning Repository, `http://archive.ics.uci.edu/ml/`
21. Freire, A.L., Barreto, G.A., Veloso, M., Varela, A.T.: Short-term memory mechanisms in neural network learning of robot navigation tasks: A case study. In: 6th Latin American Robotics Symposium (LARS), pp. 1–6 (2009)
22. Yáñez-Márquez, C.: Associative Memories Based on Order Relations and Binary Operators (In Spanish). PhD Thesis. Centro de Investigación en Computación, Mexico (2002)
23. Steinbuch, K.: Die Lernmatrix. Kybernetik 1(1), 36–45 (1961)
24. Willshaw, D., Buneman, O., Longuet-Higgins, H.: Non-holographic associative memory. Nature 222, 960–962 (1969)
25. Anderson, J.A.: A simple neural network generating an interactive memory. Mathematical Biosciences 14, 197–220 (1972)
26. Hopfield, J.J.: Neural networks and physical systems with emergent collective computational abilities. Proceedings of the National Academy of Sciences 79, 2554–2558 (1982)
27. Ritter, G.X., Sussner, P., Diaz de León, J.L.: Morphological Associative Memories. IEEE Transactions on Neural Networks 9, 281–293 (1998)
28. Wall-Following Robot Navigation Data Data Set (2010), `http://archive.ics.uci.edu/ml/datasets/Wall-Following+Robot+Navigation+Data` (released)
29. Navarro, R., Pineda, G.: Solution to the Wall-Following problem by using Morphological Associative Memories, Thesis, Escuela Superior de Ingeniería Mecánica y Eléctrica, Mexico City (2011)
30. Acevedo, M.E., Yáñez, C., López, I.: Alpha-Beta Bidirectional Associative Memories: Theory and Applications. Neural Processing Letters (26), 1–40 (2007)