Pergamon

Neural Networks, Vol. 8, No. 9, pp. 1439–1456, 1995
Copyright © 1996 Elsevier Science Ltd
Printed in Great Britain. All rights reserved
0893–6080/95 $9.50+.00

0893-6080(95)00065-8

CONTRIBUTED ARTICLE

Multiple Elastic Modules for Visual Pattern Recognition

SOHEIL SHAMS

Hughes Research Laboratories

(*Received* 1 *November* 1994; *revised and accepted* 11 *April* 1995)

Abstract—*Fast synaptic plasticity, used to associate topologically ordered features in an input image to those of previously learned objects, has been previously proposed as a possible model for object recognition (von der Malsburg & Bienenstock, 1987, Europhysics Letters, 3(11), 1243–1249). In this paper, it is argued that in addition to rapid link dynamics, fast receptive field size dynamics are necessary to automatically escape from poor local matches and also allow for simultaneous recognition of multiple objects. Furthermore, a feature locking mechanism with a properly designed hysteresis property is needed to handle complex, cluttered, and dynamic scenes. The multiple elastic modules (MEM) model, described in this paper, utilizes newly developed dynamics that locate and recognize a previously learned object based on expected spatial arrangement of local features. The MEM model can be viewed as using a deformable template of an object to search the input scene. Unlike many of the current artificial neural network models, the proposed MEM model attempts to capture many of the functions available in the biological visual system by providing mechanisms for: multi-modal feature integration, generation and maintenance of focus of attention, multi-resolution hierarchical searching, and top-down expectation driven processing coupled with bottom-up feature activation processing. In addition, the MEM dynamics, unlike similar template matching approaches (Konen et al., 1994, Neural Networks, 7(6/7), 1019–1030; Yuille et al., 1992, International Journal of Computer Vision, 8(2), 99–111), does not converge to false objects when there are no sufficiently familiar objects in the scene. The performance of the MEM model in detection and recognition of objects through a number of computer simulations is demonstrated.*

Keywords—Self-organization, Labeled graph matching, Combinatorial optimization, Dynamic focus of attention, Automatic target recognition, Deformable templates.

1. INTRODUCTION

It has been shown that an ensemble of simple processing elements can solve computationally difficult problems through collective computation (Hopfield, 1982; Durbin & Willshaw, 1987; Kohonen, 1987). A number of models have been proposed for formulating visual object recognition using such collective computation (Fukushima, 1987; von der Malsburg & Bienenstock, 1987; Nasrabadi & Li, 1991). In visual pattern recognition, objects can be defined as patterns of local relationships between local multi-dimensional features. It is therefore possible to store a labeled graph representation of an object having each node label indicate the sensitivity to a particular feature and the links represent the expected relative arrangement of these features (see Figure 1). In order to achieve robustness with respect to small deformation, the links of the graph are made elastic. These elastic links can accommodate a certain level of smooth local elongations and contractions to the model graph, the extent of which is dependent on the elasticity of the link. Although I do not directly address the issue

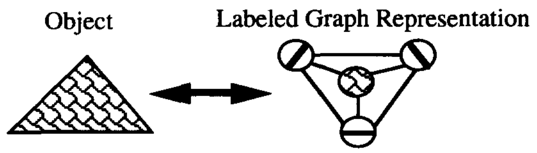


FIGURE 1. A triangular object with a specific texture can be represented by three oriented edge neurons and a texture sensitive neuron. The expected spatial organization of the features are encoded in the interconnection links between the neurons.

Acknowledgements: I wish to thank Patricia A. Keaton for her help in developing and executing the gabor filter routines and for helpful discussions and review of this manuscript. I would also wish to thank Bruce R. Hoff, Yuri Owechko, Kurt Reiser, Ladan Shams, and Bernard H. Soffer for their helpful comments and review of the manuscript and Alex Gebbie and David Kelly, of the Hughes Missile Systems Company, for their support of this project. A final thanks to the anonymous reviewers for their helpful comments and suggestions.

Requests for reprints should be sent to S. Shams, Hughes Research Laboratories, 3011 Malibu Canyon Road, Malibu, CA 90265, USA; E-mail: shams@maxwell.hrl.hac.com.

of learning these model graphs in this paper, it seems rather plausible to construct these graphs using a Hebbian style learning procedure (Reiser, 1991). The process will involve forming links between labeled nodes which are consistently observed in a particular spatial arrangement. The variance in the spatial organization of linked features, over repeated observations, can automatically adjust the elasticity of the links connecting the features through the learning process. Features with very small variations in their spatial relationships, over many observations, will have a very rigid elastic connection, whereas those with relatively large variations in their spatial arrangement, will have looser elastic connections to accommodate for these variations.

Having stored a number of learned objects as labeled graphs, the problem of object detection and recognition can be transformed into finding a "good" match between the stored graph labels and the visual scene, while preserving the topological arrangement of the graph nodes (von der Malsburg, 1988). This type of graph matching problem has been termed the subgraph isomorphism problem in combinatorial optimization. Finding exact solutions to the subgraph isomorphism problem has been shown to be *NP-hard*, and thus computationally impractical for graphs with a large number of nodes. Therefore, most approaches to solving this problem have been heuristics which attempt to find acceptable "close to optimal" solutions, albeit not the exact optimal solution. The self-organizing MEM model, which I describe here, has been successfully applied to a number of large combinatorial optimization problems including labeled graph matching as well as a multi-sensor multi-target deghosting and tracking problem with a solution space of 10^{11} possibilities (Shams, 1994). One of the labeled graph matching problems, described later in this paper, involves the assignment of a 123 node labeled graph to a 13,000 node labeled graph. In a conventional relaxation based optimization technique [e.g., Hopfield net (Hopfield & Tank, 1985)], an explicit $123 \times 13{,}000$ assignment matrix would be needed. This translates into roughly 1.6 million neurons and $\sim 10^{12}$ synapses. Neural networks of this scale would be practically ineffective for real-world applications.

There are a number of novel features in the proposed MEM model that enables its application to such large scale optimization problems. The most fundamental of these is the explicit enforcement of *a priori* knowledge constraints on the search space. The significant improvement in performance of neural network optimization techniques, through the use of strongly enforced constraints, has been previously demonstrated through statistical physics techniques (Simic, 1990, 1991). In the MEM model, the explicit assignment matrix, used by relaxation networks, is eliminated, thus considerably decreasing the required memory size and significantly reducing the convergence time (Mjolsness, 1995). Also, the self-organizing dynamics of the MEM model, continuously enforce specific constraints on the arrangement of viable assignments. The formulation of a dynamic and local receptive field for each neuron also strongly enforces constraints to limit the search area to a local neighborhood of likely solutions. The use of labeled graph nodes introduces further constraints which are fully exploited by the MEM model to limit the search space. In addition to the explicit enforcement of these problem constraints, the MEM model utilizes a number of other techniques to achieve its rapid convergence and high recognition accuracy. These include annealed stochastic search [similar to simulated annealing (Kirkpatrick et al., 1983)], simultaneous search for multiple objects, a new mechanism for adaptive focus-of-attention (through rapid synaptic modulation of receptive field location and size), and a new method of simultaneous top-down and bottom-up multi-resolution hierarchical matching.

In Section 2 of this paper, I describe the details of the MEM model. The application of the MEM model to visual object recognition is presented in Section 3, with simulation results of single and multiple object recognition. Current work on the MEM model and future research directions are given along with the concluding remarks in Section 4.

2. ELASTIC MODEL MATCHING

The basic idea behind the multiple elastic modules (MEM) algorithm for object recognition is to find close to optimal matches between a number of previously learned disjoint elastic graphs (deformable models) and the pattern of activity in the visual field. The system is organized into three layers as shown in Figure 2. The first layer, L0, functions similarly to the retina by receiving the input image. The next layer, L1, consists of position-sensitive feature-specific cells which are followed by the final object memory layer, L2, where the model graphs are stored. The stored graphs are constructed from a set of labeled nodes (neurons) i with feature labels f_i, where each feature f_i indicates a level of sensitivity to a particular local feature (e.g., orientation, color, texture, etc.). The nodes of the model graphs, in layer L2, are interconnected through a set of connections g_{ij}, where $g_{ij} = 1$ if neurons i and j are connected, and $g_{ij} = 0$ if they are not connected. A set L_i is defined to represent all the neurons connected to neuron i. The relative spatial arrangement of neurons, as defined by the stored model graph, can be encoded in the links of the connected neurons by a vector $\boldsymbol{\delta}_{ij}$ between each neuron i and neurons $j \in L_i$. It is important to note

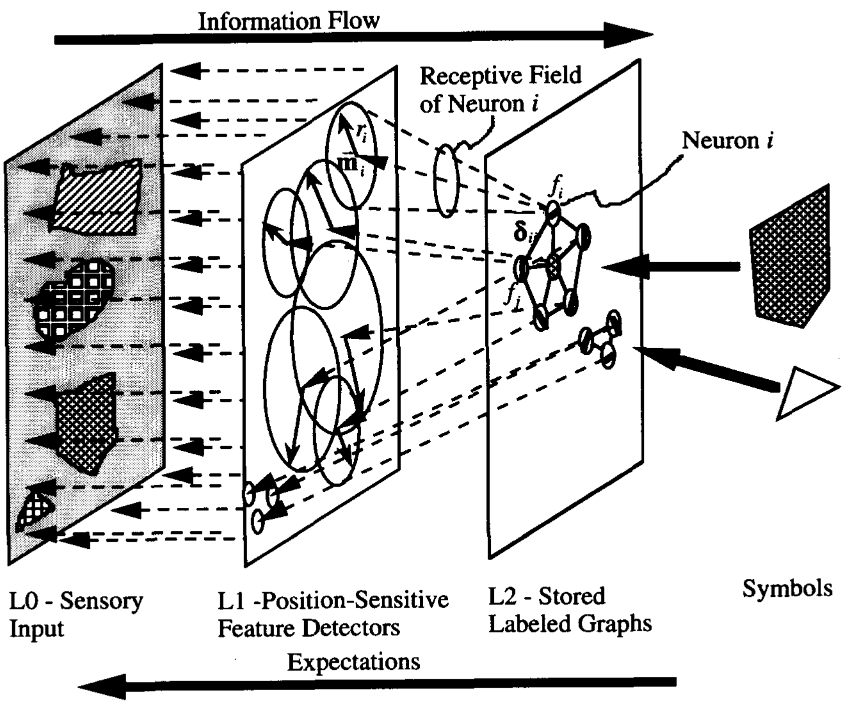


FIGURE 2. Two objects are stored in memory (layer L2). They are represented as labeled graphs with spatial relationships enforced through δ_{ij}. The input image is placed on layer L0 (retina). The cells in layer L1 have fixed receptive field connections to cells of layer L0. The L1 cells are activated by the presence of specific features (e.g., a short horizontal line) at corresponding spatial locations in the image plane (L0). The model graphs in layer L2 can receive input from a dynamically changing local receptive field area of cells in layer L1. The receptive field of each cell i in layer L2 is specified by the vector m_i, indicating its center and r_i indicating its radius. The dynamics of the MEM algorithm moves the receptive fields of a connected graph towards active cells in layer L1 with corresponding feature labels. When the receptive field centers of the model graph neurons m_i are properly aligned with cells in the L1 layer (i.e., having similar topological arrangement of feature labels), the receptive field sizes r_i of these L2 neurons are reduced causing the network attention to focus more closely on the object. This is illustrated by the triangular object in this figure.

that g_{ij} specifies whether or not two neurons are interrelate, and δ_{ij} specifies how they are related. For example, $\delta_{ij}=0$ and $g_{ij}=1$ would indicate that the features associated with neurons i and j, $f_i=$ "green" and $f_i=$ "vertical edge" for instance, should be found at the same spatial location.

There are three primary variables associated with each neuron of layer L2 which are used for locating and recognizing objects in the input scene. These are, $\mathbf{m}_i$, the receptive field center of neuron i projected onto layer L1, r_i, the radius of the receptive field, and h_i, the level of locking or binding between neuron i and the input. The neural dynamics associated with $\mathbf{m}_i$ in the MEM model is similar in concept to the topology preserving Kohonen's self-organizing maps (Kohonen,1987). The algorithm randomly selects a single position sensitive neuron from layer L1 at location $\mathbf{x}$ with feature sensitivity $\mathcal{F}(\mathbf{x})$. In order for this selected L1 cell to have any effect on any L2 neuron, it must fall within the receptive field of at least one L2 neuron. To increase processing speed, in the simulation results reported in the following section, the L1 cell is randomly selected from the area encompassed by the largest receptive field r_i of all L2 neurons. All neurons in L2 whose receptive fields encompass the selected L1 neuron, compete using a winner-take-all (WTA) mechanism based on the distance between their receptive field centers $\mathbf{m}_j$ and the position of the selected L1 neuron $\mathbf{x}$, as well as their respective feature similarity. This competition can be formally represented as

$$\mathbf{m}_i^* = \min_{j \in A}\{\|\mathbf{m}_j - \mathbf{x}\| R(\mathcal{F}(\mathbf{x}), f_j)\}, \tag{1}$$

where $\mathbf{m}_i^*$ is the receptive field center of the "winning" neuron, A is the set of all competing neurons, and $R(x,y)$ is a bounded monotonic nonlinear measure of the similarity between two features x and y having a range (0,1), with small values of R denoting close similarity. The multiplicative combination of the Euclidean distance value and feature similarity is one of many possible combinations of these two terms. This particular formulation was used because it offered good performarnce (i.e., fast and reliable detection and recognition) on simulation experiments. A more rigorous analytical evaluation of the effects of this competition function on network performance is planned for a later study.

The dynamics of $\mathbf{m}_i$ is defined by the self-organizing rule

$$\mathbf{m}_i^* \leftarrow \mathbf{m}_i^* + \alpha[\mathbf{x} - \mathbf{m}_i^*], \tag{2}$$

where α is the update rate of the winning neuron. In the MEM model, the update rate α is scaled proportional to the similarity of the feature labels between the winning neuron and the selected L1 neuron. The closer the similarity, the higher the update rate. This is accomplished formally by having $\alpha = \alpha_o[1 - R(\mathscr{F}(\mathbf{x}), f_i^*)]$, where $\alpha_o < 1$ is the maximum update rate. In the simulations described in Section 3, a relatively large $\alpha_o = 0.8$ value was used to achieve rapid recognition.

A fundamental principle of the MEM model is to implement topology preserving dynamics on the updating of the receptive field centers $\mathbf{m}_i$ of L2 neurons. More specifically, when a receptive field center of a neuron i is updated, the receptive field centers of all its connected neighboring neurons, $\mathbf{m}_j$ with $j \in L_i$, should be updated such that the desired topological relationships between these neurons, $\boldsymbol{\delta}_{ij}$, are preserved. A novel feature of the MEM model is that unlike other topology preserving self-organizing models, such as Kohonen's SOM (Kohonen, 1987), where only those neurons within a specific progressively shrinking range of the winning neuron are updated, the MEM model updates all the neurons in the connected subgraph of the winning neuron. As described earlier, a connected subgraph in the MEM model refers to a graph representation of a single object, such as the triangle in Figure 2. The updating procedure propagates the change in the neuron receptive field center of the winning neuron $\mathbf{m}_i^*$ [given by eqn (2)] throughout the connected portion of the graph while attempting to preserve the expected spatial relationships $\boldsymbol{\delta}_{ij}$ between the neurons. The specific formulation is given by

$$\mathbf{m}_{j(u+1)} \leftarrow \mathbf{m}_{j(u+1)} + \alpha'_{j(u+1)}[\mathbf{m}_{j(u)} + \boldsymbol{\delta}_{j(u)j(u+1)} - \mathbf{m}_{j(u+1)}], \tag{3}$$

where neuron $j(u+1) \in L_u$ and $0 \le u \le K - 1$ with K being the diameter of the connected graph and u the depth of the graph from the winning neuron, with $\mathbf{m}_{j(0)} = \mathbf{m}_i^*$. In other words, after the winning neuron's receptive field center is updated according to eqn (2), all of the neurons directly connected to the winning neuron ($u = 1$) are updated according to eqn (3) followed by all the neurons connected to these neurons ($u = 2$), and so on following a breadth-first propagation strategy. In computer simulations, it has been computationally advantageous not to update those neurons which are changed by less than a specific threshold level without loss in the accuracy and speed of the recognition process. In addition, by limiting the application of eqn (3) only to those neuron receptive fields which are noticeably modified (more than 1 pixel for example), the amount of computation used during the "fine-tuning" stages of the recognition is considerably reduced causing a faster convergence. The update rate of non-winning neurons α'_j is dynamically calculated as a monotonically increasing nonlinear function of the local deformation of the graph around neuron j, specified by p_j. The local graph deformation about a neuron i is calculated as

$$p_i = \frac{\sum_{j \in L_i} \|(\mathbf{m}_i - \mathbf{m}_j) - \boldsymbol{\delta}_{ij}\|}{|L_j|}. \tag{4}$$

One of the most important aspects of the MEM algorithm is to associate a specific dynamically changing receptive field size r_i, with each neuron in layer L2. The size of the neuron receptive field controls and limits the effect of various input stimuli on each neuron. Collectively, all the receptive fields of the neurons, associated with a particular model graph, determine its focus-of-attention. The larger the receptive field of a neuron in L2, the less likely it has bound to a particular input feature. Therefore, during the initially random state of the network, all the neurons have very large receptive fields (on the order of the field of view). As time progresses, the neural dynamics will reduce the receptive fields of the neurons to focus in on the areas where correct matching is most likely. Recognition is established when the receptive field sizes of all the neurons in the model graph are close to zero. If no sufficiently good match is found between the stored models and the input scene, the network dynamics will keep the receptive fields in a rather large state and the network will continue indefinitely to search for a familiar object. The receptive field size dynamics associated with each neuron i is represented as:

$$r_i = a(t)p_i h_i + e_i + \varepsilon_i \tag{5}$$

where $a(t)$ is a monotonically decreasing scaling parameter, similar to the annealing parameter K used in the elastic net algorithm (Durbin & Willshaw, 1987), e_i is an expectation variable which is described later in detail, and ε_i is a small constant specifying the minimum receptive field size. It should be noted that unlike commonly used annealing schedules which converge to zero, in the MEM model, $a(t)$ starts at a high value and asymptotically approaches a value larger than one. In the simulations described in Section 3, a value of $a(0) = 8$ and $a(\infty) = 3$ were selected. More detailed discussion on this parameter, as well as others used in the MEM model are given in the Appendix. Additionally, the value of ε_i for a neuron i is set equal to the feature size of its corresponding feature f_i. For example, if f_i represents a small horizontal line being detected by an L1 neuron, with a fixed receptive field size of three pixels, ε_i is set equal to 3.

It can be seen from eqn (5) that the receptive field size is a direct function of the amount of local deformation p_i, scaled by the locking variable h_i (described in detail below) having a range (0,1). The rationale is that we expect a good match between a stored graph and the input image to cause only a moderate amount of deformation of the elastic graph. If the local deformation is large, then the network probably has not found a good match and should have a wide search area. On the other hand, if p_i is small, it is likely that a good match is in the local spatial neighborhood and the receptive field size is therefore reduced dynamically according to eqn (5). The variable h_i is used to implement a feature "locking" mechanism having a small value if neuron i is bound to a specific point in the image at spatial location $\mathbf{m}_i$. This variable reduces the receptive field size of the neuron when the rate of change in the neural dynamics associated with $\mathbf{m}_i$ falls below a certain rate. The value of h_i can be calculated as

$$h_i = S(\gamma_1 l_i) \text{ and } h'_i = S(\gamma_2 l_i), \text{ with } \gamma_2 < \gamma_1, \tag{6}$$

where

$$\frac{dl_i}{dt} = -\tau_h h'_i + \delta_h(1 - h'_i), \tag{7a}$$

and

$$\frac{dl^*_i}{dt} = -\tau^* \alpha h'_i + \frac{d\mathbf{m}^*_i}{dt}(1 - h'_i). \tag{7b}$$

In eqn (6), S is the sigmoid squashing function of the form $S(x) = 1/(1 + e^{-x})$, γ_1 and γ_2 are used to adjust the gain (steepness) of the sigmoid functions, and l_i is the output of a leaky integrator with a decay constant τ_h and a constant positive input δ_h, as defined by eqn (7a). The rate of change in l^*_i for the winning neuron is calculated slightly different as defined by eqn (7b). The ratio between τ_h and δ_h determines the resting value of h_i. For example, if $\tau_h = \delta_h$ then h_i has a tendency to converge to a value of 0.5. The magnitude of these parameters (τ_h and δ_h determines the convergence rate to this resting value (more details on these parameters are given in the Appendix). If a neuron i is never selected as a winning neuron, by eqn (1), its corresponding locking value h_i will over time converge to the resting value determined by τ_h and δ_h. Such a case will only occur if the feature represented by this neuron, f_i, a long vertical line associated with the left side of a rectangle for instance, is not within the receptive field of neuron i. This situation arises either by not having the model graph's focus-of-attention properly aligned with the object in the scene or occlusion of the feature by a different object when proper alignment of the model graph's focus-of-attention has been established. In either case, the locking parameter will have a "non-committed" value (e.g., $h_i = 0.5$). On the other hand, if neuron i is selected as the winning neuron by eqn (1), its locking value is updated according to eqn (7b) and (6). If the receptive field center $\mathbf{m}_i$ is close to the selected L1 neuron's location $\mathbf{x}$, causing a small change in the receptive field center (small $d\mathbf{m}^*_i/dt$), and the corresponding feature labels [f_i and $\mathcal{F}(\mathbf{x})$] are similar, leading to a large update rate α, then the focus of attention of neuron i will be tightened around the receptive field center $\mathbf{m}_i$. The change in receptive field center $d\mathbf{m}^*_i/dt$ is consecutively small, only if its locally connected neighbors, $j \in L_i$, have located corresponding matching feature labels in the appropriate topological arrangement. In other words, when a good local match of the model graph and the input image is found, $d\mathbf{m}^*_i/dt$ will be small. If the local arrangement of features in the input does not match the model graph, large deformation of the model graph will increase the receptive field of the neurons, which in turn leads to a larger rate of change in the receptive field centers $d\mathbf{m}^*_i/dt$. This will cause an increase in the locking parameter values h_i which further broadens the focus-of-attention of the model graph to search for more likely match sights. These dynamics are illustrated through a number of example simulations in the next section. The parameter τ^* in eqn (7b) is used to adjust the maximum amount of movement in $\mathbf{m}^*_i$ tolerated while maintaining lock.

The variable h'_i in eqns (6) and (7) is introduced to limit the growth and decay of l_i; in effect controlling the degree of hysteresis of the locking parameter h_i. Since the activation of L1 neurons (inputs to the L2 neurons) is performed randomly, the hysteresis of h_i is used to keep a locked neuron focused until the next time the neuron is selected as the winner. Since we expect a neuron to be selected as the winning neuron roughly once every N iterations, where N is the number of neurons in L2, the amount of hysteresis is set to compensate for the effects of eqn (7a) such that a locked neuron will remain locked for at least N iterations without being selected by eqn (1), the details of the parameter setting are given in the Appendix. In addition this hysteresis mechanism helps avoid locking to spurious features by accepting only a small movement of $\mathbf{m}^*_i$. This feature is also helpful in the dynamic scene environment where the object which has been recognized is slowly moving in the field of view. Small movements in the location of input features, as represented by $d\mathbf{m}^*_i/dt$ will be tolerated without losing lock. Therefore, the focus of attention

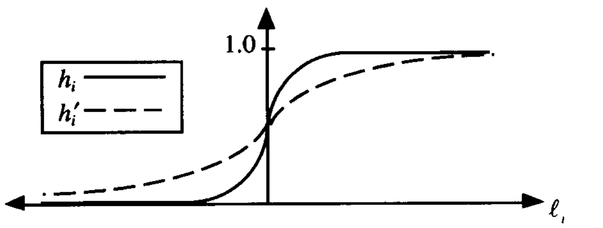


FIGURE 3. The magnitude of the locking value h_i and its associated limiting value h_i' as a function of l_i. As neuron i locks to a particular point in the image $h_i \to 0$. The h_i' value limits the reduction of l_i according to eqn (7). The h_i' value in effect limits the extent of the l_i in either the positive or negative directions.

for each object will gradually move with the object. The implementation of the hysteresis mechanism is depicted in Figure 3 and described in its caption.

Whereas the scaling parameter $a(t)$ and the locking parameter h_i are used to shrink the receptive field size, the variable e_i in eqn (5) is used to expand it. This variable controls the "expectation" of neuron i in finding a desired feature f_i at location $\mathbf{m}_i$. If the desired feature is within the receptive field of neuron i, then this neuron will be regularly successful at being selected for updates according to eqns (1) and (2). On the other hand, the dynamics of the system are designed such that if the desired feature is not found within the receptive field, its size r_i will be gradually enlarged until the correct feature is found. This is accomplished through the expectation variable e_i. This variable allows the model to escape from poor matches, where only a few neurons in the graph have been correctly matched to features in the input in search of better solutions. The dynamics of the expectation variable is given by

$$e_i = Q(k_i), \tag{8}$$

where Q is a bell-shaped function and $k_i = 0$, if $m^* = m_i$, otherwise $(dk_i/dt) = 1$ (see Figure 4). In other words, k_i is continuously being incremented, which causes the expectation value to grow. If neuron i is selected by eqn (1), k_i will be reset to its base value of zero and the expectation value will be correspondingly reduced. As previously stated, each neuron in layer L2 is expected to be selected about once every N iterations. Therefore, the expectation value remains

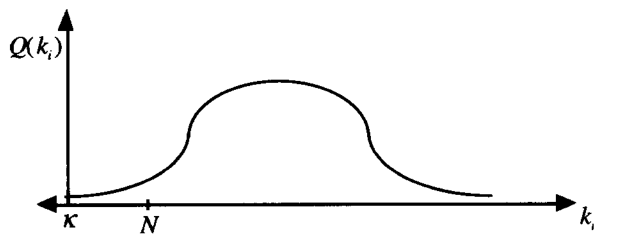


FIGURE 4. The expectation value e_i as a function of iteration counter k_i.

fairly small for about N iterations, where it begins to rapidly increase to a large value several times the minimum receptive field size ε_i. However, if the neuron is not selected after a relatively long period, on the order of several times N iterations, the expectation value will begin to decline since the expected feature is most likely either missing or occluded, hence the bell-shape form of the Q function.

The various steps of the MEM model, along with their corresponding computational complexity, are summarized below. Note that N indicates the number of neurons in layer L2 (number of nodes in all model graphs) and M indicates the number of nodes used in a single model graph (M could be approximated as N/L, with L being the number of model graphs in the memory). In addition, it is assumed that the model graphs are sparsely connected. This is a valid assumption since most graphs have only local interconnections.

1. Assign random receptive field centers $\mathbf{m}_i$ to all neurons on layer L2. Note that these could all be the same value as demonstrated by one of the examples in the next section [complexity $O(N)$].
2. Initialize expectation counters k_i to zero and locking counters l_i to a large positive value [complexity $O(N)$].
3. Initialize the annealing value $a(t)$ and calculate the deformation values p_i, expectation values e_i, locking values h_i, and receptive field sizes r_i of all L2 neurons according to eqns (4), (8), (6), and (5), respectively [complexity $O(N)$].
4. Select a random point $\mathbf{x}$ on layer L1 falling within the receptive field of the layer L2 neuron with the largest receptive field size [complexity $O(1)$].
5. Perform the WTA operation to select a winning neuron according to eqn (1) [complexity $O(N)$].
6. Modify the receptive field of the winning neuron $\mathbf{m}_i^*$, according to eqn (2) [complexity $O(1)$].
7. Propagate the change in location of $\mathbf{m}_i^*$ through the connected portion of the graph using eqn (3) to update other receptive field centers $\mathbf{m}_j$ [complexity $O(M \log M)$].
8. Increment expectation counters k_i and calculate locking counter values l_i according to eqn (7) [complexity $O(N)$].
9. Reset the expectation counter of the winning neuron $k_i^* = 0$ [complexity $O(1)$].
10. Lower the annealing term $a(t)$ and update the deformation values p_i, expectation values e_i, locking values h_i, and receptive field sizes r_i of all affected neurons according to eqns (4), (8), (6), and (5), respectively [complexity $O(N)$].
11. Go to step 4.

The total computational complexity of the MEM model is thus $O[z(N+M\log M)]$, where z is the number of iterations allotted to complete the recognition process. In general, z is bounded below by N, since each neuron needs to be selected at least once. Empirically, z seems to scale as a constant multiple of N, with the constant ranging from 5 to 20. This multiplicative constant can be reduced, by selecting very salient (i.e., informative) features in the construction of the model graph. This becomes intuitively apparent by considering an example where a red colored object is to be found in a scene. If the only red object in the input scene is the target object, and the model graph contains a specific L2 neuron with the label "red", then the recognition process will be extremely fast, since the red labeled L2 neuron will quickly localize the object allowing the other neurons to bind to their corresponding points in the image.

If z can be assumed to be a constant multiple of N, as discussed above, and M be considered as a constant relative to N, the total computational complexity of the MEM model can be simplified to $O(N^2)$. This simplification specially holds when there are many objects in the memory ($M \ll N$). Parallel processing can be utilized to decrease the computation time of the model. The compute intensive portion of the model is the calculation of distances between $\mathbf{x}$ and all N neurons in L2 and the WTA operation (step 4 above), in the main loop of the algorithm. These basic operations are inherently parallel and are commonly used by other neural networks [e.g., Kohonen's SOM (Kohonen, 1987)]. The distance calculations can be performed entirely in parallel, on an architecture with N processors. The WTA calculation can then be performed in constant time with certain parallel architectures (Shu et al., 1991; Shams & Gaudiot, 1995). Thus, with an N processor machine, the computation can be performed in $O(N)$ time, if all L objects in memory are to be recognized. If only a small subset of the known objects are present in the scene (a reasonable assumption for large scale systems) a much smaller, and relatively constant, computational time of $O(M)$ might be possible with ($z \approx O(M)$). This scaling property seems to be in line with biological vision whose recognition rate does not slow down with learning of new objects.

3. SIMULATION RESULTS

The performance of the MEM algorithm was evaluated on the task of automatic identification of an object in a natural, cluttered scene. The input image captured a desktop scene containing a toy jeep along with a number of other partially occluded objects under poor lighting conditions. The model graph was generated from a simple line drawing of a jeep being viewed from a perspective similar to the input image. The line drawing representation of an object is at a fairly high level of abstraction. The basic features which can be obtained from such a representation are oriented edge information and their relative spatial distribution. Other target specific features, such as color and texture, can be incorporated into the model graph if the *a priori* information is available. In order to keep the model as general as possible, I chose to limit the model graph to consist of only oriented edge features. This representation allows for recognition of "jeep-like" objects ranging from sketch drawing of a jeep, to toy jeeps, and even to "real" jeeps.

Working at the line drawing level of abstraction does inherently introduce robustness to small shape variations. For example, the line drawing used to generate the model graph of Figure 5a, assumed that the jeep's windshield is composed of two distinct windows separated by some distance. It also further assumed certain proportions (length versus width of the hood for example) which were not in exact correspondence with the actual toy jeep in the input image. However, the network dynamics tolerated these local deformations through the available slack in the elastic connections between the feature nodes.

As mentioned in the introductory section, presently, model graphs are not learned; they are directly stored in model memory (layer L2). To carry out the present experiments, a heuristic procedure was used to construct a model graph from the line drawing based on the orientation and relative relationship of the line segments (see Figure 5a). The expected edge orientations are directly extracted from the line drawing without the use of any image processing (e.g., oriented edge filters). Two types of neurons were used in the construction of the model graph. The first, shown as small green asterisks, are sensitive to small oriented edges in the input, while the second, shown as large red asterisks, are sensitive to larger oriented edges. Neurons corresponding to the red asterisks have a large receptive field size and can therefore search the entire image. Since the red and green asterisks are connected by an elastic connection, when a candidate coarse-grain feature is selected for match, the related fine-grain feature detecting neurons are also "dragged" to a close spatial neighborhood of the coarse-grain neuron. If the correct match is in this local neighborhood, the fine-grain neurons will converge to the appropriately labeled points in the image. Otherwise, the fine-grain neurons will move in a different direction, causing an increase in the local deformation of the graph and therefore halting the coarse-grain neuron from following a false trajectory. Use of this approach allows for a simultaneous multi-level hierarchical search for large and small features by alternately selecting coarse-grain and fine-grain input L1 neurons for presentation to the L2 neurons.

Each L2 neuron at a specific layer of this hierarchy is connected to its small local neighborhood of neurons, with the coarser-grain cells having longer range connections. By employing many fine-grain features in the model graph, the robustness to scale, distortion, and perspective changes are increased since small, but smooth, deformations in many cells can lead to a large global deformation. This is similar in concept to linear approximation of a curved line with a large number of linear patches. In addition, by using multiple nodes on a single line, the system robustness to occlusion is enhanced considerably. For the jeep example of Figure 5, I found that by using only 30% of the L2 neurons (maximum of three nodes for any line segment), I can consistently produce correct recognition in the scene shown in Figures 5b–d. The more complex model graph is used in this paper to demonstrate the power of the algorithm.

A key feature of the MEM approach to object recognition is that the entire model graph, at all levels of the hierarchy, is used for the search. A more conventional approach might use coarse-grain features in the initial stage of processing to localize on a likely match area and then proceed with more fine-grain matches. This approach can be readily implemented in the MEM model by only activating L1 neurons representing coarse-grain features until a certain confidence level has been reached, as indicated by the locking values. The fine-grain-feature detecting neurons of L1 can then be activated to complete the match. I chose not to implement this scheme for a number of reasons. One, a sequential coarse to fine grain search requires additional *ad hoc* parameters and thresholds to know when to transition from one level to the next. Two, an additional control mechanism (Grossberg, 1976; Olshausen et al., 1993) would be required to exclude previously unsuccessful coarse-grain matches (those who failed to produce sufficient fine-grain correspondences) from being selected in the proceeding search iterations. The interleaved coarse-grain/fine-grain search used in the MEM allows for fast convergence onto the target since neurons of both levels aid each other in moving towards a good match and they would disagree when the move is in a false direction.

The input image was processed using oriented gabor filters with 12 different orientations (spaced at every 15°) at two different scales (Buhmann et al., 1989). This processing implements the feature-sensitive cells of layer L1 generating for every pixel in the input image a 12-dimensional vector encoding the presence or absence of specific small-scale oriented edges, and similarly a 12-dimensional vector for alternating pixels detecting larger-scale features. The gabor kernel of the small-scale feature detecting L1 neurons represents an approximately three pixel wide receptive field, and the large-scale feature detecting neurons represent an approximately six pixel wide receptive fields. The large-scale features are subsampled, by using one fourth the number of small-scale feature detecting cells. This subsampling is feasible since the large-scale features have larger receptive fields. In the example image shown in Figure 5b–d, slightly over 10,000 fine-grain and over 3000 coarse-grain feature cells were active (having outputs greater than a predefined threshold), representing a 123 to 13,000 node subgraph isomorphism problem, as discussed earlier in Section 1. Figure 6 shows a sample of the response values to 0° and 90° fine- and coarse-grain edge detecting gabor filters taken from the scene shown in Figure 5. Each pixel in these images corresponds to an active cell with the response magnitude proportional to the darkness of the pixel.

The use of gabor filters was partially motivated by their biological plausibility in addition to their simple control over size and orientation specificity. These filters can also be used to detect specific textures, but this feature was not utilized in these experiments. It should be made clear that the gabor filter is used only once at the initial stage of the algorithm to determine the pattern of activity in the L1 layer. During the iterative portion of the algorithm, the magnitude of the initially calculated responses is used to simulate the L1 layer neurons. In a real-time dynamic application, a fast parallel pre-processor can be used to continuously update the L1 layer neurons as objects move in the scene.

The MEM algorithm starts by assigning random values to each neuron's receptive field center $\mathbf{m}_i$. Due to this random initial state, there is a great amount of local deformation of the model graph, as defined by eqn (4), leading to large receptive fields for all neurons in the network. The MEM dynamics will quickly self-organize the neurons in L2 into a much less deformed version of the model graph (see Figure 5b). Random points in the feature plane (layer L1) are activated sequentially to attract appropriate points of the model graph (layer L2). The network will generally locate the object within a few hundred iterations, where one iteration is a single activation of a randomly selected L1 cell. The network will then start to fine-tune the connections to align all the fine-grain neurons with their corresponding points in the image plane. Cells in the L1 layer are selected in a manner which alternates between coarse-grain and fine-grain feature detectors. This procedure results in a simultaneous top-down and bottom-up search. This method was selected to improve system robustness in the presence of both high and low frequency noise. The state of the network after 4000 iterations is

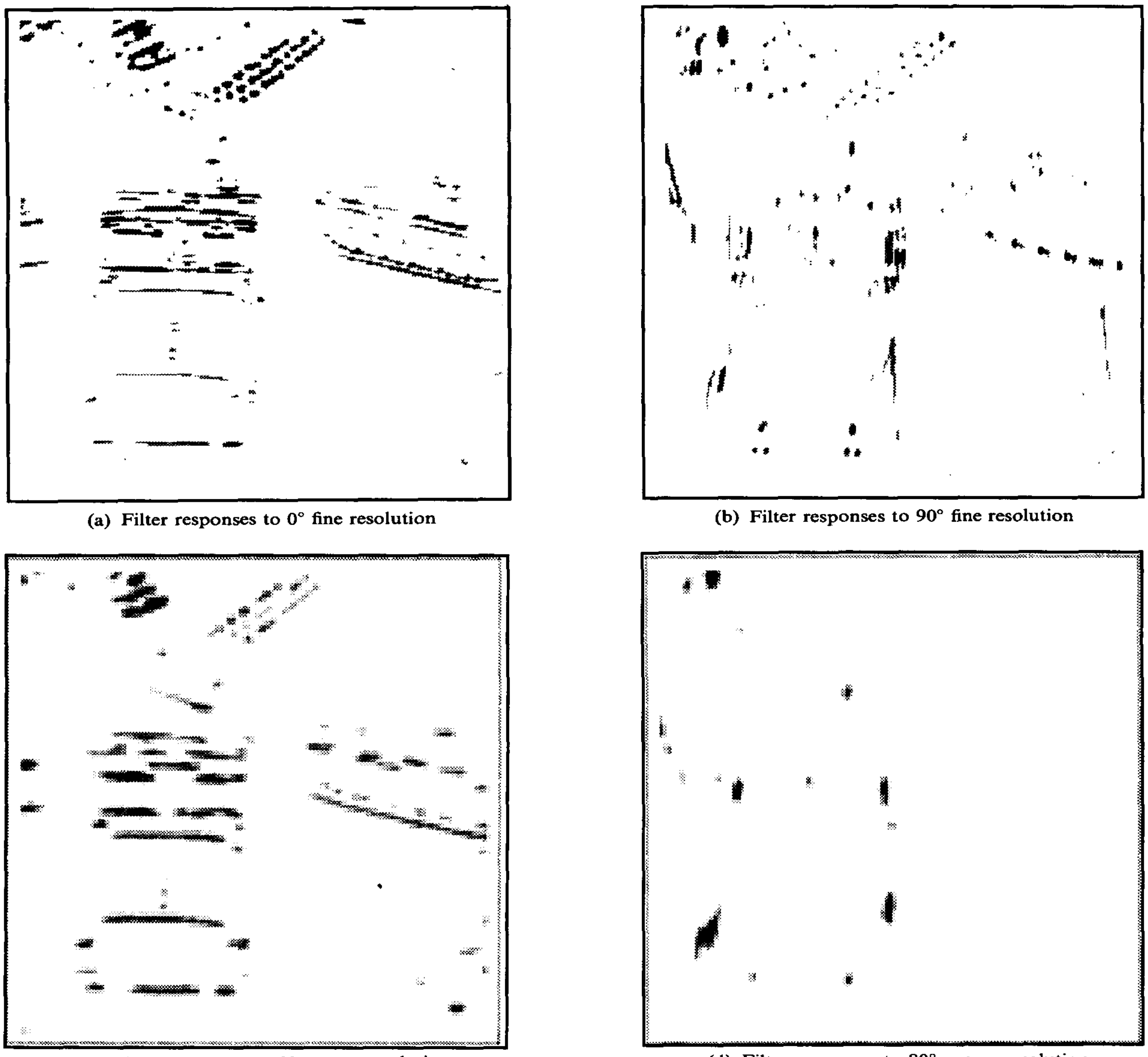

(a) Filter responses to 0° fine resolution
(b) Filter responses to 90° fine resolution
(c) Filter responses to 0° coarse resolution
(d) Filter responses to 90° coarse resolution

FIGURE 6. Example responses of L1 cells to the scene of Figure 5. (a) Output of cells sensitive to fine-grain horizontal edges. (b) Output of cells sensitive to fine-grain vertical edges. (c) Output of cells sensitive to coarse-grain horizontal edges. (d) Output of cells sensitive to coarse-grain vertical edges.

shown in Figure 5d. The time evolution of the system as represented by the sum of all the neuron locking values is shown in Figure 7. In these simulations, the resting value of the locking term h_i was set to 0.5 via the τ_h and δ_h parameters, described in the previous section. A neuron was considered locked if its locking value h_i reached below 0.1. An object was considered locked when the sum of all locking values, associated with the object, fell below $0.1M$, where M is the number of nodes in the object's model graph. It should be emphasized that in the MEM model, the locking effect is not binary (locked/unlocked), rather there is a continuous range of values between full lock and full unlock. Nevertheless, the gain parameter γ_l of the locking value's sigmoid function [eqn (6)], can be adjusted to approximate a discrete valued locking function.

It is a unique feature of the MEM algorithm to escape poor local matches in search of better solutions. A common technique used for escaping local minima is the simulated annealing algorithm (Kirkpatrick et al., 1983). With this approach, the global system temperature parameter is gradually reduced, based on a fixed cooling schedule. In the MEM algorithm, the receptive field size acts analogous to the temperature parameter of simulated annealing. In effect, the MEM approach implements a non-homogeneous temperature field where the neurons with "good" matches are at a much lower temperature, compared to those with "poor" matching. Due to the nature of the receptive field size calculation [eqn (5)], the temperature gradient is smooth and does not change abruptly between adjacent neurons, thus allowing for a

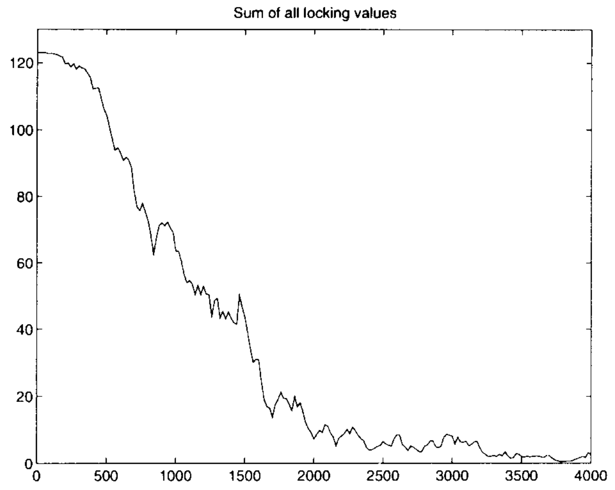


FIGURE 7. Sum of all the locking values in the graph as a function of time. After roughly 500 iterations, the network has localized the jeep at a coarse level. After 2000 iterations the network has locked to the input image and proceeds to further "fine-tune" the match.

smooth traversal of the energy space through a self-annealing process. Consequently, the locking values modulate the local temperature level, as defined in eqn (5). Low locking values hold the local temperature down (even in the presence of local deformations), while high locking values allow the local temperature to increase. In the MEM model, the dynamics of the locking parameter do not follow a monotonically decreasing "cooling schedule" and therefore the system temperature can indeed increase if a good solution is not found. This effect is illustrated in the example shown in Figure 8. Here the jeep model graph of Figure 5a is used to locate a jeep in a scene with no jeep-like object. The resting level of the locking values where set to 0.33 through parameters τ_h and δ_h. This lower resting level, from the previous experiment, was used to emphasise the non-monotonicity of the locking values. As shown in Figure 9, the network never locks to any part of the scene since the expected arrangement of features, defined by the model graph, is not found in the input. The dynamics of this example are quite interesting. The network makes a hypothesis that the vertical edges generated by the propeller are to be associated with one side of the jeep (Figure 8a). However, since other neurons in L2 are not successfully bound to neurons in L1, the network will quickly abandon that hypothesis and try a different one, such as associating the airplane wing with the horizontal edge of the jeep's hood line (Figure 8b). This process continues indefinitely, since there are no jeeps in the image.

It should be noted, however, that there is a tradeoff between robustness and "false" detection. As stated earlier, the interconnection links between features are elastic and their elasticity can be increased to allow for larger distortions in the input image, and thus improve robustness to changes in perspective and size. However, this same feature will allow the network to satisfactorily lock onto a "jeep-like" object that is not a jeep, a pick-up truck for example. In order to increase the certainty in the match, stiff interconnection links can be used. If these links are set to hard non-elastic connections, the network will implement a basic edge matching operation, similar to a direct correlation of the edges in the model with those in the input, with the exception that the search is performed stochastically by the MEM model instead of being scanned.

As stated in Section 1, it is my intention to incorporate a hierarchical learning mechanism in the MEM model. With such a learning procedure, a model graph similar to Figure 5a will be learned through repeated observation of jeep-like objects, such as jeeps, pickup trucks, etc. The elasticity of interconencting links of this graph will be relatively large. Further training can generate specific nodes to differentiate between jeeps and pickup trucks. The interconnection links between jeep specific, as well as pickup truck specific, graph nodes will be stiffer, since more detailed *a priori* knowledge is available. With an appropriate learning algorithm, the link elasticities can be automatically learned by the network depending on experience.

Another unique feature of the MEM algorithm is its ability to simultaneously search for multiple objects. This feature has a number of important benefits over sequential matching of multiple objects. First, the computation associated with detection and recognition of multiple objects will be inherently parallel, greatly simplifying the task of implementation on parallel processing hardware. More importantly, simultaneous search speeds up the recognition process by dividing the search space into regions of most likely match between each object and the input scene. In other words, it should be easier to recognize a familiar object in a scene with other familiar objects, than a scene cluttered with unfamiliar objects. In order to demonstrate the simultaneous multiple object recognition characteristic of the MEM model, a new scene with multiple objects, including a toy jeep and a toy tank, was utilized. The jeep in the image was partially occluded by another object (see Figure 10). A new model graph for a tank was generated using a CAD line drawing, Figure 10a, and added to the model graph memory. These two disjoint graphs correspond to neurons of layer L2 in Figure 2. Multiple frames of the time evolution of the network are shown in Figure 10b–d. The initial configuration of all neuron receptive field centers $\mathbf{m}_i$ were set equal to each other with all $\mathbf{m}_i$s pointing to the center of the image. It can be seen in Figure 10b

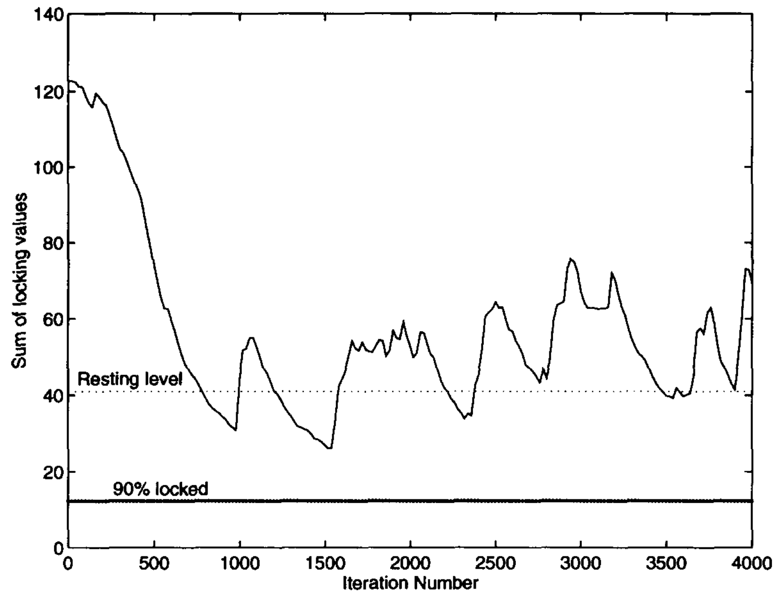


FIGURE 9. Sum of all the locking values as a function of time. The network repeatedly loses its attempt at locking to the input because of insufficient correspondence between the expected arrangement of features, given by the model graph, and the input scene activation of L1 cells.

that the L2 neurons quickly self-organize into a jeep-like and a tank-like arrangement, according to the stored δ_{ij} values, and begin converging toward the appropriate points in the image (see Figure 10c). Note that the expected tank model is shorter in the x-direction and longer in the y-direction from the input image and the network dynamics appropriately deform the model graph to fit the data. However, the corresponding L2 layer neurons of the tank model were slower in converging due to this larger deformation. After 1000 iterations, the network has localized the objects and proceeds with "fine-tuning" of the match. By 4000 iterations, almost all L2 neurons have fully locked and the L2 neuron receptive field centers are aligned properly with neurons in the L1 layer. The time evolution of the network, based on the sum of all locking values from both model graphs, is shown in Figure 11.

The simulation results reported in this section were implemented on a Sun Sparc-10 workstation. The simulation code was written and executed using the MatLab development package. The implementation code had many inefficiencies and incorporated repeated time consuming graphic updates which were tolerated in order to quickly establish a proof of concept design. Under these conditions, the wall clock execution time of the algorithm was approximately 5 ms per iteration per number of L2 neurons. Therefore, approximately 2 min were required to reach the 200 iterations point of the 123 node jeep graph shown in Figure 5b, and about 40 min to reach the 4000 iterations point shown in Figure 5d. A speedup factor of 10 seems quite plausible by simply optimizing and compiling the code on the same serial hardware.

4. DISCUSSION

The MEM model is a new method for rapid optimization in a large state space. In this paper I have demonstrated its use in visual object recognition through optimization of the spatial alignment between expected and input edge features. In this application, the dynamics of the MEM model are used to converge to solutions with the labeled nodes of the model graph being spatially aligned with similarly labeled features extracted from the input image, while tolerating a certain amount of local deformation of the model graph. Although the MEM model currently only deals with 2-D shapes, extensions to 3-D object recognition can be conceived by utilizing 3-D model graphs. In such an implementation, properly weighted lateral connections between L2 neurons, for communication of the locking values, can aid in recognition and determination of the perspective view of the object. The basic mechanism will remain similar except it will involve simultaneous searching in feature space as well as orientation and perspective rotation spaces. Since the basic optimization principle of the MEM model is regularization in different modalities, only a few perspective graphs will be needed to define a 3-D object, thus alleviating a need for a large number of redundent neurons to define a single model graph.

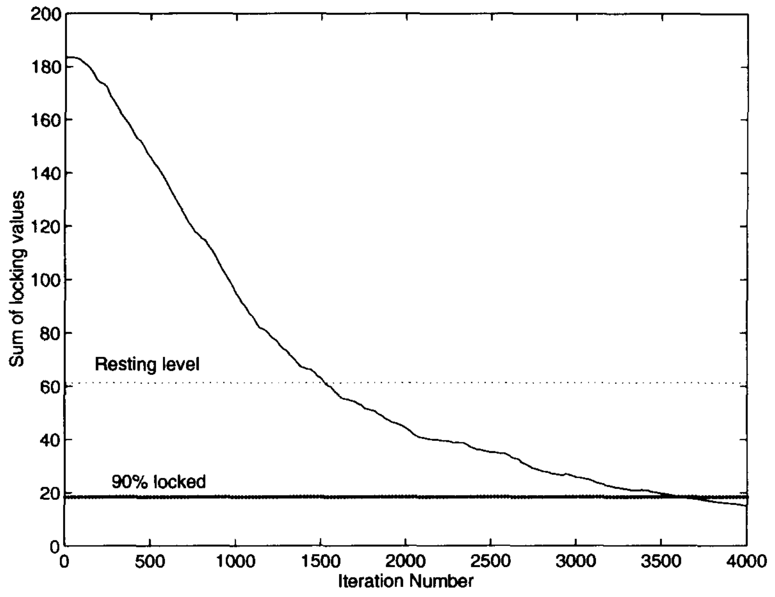


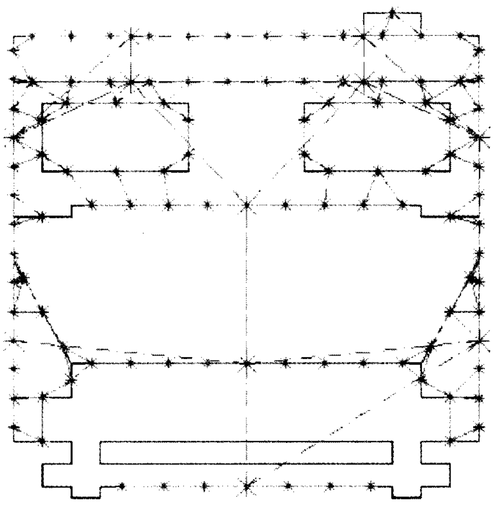
FIGURE 11. Sum of all the locking values, from both the jeep and tank graphs, as a function of time. The network converges smoothly to a locked state by finding both objects in the scene.

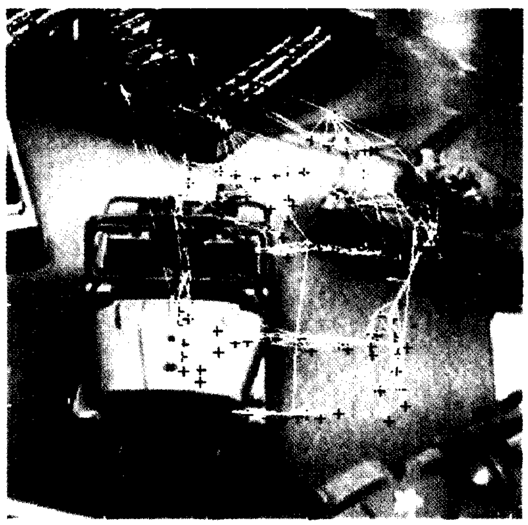
The MEM model can allow for moderate deformation of this graph in perspective space to interpolate between the stored perspective views (Poggio & Edelmann, 1990). Similarly, the MEM model can be extended to allow for scale invariance by concurrently searching scale space along with feature space. Recent experiments in combining simultaneous searches in feature and scale space have been very encouraging, allowing for translation, size, and distortion invariant object recognition, without significant addition in convergence time and computational requirements. Further work in this area is anticipated to include the addition of orientation and 3-D perspective invariances. It is also possible to extend these ideas further by searching abstract spaces [e.g., shape space (Cutzu & Edelman, 1995)], since there can be a smooth measurable space spanning various objects. This scheme will allow the system to recognize an unfamiliar object *and* know that it is "something between a pickup truck and a car", for example.

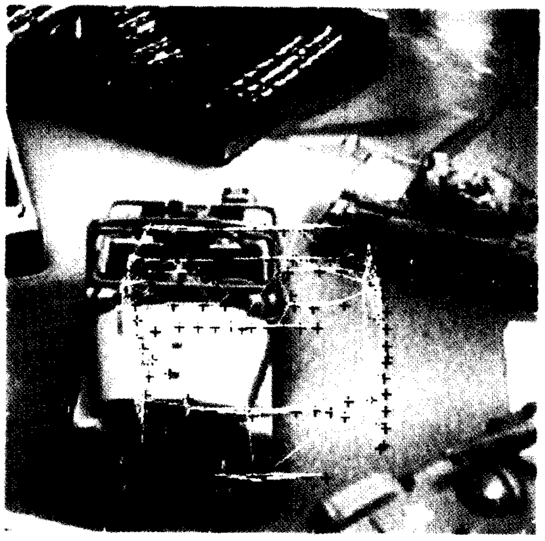
One of the major contributions of the MEM algorithm is to incorporate a dynamic focus-of-attention mechanism. This mechanism is implemented through appropriate dynamics of neuron receptive field location and size values which allow the network to quickly zero-in on good matches and escape poor ones. The performance of the MEM model is further enhanced by having feature node labels along with receptive field locations implementing a top-down expectation, which is recurrently coupled with the feature activation information being propagated from the input bottom-up. This can be viewed as generating hypotheses on the top-level and having low-level activations verify or reject the hypotheses (Arbib, 1989). In current experiments, explicit feedback connections are being tested to increase the probability of activation of L1 neurons according to the expectation level of L2 neurons (specified by their receptive field location and size). This type of processing is reminiscent of bi-level recurrent neural networks like adaptive resonance theory (ART) (Grossberg, 1976) and bidirectional associative memory (BAM) (Kosko, 1988).

The reliance of the MEM model on local spatial relationships of features makes it especially suitable for tracking moving objects by requiring only minimal computation to modify the receptive field locations $\mathbf{m}_i$ from frame to frame. This feature has already been verified on a target tracking problem (Shams, 1995). Integration of multiple sensory information, such as infrared and visible sensors, in advance automatic target recognition (ATR) applications, can also be easily implemented in the MEM model by having appropriate node labels used in the construction of the model graphs. Their integration can be trivially implemented through the δ_{ij} matrix. The current state of the MEM model is applicable to a wide range of ATR missions where the expected angle of approach, size, and orientation of the target is known to a certain degree. It is important for future advance mission management capabilities to allow for real-time changes in the approach trajectory where 3-D perspective and size invariance will be more demanding.

There are a number of additional benefits inherent in the graph representation presented here which remain to be developed. Hierarchical organization of


(a) Model graph generated from line drawing


(b) State of the network after 60 iterations


(c) State of the network after 200 iterations

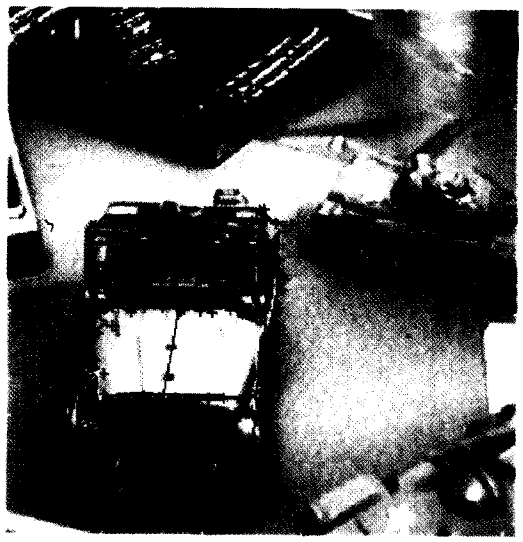

(d) State of the network after 4000 iterations

FIGURE 5. The MEM algorithm used to find a toy jeep in a cluttered scene. (a) The line drawing (dark blue), made by a CAD tool, was processed using a heuristic algorithm to generate fine-grain (small green asterisks) and coarse-grain (large red asterisks) oriented edge locations. The fine-grain nodes are interconnected to small local neighborhood (light blue lines) and the coarse-grain features are interconnected to a larger neighborhood (red broken lines) of nodes. (b) Spatial arrangement of L2 neuron receptive field centers m_i overlaid on the input image after 60 iterations. (c) After 200 iterations the receptive field centers are distributed similar to the model graph seen in (a). (d) After 4000 iterations the majority of neurons have locking values below 0.1 indicated by the red inter-node links.

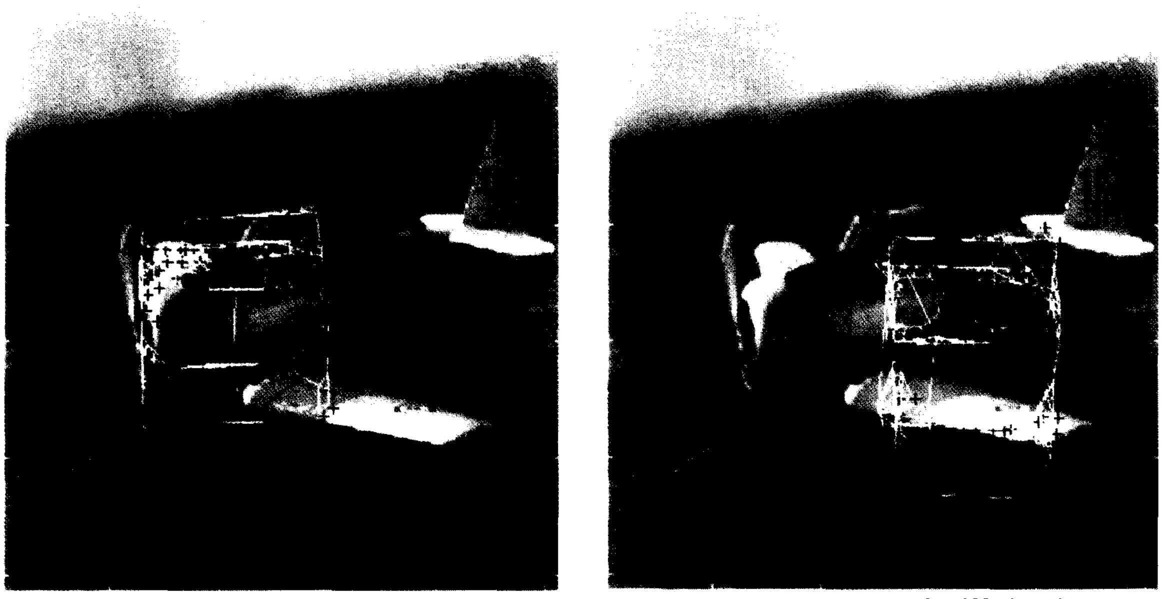
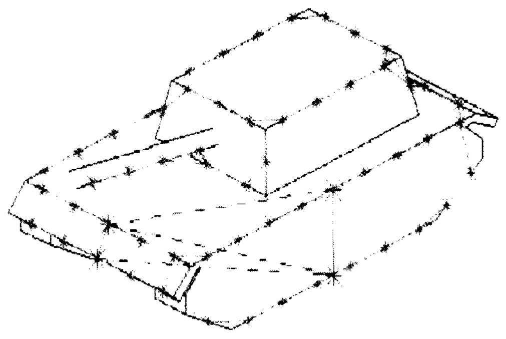
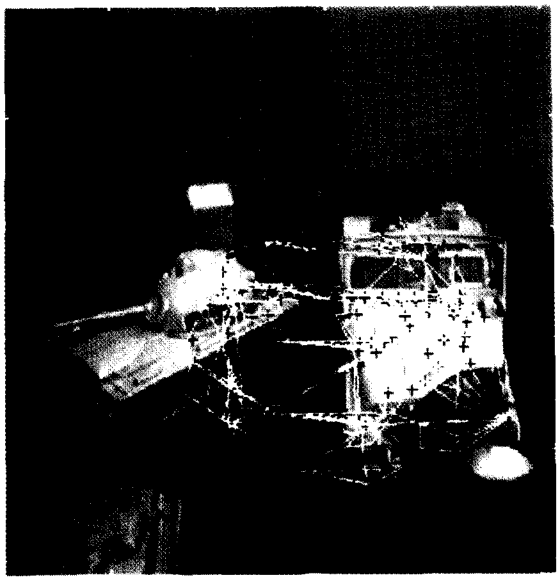
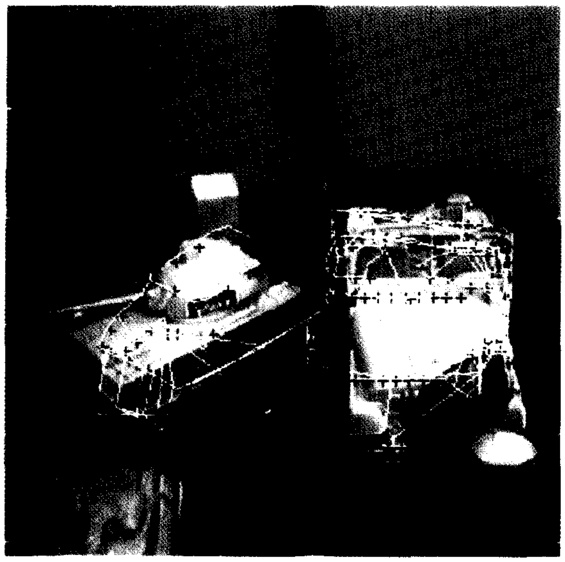

(a) Network state after 600 iterations (b) Network state after 3920 iterations

FIGURE 8. Finding a jeep in an image without a jeep will never converge. Many hypotheses are tried, then abandoned having received insufficient supporting evidence from the L1 neurons.


(a) Model graph generated from line drawing


(b) State of the network after 40 iterations


(c) State of the network after 200 iterations

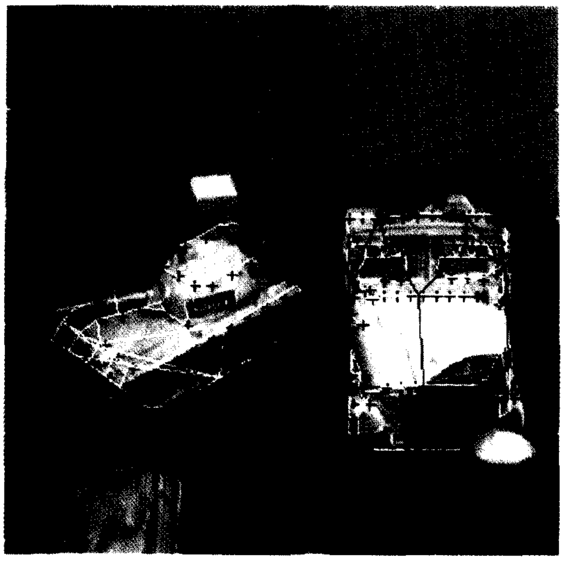

(d) State of the network after 1000 iterations

FIGURE 10. The MEM algorithm used to simultaneously find a toy jeep and a toy tank in a cluttered image. The jeep is partially occluded by a tape dispenser. (a) The CAD drawing of the tank (dark-blue lines) and associated feature nodes with their expected interrelationships; green asterisks indicate fine-grain features and red asterisks indicate coarse-grain features. (b) Spatial arrangement of L2 neuron receptive field centers m_i overlaid on the input image after 40 iterations. (c) After 200 iterations the receptive field centers from the jeep and tank model graphs are moving towards their respective objects in the image. (d) After 1000 iterations both objects have been found and most of the neurons associated with the jeep are locked to the image.

data can be easily implemented through the use of multi-layer graphs. High-level concepts or schema's (Arbib, 1989) can be activated through the propagation of locking parameter values. This procedure will allow construction of graphs where a "jeep" neuron and a "car" neuron will both be active. The activation of these neurons will arise from the proper activation of the hood, tire, door, and windshield neurons having been activated in a specific spatial arrangement which have themselves been activated based on specific arrangement of finergrain features (from layer L1). In addition, more biologically plausible extensions to the MEM model are currently under investigation.

REFERENCES

Arbib, M. A. (1989). *The metaphorical brain 2. Neural networks and beyond.* New York: John Wiley.

Buhmann, J., Lange, J. & von der Malsburg, C. (1989). Distortion invariant object recognition by matching hierarchically labeled graphs. *Proceedings of the Inter. Joint Conf. on Neural Networks.* Washington D.C. (Vol. **1**, pp. 155–159).

Cutzu, F., & Edelman, S. (1995). Explorations of shape space. Technical Report CS-TR-95-01, The Weizmann Institute of Science, January 27.

Durbin, R. & Willshaw, D. (1987). An analogue approach to the traveling salesman problem using an elastic net method. *Nature* **326,** 689–691.

Fukushima, K. (1987). Neural network model for selective attention in visual pattern recognition and associative recall. *Applied Optics,* **26**(23), 4985–4992.

Grossberg, S. (1976). Adaptive pattern classification and universal recoding. II. Feedback, expectation, olfaction, and illusions. *Biological Cybernetics,* **23,** 187–202.

Hopfield, J. J. (1982). Neural networks and physical systems with emergent collective computational abilities. *Proceedings of the National Academy of Science USA,* **79,** 2554–2558.

Hopfield, J. J. & Tank, D. W. (1985). "Neural" computation of decisions in optimization problems. *Biological Cybernetics,* **52,** 141–152.

Kirkpatrick, S., Gelatt, C. D. Jr & Vecchi, M. P. (1983). Optimization by simulated annealing. *Science,* **220,** 671–680.

Kohonen, T. (1987). *Self-organization and associative memory* (2nd edn.). Springer Series in Information Sciences. Berlin: Springer-Verlag.

Konen, W., Maurer, T., & von der Malsburg, C. (1994). A fast dynamic link matching algorithm for invariant pattern recognition. *Neural Networks,* **7**(6/7), 1019–1030.

Kosko, B. (1988). Bidirectional associative memories. *IEEE Transactions on Systems, Man, and Cybernetics,* **18,** 49–60.

Mjolsness, E. (1995). Personal communication.

Nasrabadi, N. M. & Li, W. (1991). Object recognition by a Hopfield neural network. *IEEE Transactions on Systems, Man and Cybernetics.* **21**(6), 1523–1535.

Olshausen, B. A., Anderson, C. H. & Van Essen, D. C. (1993). A neurobiological model of visual attention and invariant pattern recognition based on dynamic routing of information. *The Journal of Neuroscience,* **13**(11), 4700–4719.

Poggio, T., & Edelman, S. (1990). A network that learns to recognize three-dimensional objects. *Nature,* **343,** 263–266.

Reiser, K. (1991). *Learning persistent structure.* Ph.D. Thesis, University of Southern California.

Shams, S. (1995). A new self-organizing model for angel-only target deghosting and tracking. *Proceedings of the World Congress on Neural Networks,* Washington, DC, Vol. 2, pp. 646–651.

Shams, S., & Gaudiot, J.-L. (1995). Implementing regularly structured neural networks on the DREAM machine. *IEEE Transactions on Neural Networks,* **6**(2).

Shu, D. B., Nash, J. G., Eshaghian, M. M., & Kim, K. (1991). Implementation and application of a gated-connection network in image understanding. In *Reconfigurable massively parallel computers,* H. Li and Q. F. Stout (Eds.). Englewood Cliffs, NJ: Prentice Hall.

Simic, P. D. (1990). Statistical mechanics as the underlying theory of 'Elastic' and 'Neural' optimisations. *Networks,* **1,** 89–103.

Simic, P. D. (1991). Constrained nets for graph matching and other quadratic assignment problems. *Neural Computation,* **3,** 268–281.

von der Malsburg, C. (1988). "Pattern Recognition by Labeled Graph Matching." *Neural Networks.* **1:** 141–148.

von der Malsburg, C., & Bienenstock, E. (1987). A neural network for the retrieval of superimposed connection patterns. *Europhysics Letters,* **3**(11), 1243–1249.

Yuille, A. L., Hallinan, P. W. & Cohen, D. S. (1992). Feature extraction from faces using deformable templates. *International Journal of Computer Vision,* **8**(2), 99–111.

APPENDIX

The MEM model utilizes a number of parameters to control its internal dynamics. Many of these parameters have physical meaning associated with them; others can be determined by specific application requirements. In this appendix a detailed description of these parameters and their effect on the network dynamics are presented.

1. *Parameters controlling the receptive field size.* The receptive field size of a neuron i is represented as r_i, is defined by eqn (5). All the terms in this equation, except for $a(t)$ and ε_i, are dynamically calculated by other variables in the system. Therefore, the user only has to specify the minimum receptive field size ε_i and the scaling value $a(t)$. As discussed in the paper ε_i is set equal to the receptive field size of the feature detecting L1 neurons with similarly labeled features. The scaling parameter $a(t)$ is set such that at $t=0$, the receptive field size of all neurons can cover the entire image. The magnitude of this variable is directly related to deformation measure p_i. By redefining p_i to be normalized as

$$p_i = \frac{\sum_{j\in L_i} \|(\mathbf{m}_i - \mathbf{m}_j) - \boldsymbol{\delta}_{ij}\|}{\sum_{j\in L_i} \boldsymbol{\delta}_{ij}} \tag{A.1}$$

we can ensure that a constant value for $a(t)$ can be used for any arbitrary shaped model graph. In fact, I have used a constant $a(t)=5$ value for many simulations with good recognition performance.

2. *Parameters controlling the receptive field center updating.* The update rate parameter α, as discussed in the paper, is directly calculated from the similarity function $R(\mathbf{x}, \mathbf{y})$ and the maximum update rate α_o. The smilarity function was implemented as a normalized inner product of feature vectors $\mathbf{x}$ and $\mathbf{y}$ (indicating the cosine of the angle between the two feature vectors). The maximum update rate α_o can be set to any value less than 1. I found that values in the range 0.7–0.8 seem to work best for most of the simulations.

The updating rate of neighboring neurons α_i are dynamically calculated based on a predefined elasticity value for the links and the amount of deformation in the graph. What is required is to have high tension when the graph is much deformed and little tension when its slightly deformed. This can be implemented as:

$$\alpha_i' = \frac{\alpha p_i}{p_i + \eta_i} \tag{A.2}$$

where η_i is used to control the elasticity of neuron i. A different approach was used for the simulations described in the paper by defining the update rate as:

$$\alpha_i' = \alpha S[(1 - h_i)(p_i - \eta_i)] \tag{A.3}$$

where S is the sigmoid nonlinearity having the range (0,1). This formulation changes the elasticity of the links as a function of the locking value. If the neuron is locked, small movements $< \eta_i$ do not move the receptive field center of neuron i. On the other hand if the neuron is not locked at all ($h_i = 1$), then $\alpha_i' = 0.5$ regardless of the local deformation p_i. The only parameter which needs to be manually adjusted in both cases is η_i. In the simulations this value was set equal to $2\varepsilon_i$.

3. Parameters controlling the locking values. The control of the locking values is the most important, and delicate part of the MEM dynamics. However, in many applications, where the target of interest can be easily discriminated from the background (having a novel feature) this variable can be completely removed (set all $h_i = 1$) with little effect on the recognition rate. The significance of this value is that it enables the system to have a measure of "confidence" in the recognition. There are five free parameters that must be specified by the user in eqn (6) and (7), namely γ_1, γ_2, τ_h, τ^*, and δ_h. Other parameters, such as α, h_i', l_i', and (dm_i^*/dt) are dynamically calculated. There are a number of constraints and interdependencies between the free parameters that can help simplify the parameter selection process. The first step in this process is to understand the effect of these parameters on the locking value h_i. As discussed in the paper, the auxillary variable h_i' is used to limit the growth and decay of h_i and is itself a direct function of the internal integrator value l_i. In order to calculate the steady state value of h_i, we can set $(dl_i/dt) = 0$ in eqn (7a). Then we can define $\rho_h = (\tau_h/\delta_h)$ and $\sigma = (\gamma_1/\gamma_2)$, and with some algebra determine the steady state value of h_i based on ρ_h and σ as

$$\hat{h}_i = S\left[\sigma S^{-1}\left(\frac{1}{\rho_h + 1}\right)\right], \tag{A.4}$$

S^{-1} is the inverse if the sigmoid function defined as

$$S^{\prime}(y) = \ln\left(\frac{y}{1 - y}\right).$$

A suitable value for $\hat{h}_i$ will be in the range 0.3–0.5.

4. Parameters controlling the expectation values. The only user definable parameter for the expectation variable [(eqn. (8)] is the shape of the function Q. The basic shape of the function is given in Figure 4 of the paper. A specific formulation of this function, which was used in my simulations, is

$$Q(k_i) = \frac{10\varepsilon_i}{1 + [(k_i - 3N)/N]^4}. \tag{A.5}$$