Scene Analysis by Integrating Primitive Segmentation and Associative Memory

DeLiang Wang, *Senior Member, IEEE,* and Xiuwen Liu, *Associate Member, IEEE*

***Abstract*—Scene analysis is a major aspect of perception and continues to challenge machine perception. This paper addresses the scene-analysis problem by integrating a primitive segmentation stage with a model of associative memory. Our model is a multistage system that consists of an initial primitive segmentation stage, a multimodule associative memory, and a short-term memory (STM) layer. Primitive segmentation is performed by locally excitatory globally inhibitory oscillator network (LEGION), which segments the input scene into multiple parts that correspond to groups of synchronous oscillations. Each segment triggers memory recall and multiple recalled patterns then interact with one another in the STM layer. The STM layer projects to the LEGION network, giving rise to memory-based grouping and segmentation. The system achieves scene analysis entirely in phase space, which provides a unifying mechanism for both bottom-up analysis and top-down analysis. The model is evaluated with a systematic set of three-dimensional (3-D) line drawing objects, which are arranged in an arbitrary fashion to compose input scenes that allow object occlusion. Memory-based organization is responsible for a significant improvement in performance. A number of issues are discussed, including input-anchored alignment, top-down organization, and the role of STM in producing context sensitivity of memory recall.**

***Index Terms*—Associative memory, grouping, integration, locally excitatory globally inhibitory oscillator network (LEGION), scene analysis, segmentation, short-term memory (STM).**

I. Introduction

Much of neural network research centers on pattern recognition [27], [38], where the fundamental premise is that patterns are presented to the system one at a time. In reality, however, such an interface can rarely be taken for granted; when one looks around, it is clear that the scene (or image) around us is almost invariably composed of multiple patterns or objects. Many years of study in computer vision [31] prove that segmenting a scene into meaningful entities for subsequent processing (e.g., pattern recognition) is just as hard a problem as invariant pattern recognition. In addition to the segmentation problem, neural network research also faces what we call the *alignment problem*, which refers to the problem caused by not aligning an input pattern with a stored pattern. Take a Hopfield net as an illustration, which stores five-digit patterns in the memory, as shown in Fig. 1(a). Each one of the digits is correctly recalled if an input pattern is the same as one of the stored patterns *and* is perfectly aligned with the stored pattern. This is shown in Fig. 1(b) for pattern "4." However, as shown in Fig. 1(c), the same input that is shifted two pixels to the right leads to a wrong recall.

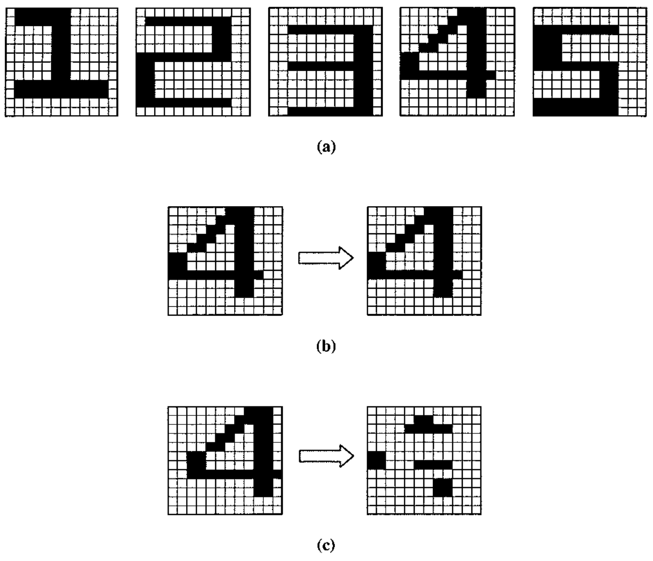


Fig. 1. Example of the alignment problem for the Hopfield net. (a) Five binary patterns, corresponding to digits "1" through "5," are stored in a 10×10 Hopfield net. Standard implementation is followed. (b) An input pattern "4," which is perfectly assigned with the stored template of "4," leads to a perfect recall of the stored "4." (c) The same input pattern shifted two pixels to the right does not yield any useful recall.

Efforts have been made to address the segmentation problem; in particular, many models of oscillatory associative memory have been proposed to achieve the recall of multiple patterns [8], [18], [30], [42] (for a nonoscillatory example see [19]). Compared to traditional associative memory based on attractor dynamics, oscillator networks are based on limit-cycle dynamics and introduce an additional degree of freedom—time. In such networks, one pattern is represented by a synchronous group of oscillators and different patterns are represented by oscillator groups that desynchronize from one another [36], [37]. Thus, within a period of oscillation, multiple patterns alternate and take turns to be activated. Although oscillatory dynamics offers an elegant representation for the result of segmentation, the performance of oscillatory associative memory is limited to very simple stimuli. A major reason is the alignment problem; multiple patterns need to be presented so that each is aligned with the corresponding stored template. This requirement leads to double predicaments. First, one should not control the spatial

Manuscript received November 3, 2000; revised July 3, 2001 and December 3, 2001. This work was supported in part by an ONR YIP Award (to D. Wang), NSF Grant (IIS-0081058), and AFOSR Grant (F49620-01-1-0027). This paper was recommended by Associate Editor D. Cook.

D. L. Wang is with the Department of Computer and Information Science and Center for Cognitive Science, The Ohio State University, Columbus, OH 43210-1277 USA (e-mail: dwang@cis.ohio-state.edu; liux@cis.ohio-state.edu).

X. Liu is with the Department of Computer Science, The Florida State University, Tallahassee, FL 32306-4530 USA (e-mail: liux@cs.fsu.edu).

Publisher Item Identifier S 1083-4419(02)01266-9.

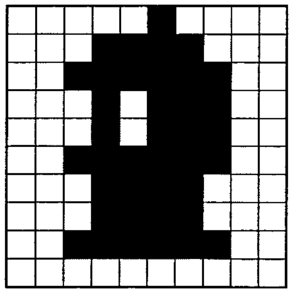


Fig. 2. Superimposed array of the four binary patterns corresponding to numerals “0” through “3.”

arrangement among patterns; indeed, there are unlimited ways of arranging a fixed number of patterns. Second, such memory models can recognize a superposition of patterns that is impossible for our visual system. This is illustrated in Fig. 2, where a superposition of four binary pixel arrays corresponding to the numerals “0” to “3” is shown. We are hard pressed to recognize the scene, whereas such superpositions are often used to test segmentation of associative memory. This “super-performance” results from the failure of taking into account that objects in the real world are opague in general and in three-dimensional (3-D) space, occlusion among the objects makes it impossible to superpose objects in the way shown in Fig. 2.

A more promising line of research separates segmentation from recognition and tackles on segmentation directly. Grossberg and Wyse [6] proposed a model for segmentation based on contour detection and subsequent region filling-in. Wang and Terman [44] proposed the use of locally excitatory globally inhibitory oscillator network (LEGION) for image segmentation, which is based on synchrony within each block of neural oscillators representing pixels that likely belong to the same object and desynchrony between the blocks. Also, Yen and Finkel [46] used oscillator networks for extracting salient contours. In these networks, segmentation is based on primitive Gestalt rules such as proximity and pixel/orientation similarity and there is no involvement of recognition. Image segmentation has been, for a long time, a major topic in computer vision and image processing and a variety of computer algorithms have been proposed [16], [31]. Despite a lot of effort, the segmentation problem eludes a general solution [41]. It has become clear that there are limitations to bottom-up segmentation based only on primitive grouping rules such as proximity and local similarity. It is well documented that prior knowledge and recognition influence the way human subjects parse a scene [2], [23] and a complete solution to segmentation would require an integration of bottom-up, primitive segmentation and top-down, knowledge driven segmentation. Little investigation, within neural networks or beyond, has been targeted to such an integration (see also [41]).

Motivated by the preceding observations, this paper takes an integrated approach to addressing the scene-analysis problem; namely, by integrating primitive segmentation and memory-based segmentation. The resulting model is a multistage system encompassing initial primitive segmentation, multimodule associative memory and short-term memory (STM) that allows for interaction between multiple recalled patterns. Underlying the multistage model is a phase space representation, whereby different patterns attain different phases or occur at different times. Our model has been evaluated using a systematic database of 3-D line drawing objects that are arranged in an arbitrary fashion and so can occlude one another. With these test scenes, a number of conceptual issues are exposed and addressed, including alignment, pattern completion and interaction, and further grouping and segmentation based on memory recall. Overall, our system successfully resolves a number of key issues in scene analysis and significantly enhances the scene-analysis ability of neural networks.

The rest of the paper is organized as follows. Section II introduces the architecture and gives an overview of the system, the detailed description of which is given in Section III. Section IV provides simulation results and further discussions are given in Section V.

II. Network Architecture and Motivation

Because primitive segmentation can organize a scene into structures, or objects, based on very general principles such as proximity and local similarity, for scene analysis it is important to perform primitive segmentation first and have recognition operate on structured objects rather than pixels directly. This computational strategy is sensible for the following two reasons. First, natural scenes are structured and our perceptual system normally responds to structured objects, not pixels. Second, structured objects can be much easier to analyze computationally than unstructured ones. In contrast, stored patterns in most associative memory models are random bit strings rather than structured objects.

Fig. 3 shows the network architecture for our model and its flow of computation is as follows. An input image is first processed by a segmentation layer and its results are fed to the associative memory layer which consists of multiple memory modules. The memory layer is fed to an STM layer, where interaction takes place between multiple recalled patterns from the memory. The STM layer also feeds back to the segmentation layer for refining the results of primitive segmentation, i.e., to perform further grouping and segmentation. It is the STM layer that embodies the result of scene analysis. We note that terms like STM[1] have different meanings in different communities and what they mean here will become clear in Section III.

Fig. 4 shows a simple example of a scene that our system is proposed to deal with. There are two objects: the pyramid-like object is at the front and it partially occludes the cube-like object. The task of primitive segmentation, ideally, is to group black pixels into connected contours and segment the contours that belong to different objects. It is well-known that T-junctions play a critical role in separating occluding contours from occluded contours [22]. Thus, we apply a T-junction detection step to the input image and its results are utilized in primitive segmentation.

[1]More specifically, the use of STM in this paper is not intended to be explanatory from the cognitive perspective. Rather it corresponds to a distinct stage of computation that processes and temporarily holds recall results from the memory layer.

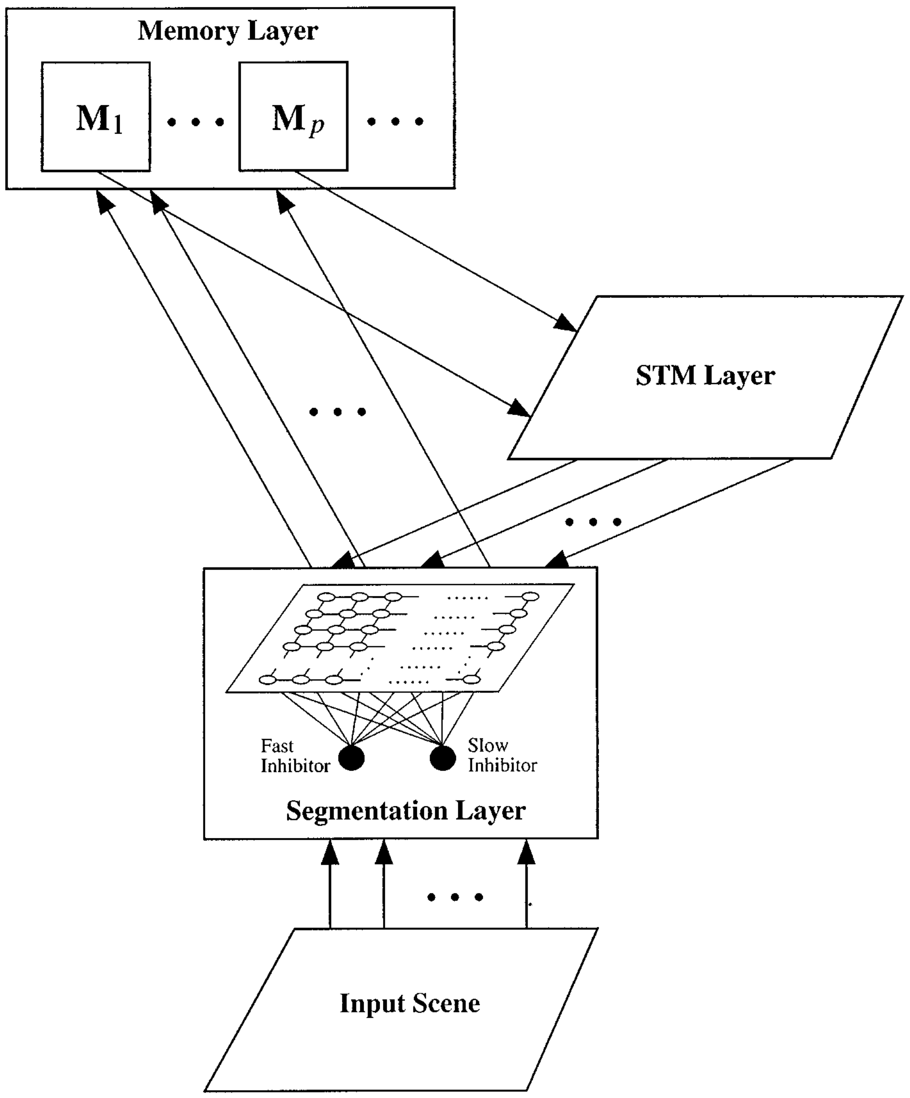


Fig. 3. Architecture of the scene-analysis system, which is composed of the segmentation layer, the memory layer, and the STM layer. Arrowheads indicate directions of network connection.

We employ a LEGION network for primitive segmentation because of its ability to segment a variety of real images (e.g., [28]) and the oscillatory correlation representation it realizes. In oscillatory correlation, a segment (structure) is represented by a synchronized oscillator group and different segments are represented by desynchronized groups [33], [37]. Because relaxation oscillators are the building blocks for LEGION, only one segment is active at any time from the segmentation layer and this greatly facilitates a solution to the alignment problem.

An active segment, which is a contour pattern in this study, inputs simultaneously to the multiple modules in the memory layer, as shown in Fig. 3. Each memory pattern is stored in a separate module through a specific coupling between units. Separate modules are used for different patterns in order to avoid difficulties with overlapping storage as well as with arbitrary spatial arrangements among overlapping patterns that are inevitably introduced when multiple patterns are first stored. There are two potential losses with such a design: lack of useful interaction between memory patterns and higher storage costs in terms of units and connections. As explained below, the first loss is compensated for with the inclusion of the STM layer. The second loss is largely eliminated by using a locally coupled network for each module instead of commonly used all-to-all connections. Locally connected networks can be used for storing patterns because objects used in our study and natural objects in general, are all connected patterns. When a pattern is stored in a module, its center of gravity is calculated—easily done in a neural network representation. Meanwhile, when an active pattern from the segmentation layer feeds to the memory layer, the center of gravity of the active pattern is also calculated and it is used to align with the center of gravity of a stored pattern. This is how the alignment problem is addressed in our model. However, an active pattern of the segmentation layer may not always correspond to a whole object; e.g., the two parts of the cube-like object in Fig. 4 would be segmented apart and each be treated as a separate segment by the segmentation layer. Thus, a shifting operation is also carried out when an active pattern of the segmentation

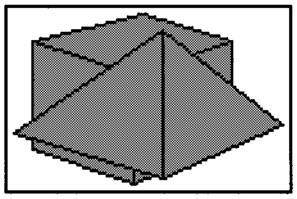


Fig. 4. Example scene that consists of two patterns: a pyramid and a cube (the latter occluding the former).

layer feeds to the memory layer. Such shifting starts from the center of gravity and gradually extends away from the center.

The STM layer plays a critical role in achieving memory-based grouping and segmentation. This layer receives input from the memory layer. In the case of successful memory recall, the recalled pattern imprints on the STM layer. If multiple recalls result from the memory layer, the recalled patterns with proper position adjustment then interact in STM, resulting in a common part. The activated pattern in the STM layer then projects to the segmentation layer (see Fig. 3). The projection from the STM layer to the segmentation layer serves to further group separate segments or segment a single segment in the segmentation layer. Note that the STM layer embodies the final outcome of scene analysis in our model.

III. Model Definition

Before we define each component in Fig. 3 mathematically, let us describe a cycle of computation and the role of each component. When an input scene, generally consisting of multiple objects with occlusions, is presented to the segmentation layer T-junction detection first operates on the scene. The result of T-junction detection is used to sever connections between the two branches of a T-junction. Afterwards, LEGION dynamics ensures that only oscillators in the segmentation layer corresponding to connected components of the input scene are synchronized and different segments are active at different times. An active segment from the segmentation layer is then fed to all the modules of the memory layer, within each of which a successful recall ensues if the active segment is a proper part of the stored pattern. All recalled memory patterns input to the STM layer with position adjustment, where interaction among the recalls leads to the activation of a common part. The activated pattern in the STM layer projects down to the segmentation layer and this top-down input may perform further grouping (and segmentation) by synchronizing different segments of the segmentation layer that are part of the STM pattern. As a result, the original segment in the segmentation layer may recruit other oscillators of the layer to form a larger segment. The following design ensures that the recurrent system of Fig. 3 stabilizes quickly after a segment first emerges from the segmentation layer. This stable activity continues until the segment becomes inactive and another one active, as oscillatory dynamics in the segmentation layer proceeds. After a few oscillation periods, the oscillatory activity in the entire system stabilizes and the result of scene analysis is embodied in the alternating activity of the STM layer.

A. Segmentation Layer

The segmentation layer is defined as a standard LEGION array. Before we specify LEGION dynamics, let us discuss T-junction detection. Due to its importance in organizing 3-D surfaces in a scene, T-junction detection has been studied previously [14], [25]. Given that we deal with 3-D line drawing patterns as illustrated in Fig. 4, we propose the following local and parallel algorithm that efficiently performs T-junction detection in this situation. It is worth noting that some of the ideas extend to more general scenarios, such as curvilinear junctions.

For T-junction detection, we conduct a local topology analysis based on the assumption that lines are one-pixel wide. Observe that a typical T-junction has three branches, two of which fit a smooth contour.

```
Algorithm for T-junction detection for 3-D
    line-drawing scenes (cf. Fig. 5):
    1.  Identify plausible junction
    pixels by checking the number of
    pixels in the eight-nearest neigh-
    borhood of a pixel. If the number is
    greater than 2, the pixel is marked as
    a plausible junction point.

    2.  Group connected plausible junc-
    tion points into junction clusters, by
    iteratively expanding a cluster to ad-
    jacent junction points. {This step is
    needed due to finite resolution of a
    line.}

    3.  For each junction cluster, detect
    its number of adjacent pixels, called
    end points. If the number is 3, con-
    tinue; otherwise, no T-junction for
    this cluster.

    4.  Trace each end point away from
    the cluster until it meets another
    cluster. Estimate a straight line from
    the end point to the meeting point. If
    the maximum distance of all the traced
    pixels to the line is less than 1.2
    pixels, then the orientation of the
    end point is taken to be the orien-
    tation of the line; otherwise back-
    track from the meeting point until the
    threshold is satisfied.

    5.  Check if the orientations of any
    two end points of a junction cluster
    are the same. If so, a T-junction is
    found.
```

Fig. 5 illustrates the algorithm. Fig. 5(a) magnifies a region around a typical T-junction. A detected junction cluster is shown

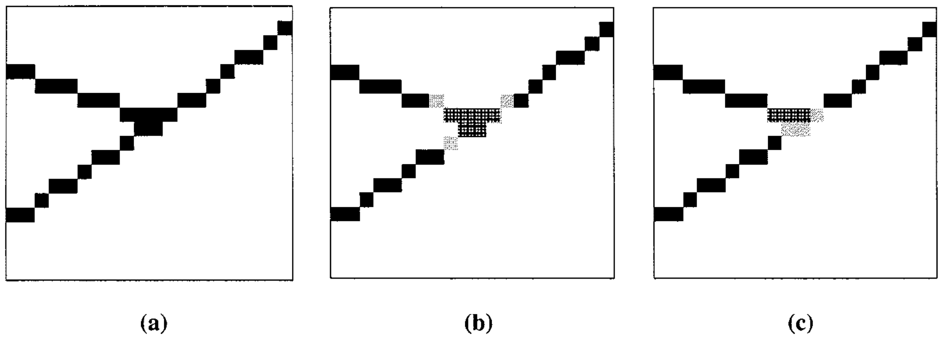


Fig. 5. Example of T-junction detection. (a) Part of an input image with a T-junction. (b) Clusters and end points. Cluster pixels are shown in checker and end pixels in gray. (c) Detected T-junction and the modified lateral connections. Here, the lateral connections between checker and gray pixels are set to zero and other connections remain unchanged.

as the group of checker pixels in Fig. 5(b). In this case, the cluster is found to be a T-junction. As a result, connections between the pixels belonging to the two branches of a T-junction are removed in the LEGION network as defined below. In this case, as shown in Fig. 5(c), the connections between checker and gray pixels are set to zero due to T-junction detection.

Each oscillator i in the LEGION network is defined as

$$\dot{x}_i = 3x_i - x_i^3 + 2 - y_i + I_i + S_i + \rho \tag{1a}$$

$$\dot{y}_i = \varepsilon\left(\alpha\left(1 + \tanh\left(\frac{x_i}{\beta}\right)\right) - y_i\right). \tag{1b}$$

(For more details on the above definition, see [33] and [43]). Here, I_i represents external stimulation to oscillator i, S_i denotes the overall coupling to the oscillator, and ρ is the amplitude of a Gaussian noise term. Noise plays a role of assisting desynchronization in addition to testing the robustness of the system. The parameter ε is a small positive number. Thus, if coupling and noise are ignored and I is a constant, (1) defines a typical relaxation oscillator with two time scales, similar to the van der Pol oscillator [35] (for a recent review, see [40]). The x-nullcline (i.e.,) is a cubic function and the y-nullcline is a sigmoid function [see Fig. 6(a)]. If $I > 0$, the two nullclines intersect only at a point along the middle branch of the cubic. This is shown in Fig. 6(a). In this case, the oscillator produces a stable limit cycle, which alternates between a phase of relatively high x values and a phase of relatively low x values, called the *active* and *silent* phases, respectively [see Fig. 6(a)]. These two phases exhibit near steady-state behavior and correspond to the right branch and the left branch of the cubic, respectively. In contrast, the transition between the two phases takes place rapidly and it is referred to as jumping. The parameter α determines relative times that the limit cycle spends in the two phases—a larger α produces a relatively shorter active phase. If $I < 0$, the two nullclines of (1) intersect also at a stable fixed point on the left branch of the cubic, as shown in Fig. 6(b). In this case, no oscillation occurs. An oscillator is stimulated if $I > 0$ and unstimulated if $I < 0$. As such, oscillations in (1) are stimulus-dependent. In the segmentation layer, an oscillator is stimulated when its corresponding pixel is on an object boundary [or black pixels, see Fig. 5(a)] and otherwise unstimulated. The above oscillator may be interpreted as a model of action potential generation, where x represents the membrane potential of a neuron and y represents the level of activation of ion channels, or oscillating bursts of neuronal spikes. Fig. 6(c) depicts typical x activity.

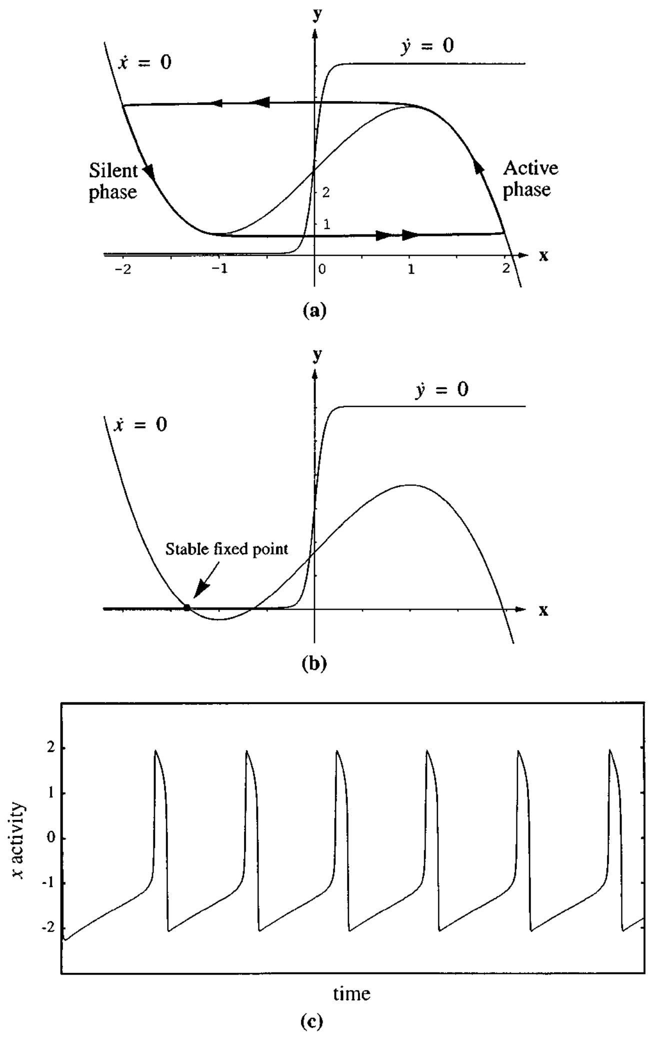


Fig. 6. Behavior of a single relaxation oscillator. (a) A stimulated oscillator. The bold curve indicates the limit cycle of the oscillator and the arrows indicate the direction of motion (double arrows indicate jumping). (b) An unstimulated oscillator, which approaches the stable fixed point. (c) The x activity of the oscillator with respect to time. The parameter values are: $\varepsilon = 0.04$, $\alpha = 9$, $\beta = 0.1$, $\rho = 0.02$, and $I = 0.8$.

In (1a), S_i denotes the coupling from other oscillators in the network as well as inhibition received from global inhibitors

$$\begin{aligned} S_i = & \text{Max}_{k \in N(i)} W_{ik} H(x_k) - W_f H(z_f - \theta) \\ & + W_u I_i H(u_i - \theta)(1 - H(z_s - \theta)) \\ & - W H\left(\sum_k H(u_k - \theta) - \theta\right). \end{aligned} \tag{2}$$

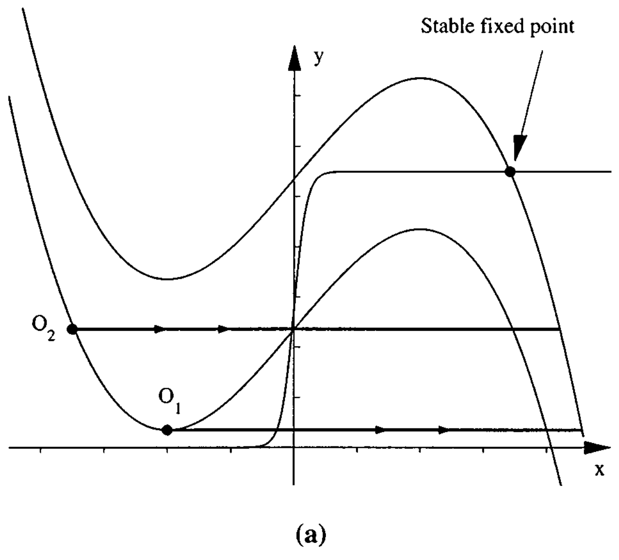
Here, W_{ik} is the connection weight from oscillator k to i and $N(i)$, the coupling neighborhood of i, is a set of eight nearest neighbors of i. W_{ik} equals 1 if and only if both i and k are stimulated and they are not on the different branches of a T-junction; otherwise, W_{ik} equals 0. The symbol H stands for the Heaviside step function. An oscillator affects its neighbors only when it is in the active phase, i.e., when its x activity is positive. In addition, oscillator i receives input from the corresponding unit of the STM layer, whose activity is denoted by u_i. The weight of this top-down connection is W_u and this connection, together with a diffuse top-down inhibition from the STM layer [last term of (2)], helps to achieve memory-based organization. Note that for a stimulated oscillator to receive top-down excitation two of its neighboring oscillators also need to be stimulated and receive top-down excitation. This condition is introduced to ensure that the oscillator corresponding to the intersecting pixel of a T-junction (see Fig. 5) is not recruited by the occluded pattern. θ is a threshold, which is always set to 0.5.

There is a pair of inhibitors within the LEGION layer: a fast inhibitor z_f and a slow inhibitor z_s. In (2), W_f is the weight of fast inhibition and z_f is defined as

$$\dot{z}_f = \phi(\sigma_\infty - z_f) \tag{3}$$

where $\sigma_\infty = 1$ if $x_i \geq \theta$ for at least one oscillator i and $\sigma_\infty = 0$ otherwise. If σ_∞ equals 1, $z_f \to 1$. The parameter ϕ controls how fast z_f reacts to external stimulation and it may be tuned to change the speed of fast inhibition for computational use. This fast global inhibition leads to desynchronization among different oscillator groups, or segments in the segmentation layer.

The lateral connectivity in the LEGION layer, which embodies the results of T-junction detection, supports primitive segmentation. Memory-based organization, on the other hand, relies on top-down input from the STM layer [see (2)]. Depending on the oscillatory input from the segmentation layer, STM activity changes quickly from one pattern to another. In order to be effective, a top-down input must be able to modify the phases of relevant oscillators rapidly; in particular, a top-down input must be able to quickly establish synchrony among oscillator groups that are currently desynchronous. How can one achieve this in continuous phase space? The main idea is to use strong top-down excitation to create a new fixed point on the right branch of the cubic and have oscillators with large phase differences approach the fixed point. As a result, when these oscillators jump down to the left branch of the cubic there can be a large phase compression. (We skip analytical subtleties here and instead refer to Terman and Wang [33] for extensive analysis on phase compression.) This is illustrated in Fig. 7. Fig. 7(a) shows two oscillators, O_1 and O_2, with a large phase separation. A strong top-down excitation lifts the cubic to a much higher position and the fixed point thus created is a stable one, just like the one in Fig. 6(b). Both O_1 and O_2 approach the fixed point. Fig. 7(b) shows a later time, when O_1 and O_2 are much nearer geometrically. Without further perturbation, however, O_1 and O_2 will be trapped at the fixed point. To make them jump back is the role of the slow inhibitor z_s.



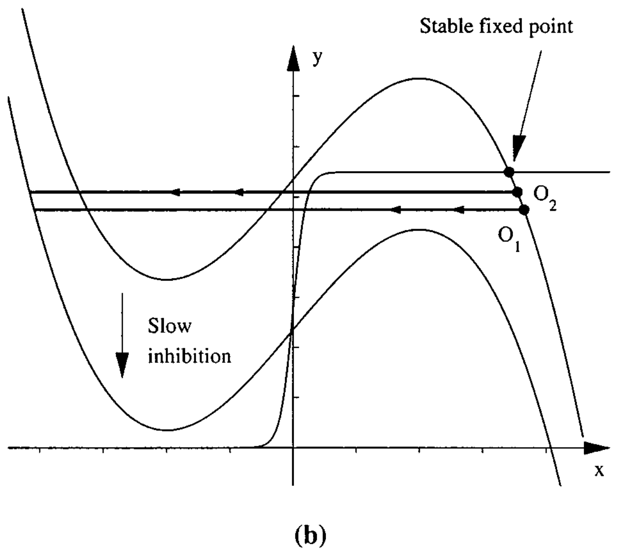


Fig. 7. Top-down synchronization with large phase separation. The cubics are the x-nullclines of the oscillators and the sigmoid is the y-nullcline. (a) Synchrony in jumping up between O_1 and O_2 is induced by strong top-down excitation, which creates a stable fixed point on the higher branch of the sigmoid. (b) Synchrony in jumping down is mediated by slow inhibition.

The slow inhibitor in the LEGION layer is stimulated when any oscillator jumps to the active phase and its activity increases slowly but decays quickly

$$\dot{z}_s = \psi\varepsilon(\sigma_\infty - z_s) - (1 - \sigma_\infty) z_s \tag{4}$$

where ε in the first term signifies slow increase with ψ as a parameter and the second term realizes fast decay. Slow increase gives time for oscillators with large phase differences to get near to the fixed point. When z_s exceeds θ [see (2)], the inhibitor nullifies top-down excitation. When this happens, as shown in Fig. 7(b), both O_1 and O_2 jump down to the silent phase with a much smaller phase difference than when they started in Fig. 7(a).

B. Memory Layer

The memory layer consists of multiple memory modules, each of which stores one pattern and is a locally connected network with global inhibition. This architecture is very similar to a standard LEGION network. However, we do not employ LEGION since, within each module, no multiple recalls can occur, thus no need for oscillations. A locally coupled network can be used for storage because each stored pattern is by itself a connected pattern. As mentioned in the previous section, local connectivity results in substantial reduction of connections, compared to standard associative memory models with all-to-all connectivity. Each stored object is a line drawing pattern. Initial storage is simply set so that neighboring units have a connection strength of 1 if they both belong to the pattern and -1 otherwise.

When there is no external stimulus to a module, the module stays silent. When a stimulus occurs, the units corresponding to the stimulus are activated. Activated units that belong to the stored pattern propagate the activation through local positive links, whereas activated units that do not belong to the memory pattern trigger the global inhibitor that in turn shuts off the entire module. Thus, the global inhibitor makes sure that only a subset of the units corresponding to the stored pattern can produce a successful recall. In addition, it is desirable that a recalled pattern does not extend beyond the stimulated regions of the input scene. In Fig. 4, for example, this means that a recall is considered unsuccessful if the recalled pattern, when aligned with the input segment that triggers the recall, extends beyond the areas occupied by the two objects. This can also be implemented via the global inhibitor. A successfully recalled pattern projects to the STM layer.

More specifically, a unit m_i in a memory module is defined as

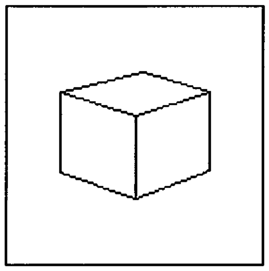
$$\dot{m}_i = S_i^M - m_i \tag{5}$$

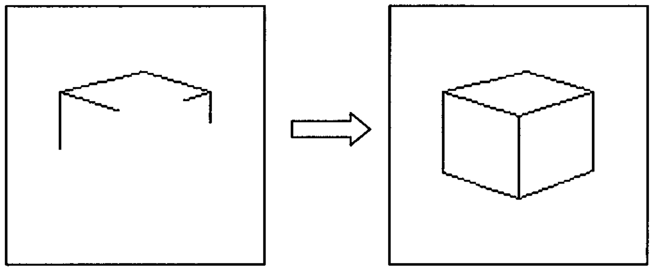
where S_i^M indicates total input to the unit (superscript M indicates the memory layer), defined as

$$S_i^M = H\left(I_i^M + \sum_{k \in N(i)} W_{ik}^M H(m_k - \theta)\right) - H(z_m - \theta) \tag{6}$$

where I_i^M, a binary number, represents external input from the segmentation layer, W_{ik}^M is the local coupling weight that encodes the stored pattern, and z_m denotes the global inhibitor of the memory module.

The global inhibitor z_m in this layer becomes activated when at least one unit that does not belong to the stored pattern is active, or when a unit is active that does not correspond to any


(a)


(b)

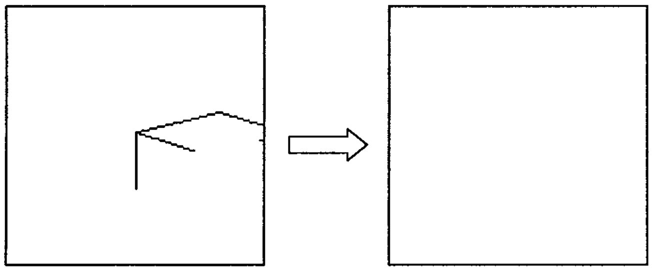

(c)

Fig. 8. Example of memory recall. (a) A stored pattern. (b) An input image shown on the left leads to a recall shown on the right. (c) Another input image shown on the left leads to a recall shown on the right.

stimulated area of the input scene. Also, the inhibitor plays a role of resetting when the external input is withdrawn:

$$\dot{z}_m = \sum_i \left[\left(H(m_i^- - \theta^-\right) + H(m_i - \theta)(1 - R_i)\right] + H\left(\theta - \sum_i I_i^M\right) - z_m \tag{7}$$

where m_i^- indicates the activity of a unit that is not involved in storage and θ^- is a threshold set to 0.1. R_i is 1 if the corresponding unit in the segmentation layer belongs to a stimulated region and 0 otherwise. Thus, the second part within the first term detects whether a memory unit corresponding to no stimulated area is activated. The third Heaviside function implements resetting when there is no external stimulation.

Fig. 8 shows an example of memory recall. Fig. 8(a) is a stored pattern. In Fig. 8(b), an incomplete input but correctly aligned with the stored pattern is presented. Units that directly receive the external input are first activated and their activation quickly propagates until the entire stored pattern is recalled. The right side of Fig. 8(b) shows a snapshot of the stable activity of the memory module after thresholding with θ [see (8)], whereby partial pattern completion is achieved successfully. When the input is withdrawn, the global inhibitor is activated and the network quickly settles to the silent state. Fig. 8(c) shows the same input pattern but shifted so that it is no longer aligned with the stored pattern. When it is presented, even though some units receive direct input, the module remains below the threshold because of the global inhibition. We point that, although a memory module does not exhibit translation invariance, the system as a whole does, as explained in Section III-D.

C. STM Layer

The STM layer computes an input-aligned intersection of recalled patterns from the memory layer. Given its simple function, the STM layer does not have lateral connections within the layer. Specifically, unit u_i is defined as

$$\dot{u}_i = H\left[\sum_p H\left(m_i^p - \theta\right) - \sum_p H\left(\sum_i H\left(m_i^p - \theta\right) - \theta\right)\right] - u_i \quad (8)$$

where p indexes a memory module. The first summation in (8) represents the corresponding projections from all the memory modules. These projections are based on locations. The second summation computes how many memory modules are activated; a memory module is activated if one of its units is active. This subtractive and diffuse inhibition ensures that only those STM units that are stimulated by every activated memory module can be triggered.

In summary, the STM layer becomes activated when triggered by the memory layer. Because the memory layer does partial pattern completion, the active pattern in the STM layer represents a whole pattern, not a fragment. When multiple patterns are recalled from different memory modules, the activated pattern in STM corresponds to the input-aligned intersection of these recalled patterns.

D. Further Notes on Interactions Between Layers

To ensure the stability of the entire network, an active segment is input to the memory layer only when the segment is stable; in other words, when all the oscillators of the segment are in the active phase. This is achieved by a window function proposed by Wang [39] that prevents the projection to the memory layer when oscillators are either jumping up or down. Also, to compensate for positional misalignment due to the possibility that an active segment from the segmentation layer is only part of a pattern, systematic shifting away from the center of gravity is performed before the segment reaches each individual memory module. Such shifting might be performed by a hypothesized shifter circuit in the brain [1]. Like the shifter circuit, feedforward shifts from the segmentation layer to the memory layer are paired with the feedback shifts from the memory layer to the STM layer so that the pattern recalled in STM reflects the position of the input segment. This is illustrated in Fig. 9, where a part of a triangle pattern in the segmentation layer recalls a whole triangle and the recalled triangle is positioned in the STM with proper alignment with the input segment. In other words, the position of an input pattern anchors those of recalled patterns. Only in this way can an appropriate intersection be applied to multiple recalled patterns. This is again illustrated in Fig. 9, whereby another pattern—a parallelogram—is recalled and STM records as a result a common part between the two objects. Because of the dynamics in memory recall, such intersection must contain the input segment as a part and thus at least as large as the input segment.

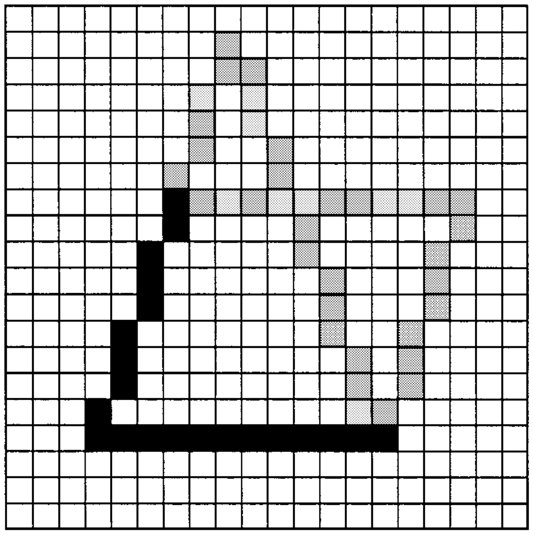


Fig. 9. STM layer performing an intersection of multiple recalled patterns from the memory layer. Two recalled patterns are a triangle and a parallelogram. The black pixels denote the common part between the two patterns and it represents STM activity.

From Fig. 9, it may seem desirable that the two objects are both recalled, instead of their intersection. This would be readily achieved by introducing oscillations into the STM layer. However, we consider that, on the whole, intersection is a better choice because the input is underconstrained when it can yield multiple recalls. In general, underconstrained input may yield many "hallucinated" patterns. Imagine seeing black through a small aperture. This black aperture is consistent with countless patterns, but it would be perceived as the aperture itself, not the patterns that are compatible with the aperture. On the other hand, we acknowledge that, under certain conditions, it may be more appropriate for an underconstrained input to recall each of the compatible patterns, particularly if the number of such patterns is small and when some recalled patterns can trigger memory-based grouping thus resolving the ambiguity.

From the definition of the system it is clear that only the segmentation layer generates intrinsic oscillations. Other layers essentially follow limit-point dynamics. It is worth noting that driven by oscillations in the LEGION layer both the memory layer and the STM layer exhibit oscillations too. Such oscillations are driven by oscillating input and they should be distinguished from intrinsic oscillations in the segmentation layer. Memory-based organization is carried out through top-down connections from the STM layer to the LEGION network. These top-down connections are nonspecific to individual patterns (see

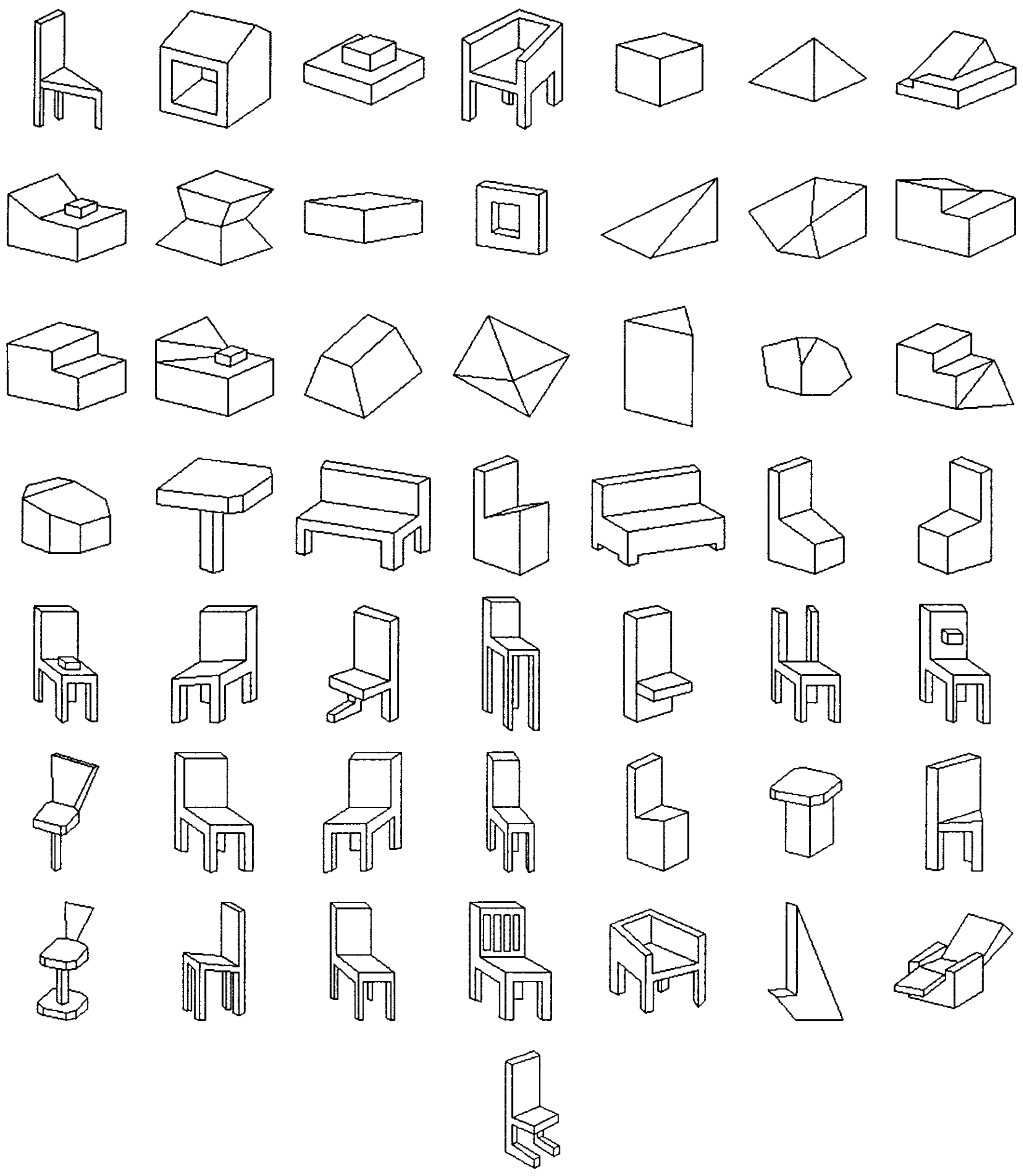


Fig. 10. Database of 50 line-drawing objects. These objects are stored in the memory layer. These patterns are available online at the anonymous ftp site "ftp.cis.ohio-state.edu/pub/leon/Liu."

Fig. 3), but carry only the phase information from the STM layer to the LEGION network. Top-down organization is both mediated by and embedded in phase. Because of the interaction between primitive segmentation and memory-based organization, the system usually takes a number of oscillation cycles to reach a stable solution.

IV. Results

A. Test Database

To systematically evaluate the performance of our system, we use a database of 50 3-D line drawing objects. These objects were created by Stark and Bowyer [32] for 3-D object recognition. Because we were unable to use their generation program, we scanned in their objects from the paper and selected 50 from a total of 101 objects that had reasonable scanning quality. From the scanned version, we then extracted vertices and reproduced a line drawing version, which is almost identical to the original. Fig. 10 shows the 50 discretized line-drawing objects used in our evaluation. We now provide these objects in both picture and line-drawing formats online to facilitate use and comparison by other researchers.

For all of the subsequent simulations, each memory module has 256×256 units and stores one discretized object (see Fig. 10). Both the segmentation layer and the STM layer consist of 512×512 units.

B. Memory-Based Grouping and Segmentation

The system defined in Section III is implemented and differential equations are solved using the fourth-order Runge–Kutta method. The following parameter values are used: $\varepsilon = 0.02$, $\alpha = 6.5$, $\beta = 0.1$, $\rho = 0.0005$, $\phi = 0.5$, and $\psi = 0.375$.

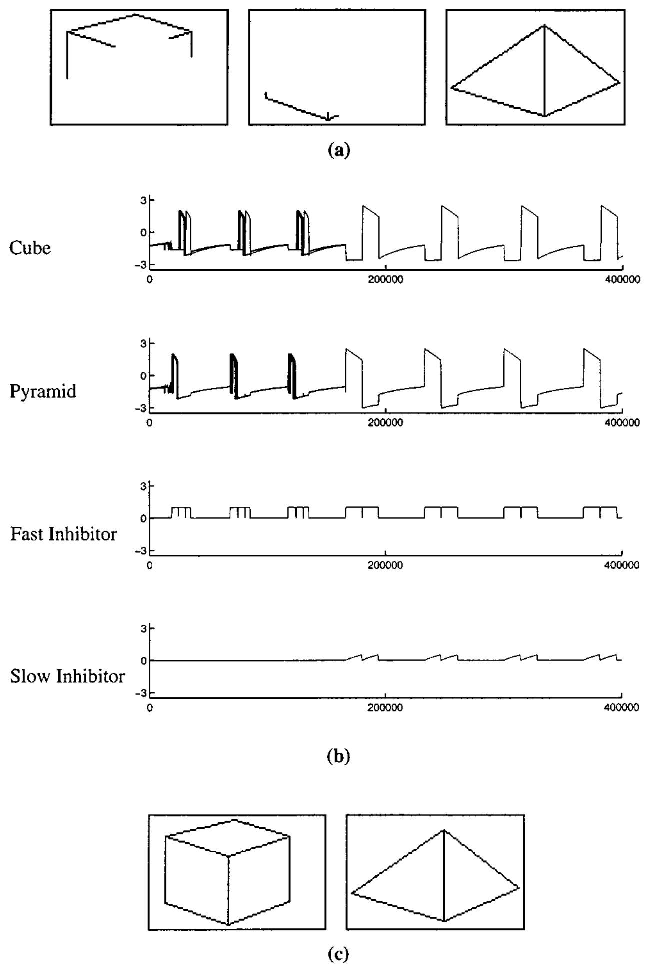


Fig. 11. Memory-based grouping with the input scene of Fig. 4. (a) Result of primitive segmentation, which yields three segments shown as three snapshots of the segmentation layer. As in subsequent figures, some cropping is done to highlight areas occupied by patterns. (b) Temporal behavior of the segmentation layer. The top two panels show the combined x activity for the cube and the pyramid, respectively. The bottom two panels show the activity for the fast inhibitor and the slow inhibitor, respectively. A total number of 400 000 integration steps are shown and top-down input is activated at integration step 160 000. (c) Two snapshots of the STM layer, showing the cube and the pyramid, respectively.

The following connection weights are used in (2): $W_{ik} = 1.75$, $W_f = 1.5$, $W_u = 15.5$, and $W = 8.5$. The system behavior is robust for a considerable range of parameter values. Moreover, the appropriate range of parameter values can be derived from a singular perturbation analysis as done in [33].

There are two stages of computation: the first stage performs primitive segmentation based on T-junction detection and LEGION dynamics and the second stage performs memory-based organization. In our simulations, to simplify computation, primitive segmentation is performed first without top-down involvement or slow inhibition. From earlier analysis on LEGION networks, it is clear that this stage takes at most as many oscillation periods as the segmentation capacity [33], [44], which can be calculated directly [15].

We first demonstrate memory-based grouping in Fig. 11. Fig. 11(a) shows the result of primitive segmentation from the segmentation layer, after T-junction detection. Due to occlusion, the cube is broken into two disconnected segments. Fig. 11(b) shows the temporal traces of the segmentation layer. From these traces one can easily tell that the primitive segmentation stage ends when the slow inhibitor is activated; this also corresponds to higher x activity shown in the first two traces. The first segment emerging from the segmentation layer after the second stage starts is the pyramid, as shown in the second panel of Fig. 11(b). While it is active, the oscillators corresponding to the other two segments are silent. This segment triggers the memory layer and the recalled pattern that is input to the STM layer is the stored pyramid. After the first segment jumps down to the silent phase, all the activity in the memory layer and the STM layer disappears. The second segment jumping to the active phase is a part of the cube. This segment recalls the entire cube from the memory layer, which then enters STM. The retained cube in STM feeds to the segmentation layer and this top-down input synchronizes the entire cube on the segmentation layer [see (2)], as shown in the top panel of Fig. 11(b). Thus, driven by memory, the two parts of the cube are grouped. Fig. 11(c) shows the instantaneous activity (snapshot) of the STM layer at two different times, corresponding, respectively, to when the cube and the pyramid are active. Note that the units corresponding to the whole cube are simultaneously active in the STM layer.

Fig. 12 demonstrates how memory helps further segmentation where primitive segmentation fails due to the failure of T-junction detection. Fig. 12(a) shows an input image with two objects, one of which partly occludes another. Fig. 12(b) shows the result of initial segmentation after T-junction detection. Two kinds of error occur. First, the two-block platform is segmented into two segments due to correct T-junction detection. We call it an error for the following reason even though T-junction detection does what it is supposed to do. A single object—the platform—is broken into pieces by primitive segmentation, which fails to perform correct scene analysis if nothing else is done. Second, due to the arrangement of the two objects, the middle intersection point between the two objects is hard to detect by any T-junction detection algorithm. As a result, the two objects form the same segment. Because the top segment can uniquely recall the memory pattern of the platform, the activated STM layer through its top-down projections recruits the lower block of the platform. As a result, the oscillators corresponding to the whole platform are synchronized in the segmentation layer. This is shown in the top panel of Fig. 12(c), which gives the temporal traces of the segmentation layer. An additional effect of this is that the combined segment of cube/platform is properly separated in phase space and after the platform segment jumps down the nonoccluded part of the cube jumps up. Thus, driven by memory a wrong segment is further and correctly separated. Such segmentation then results in the completion of the whole cube pattern. As shown in Fig. 12(c), the system behaves correctly afterwards. As in Fig. 11(c), Fig. 12(d) shows two snapshots of the STM layer, when the cube and the platform are active, respectively.

C. Systematic Evaluation

With the full database of 50 objects and a large number of simulations with different scenes composed of these objects, numerically solving the differential equations becomes computa-

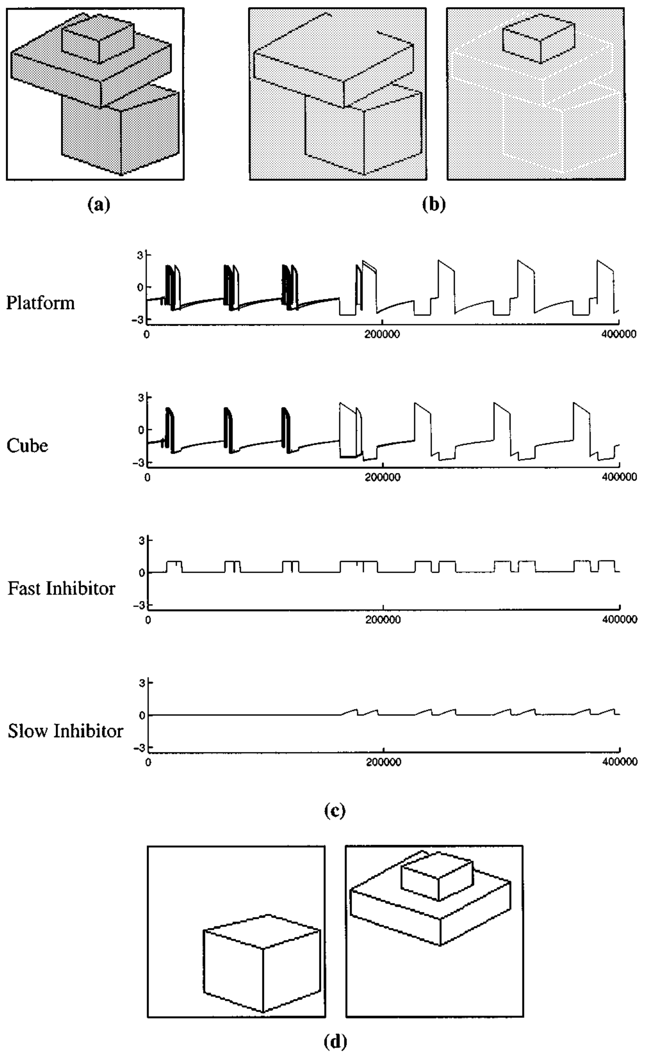


Fig. 12. Memory-based segmentation. (a) An input scene that consists of two patterns: a platform and a cube, the former occluding the latter. (b) Result of primitive segmentation, which yields two segments shown as two snapshots of the segmentation layer. (c) Temporal behavior of the segmentation layer. The top two panels show the combined x activity for the platform and the cube, respectively. The bottom two panels show the activity for the fast inhibitor and the slow inhibitor, respectively. A total number of 400 000 integration steps are shown and top-down input is activated at integration step 160 000. (d) Two snapshots of the STM layer, showing the platform and the cube, respectively.

tionally prohibitive. Thus, we have abstracted a clock-based algorithm that produces equivalent behavior to the original system and used the algorithm for the following evaluation. The clock algorithm labels a segment by an integer that increases by 1 every time when an active segment jumps down, except for the active segment itself, which is labeled "1" when it jumps down to the silent phase. The segment to jump up next is the one with the largest label.

In order to systematically test the model's capability in analyzing various scenes, we compose scenes that are composed of two to six patterns; these patterns are randomly positioned and have random depth relations for producing proper occlusion. Given that disconnected parts of an image are readily segmented in the segmentation layer, we require that each test scene is spatially connected. Note that, even with this stipulation, a huge number of scenes can be so composed.

One such scene is shown in Fig. 13(a), with four objects: one table, two chairs, and one recliner. These objects are arranged in a relatively realistic setting with proper occlusion. Fig. 13(b) shows the result of primitive segmentation. This stage yields nine segments, none of which corresponds to a single object. The result of the integrated analysis by our system is shown in Fig. 13(c) with four subsequent snapshots of the STM layer. In this case, all four objects are successfully segmented and recalled.

Fig. 14 documents the rate of successful analysis with respect to the number of objects in a scene. Note that each disconnected component in an input scene is analyzed independently; thus, we use only connected scenes. We include the case of one object for comparison purposes, since the system is known to behave correctly. By a correct analysis we mean that every object is correctly segmented and recognized by the memory layer. Each data point in Fig. 14 is derived from 1000 random test trials. The upper curve shows the result of the integrated system and the lower curve shows, for comparison, the result of primitive segmentation alone. For scenes with multiple objects, the system achieves good success rates, which decrease as the number of objects composing a scene increases. The performance of the integrated system is much better than that of primitive segmentation alone.

The main reason for analysis errors in the integrated system is that more line drawings crowd a scene and T-junction detection tends to produce more errors. Many of such scenes are hard for humans to discern. Fig. 15(a) shows a typical failure example, which is composed of four objects: two chairs, one bench, and one triangle-like wedge. Fig. 15(b) shows the result of primitive analysis, which yields two segments due to failed T-junction detection. The failure is mainly caused by accidental alignment between different objects, which happens from time to time. As shown in Fig. 15(c), no object is recalled as a result.

Corrupting a scene by adding noisy line drawings does not disrupt system performance so long as such drawings can be segmented from the rest of the scene via primitive segmentation. Even when primitive segmentation fails, memory-based segmentation can correct many errors (see Fig. 12). Obviously, there are always scenarios when the system is fooled by such corruption, which is reflected by the fact that the system performance decreases with more line drawings on a scene. On the other hand, removing parts of an object on an input scene leads to a situation similar to that caused by occlusion.

V. Discussion

The system proposed here represents an integrated approach to scene analysis. When a scene is presented, the system first performs primitive segmentation using a LEGION layer, which yields multiple segments in time. As a result, later stages of the system can concentrate on processing a segment at a time, including positional alignment and recognition. This processing strategy resembles the essential role of visual attention in human scene analysis [24], [26]. It is well documented that visual attention selects a part of a scene at a time for further processing and different parts of a scene are processed through shifts of attention. Such selection is carried out either overtly

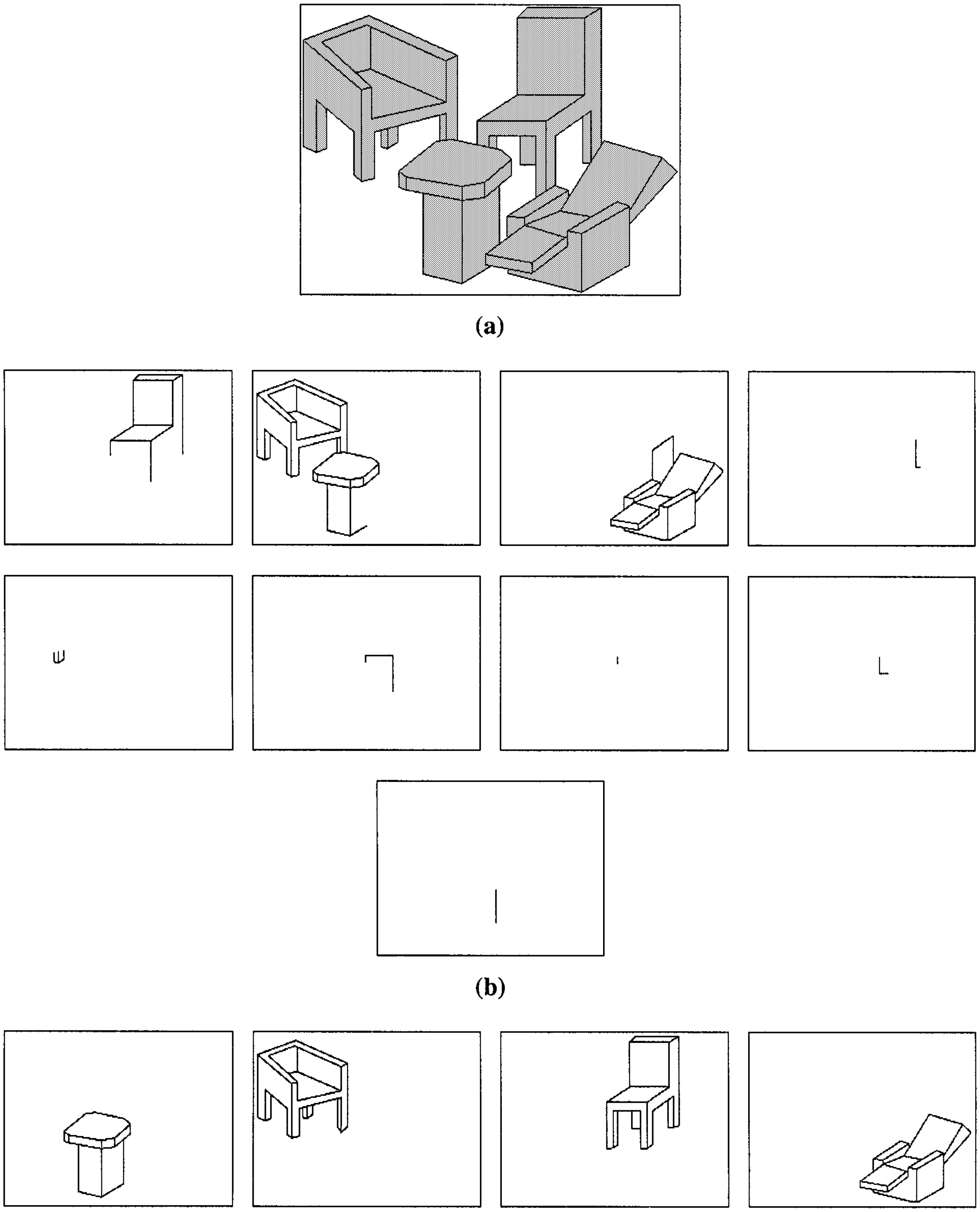


Fig. 13. Another scene-analysis example. (a) An input scene with four line-drawing objects. (b) Result of primitive segmentation, which yields nine segments shown as nine snapshots of the segmentation layer. (c) Four snapshots of the STM layer, showing the four objects resulting from integrated analysis.

through eye movements, or covertly when the eyes stay still [26]. A critical issue is what attracts attention, locations or objects? In other words, is attention object-based and location-based? This is an unresolved issue in visual psychophysics, although recent evidence seems to be more supportive of the object-based view [20], [22] [24]. Our model is consistent with object-based attention (see also [39]), because the segmentation layer yields segments that are organized structures. Our model

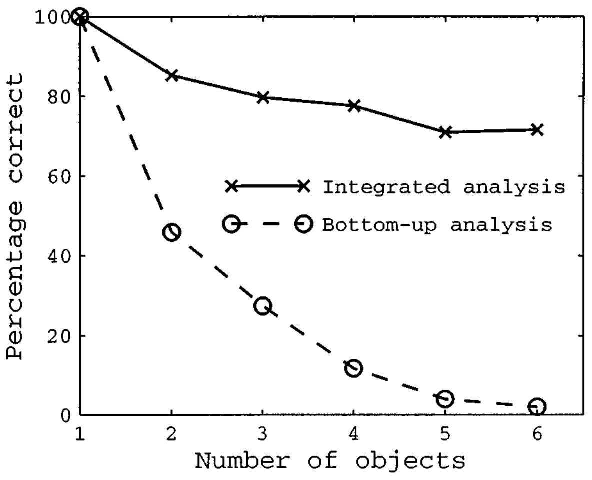


Fig. 14. Correct analysis rate with respect to the number of objects in a connected scene. Each data point is obtained from 1000 randomly generated connected scenes. The solid line shows the correct percentage of the integrated system and the dash line shows the correct percentage of primitive segmentation.

further suggests that "objects" that are attended to result from primitive segmentation. This is consistent with recent evidence from studying both healthy subjects and neuropsychological patients suggesting that attention applies to segments directly [4]. On the other hand, our model differs from the attention models of Koch [12] and Wolfe [45]; in these models, selection is based on winner-take-all competition between various locations of different activation levels that are determined by local feature detection and thus corresponds to location-based attention.

It is interesting to note that attention has a limited capacity in terms of how many objects can be simultaneously attended to ("4" is an often cited number for visual attention [26]). The limited capacity of visual attention is quite compatible with the notion of a segmentation capacity in LEGION. The latter refers to that a LEGION network, due to limited temporal resolution, can segment only a limited number of patterns [44]. This capacity is closely related to the ratio of an oscillation period to the interval of an active phase of a single oscillator (cf. Fig. 11).

It is perhaps revealing to contrast the above capacity mechanism with the capacity theory of the CAPS cognitive architecture [9], [10]. In CAPS, implemented as a production system with a working memory, the capacity is determined by the total amount of activation available in working memory. Total activation can be divided among different processes (productions) and newly fired processes may reduce the activation levels of current processes in working memory, thus preventing the total activation from exceeding a threshold. The main difference between the two capacity mechanisms is that, in LEGION, capacity results from the time when an oscillating segment is in the silent phase, the period within which other segments can reach the active phase. Thus, the LEGION capacity is an intrinsic property of *time*, whereas the notion of total activation is not. On the other hand, one could view the LEGION mechanism as a neural realization of the activation notion in CAPS. Unified treatment of perception, memory and attention has also been studied previously in symbolic models of cognitive architecture, among which EPIC [21] and ACT-R/PM [3] are prominent examples. Both EPIC and ACT-R/PM include a perceptual (and motor) module, but do not assume an explicit, limited capacity for information processing. Rather capacity stems from the demand that multiple tasks may have to use the same perceptual/motor processes and thus have to be scheduled. Our system targets at a lower-level processing than EPIC and ACT-R/PM. For example, the visual module in those models assumes a format of input that is highly

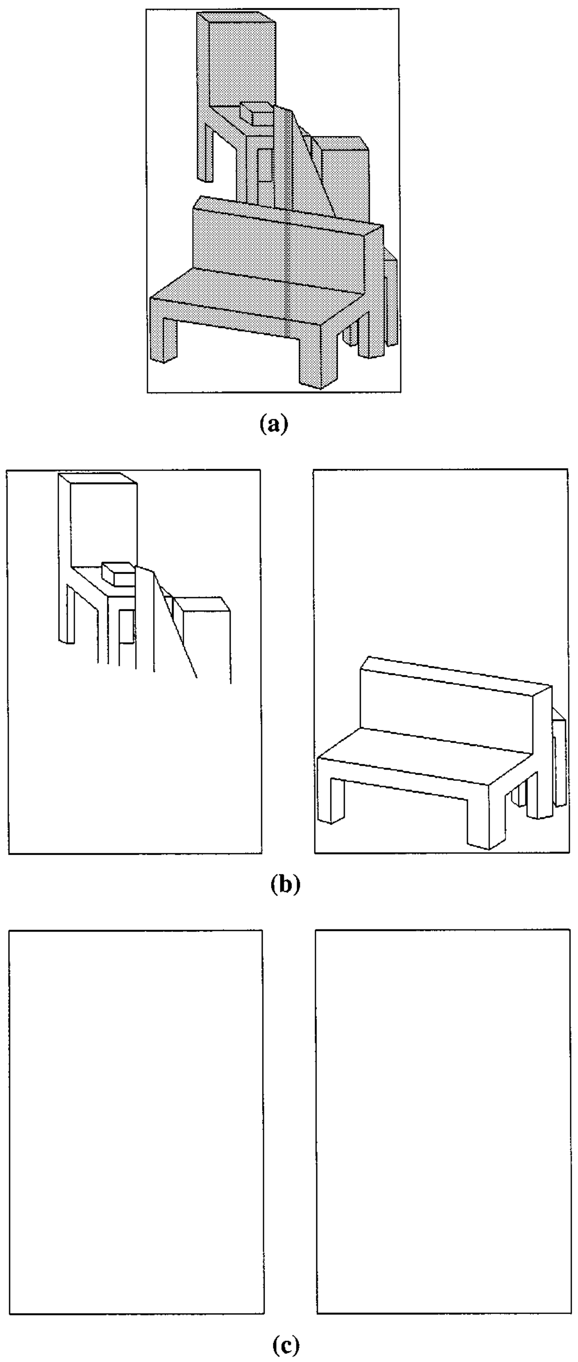


Fig. 15. Example of failed scene-analysis. (a) An input scene with four line-drawing objects. (b) Result of primitive segmentation, which yields two segments shown as two snapshots of the segmentation layer. (c) The two corresponding snapshots of the STM layer, showing no object resulting from integrated analysis.

processed, including shape and size. In other words, unlike our system, they do not deal with stimuli at the level of pixels. On the other hand, the higher level of modeling enables those systems to explain psychological data, which our model does not address presently. It would be a fascinating topic for future research to investigate whether our system possesses similar explanatory power as those of CAPS, EPIC, and ACT-R/PM.

Although the memory layer stores each pattern in a separate memory module, our system exhibits context sensitivity through the pathway from the memory layer to the STM layer. Take for example a pattern "*OSU*" that has a part "*S*." If "*OSU*" is the only pattern stored, an input "*S*" recalls the whole pattern of "*OSU*." When there are other stored patterns in the memory that also have "*S*" as a part (such as "*USC*"), the presentation of "*S*" recalls only the common part among "*OSU*" and those stored patterns. Thus, what is recalled by a particular input depends on the whole memory, or the rest of the memory is the context for a specific stored pattern. This context dependency has an interesting effect: when memory becomes larger in size the system tends to need a larger part to uniquely recall a stored pattern.

Our system can readily distinguish a new pattern from a stored pattern. When a new pattern is contained in an input scene, after segmentation there will be a time when this pattern is active in the segmentation layer while the pattern does not recall any pattern from the memory layer and thus the STM layer is totally silent. This time window could provide an opportunity for the memory layer to store the pattern. This behavior is different from that of many other associative memory models (e.g., [7]), which cannot distinguish a novel input from stored patterns but instead force memory to produce a recall every time.

It is worth emphasizing that our system performs scene analysis entirely in phase space, or in time. Primitive segmentation is carried out by the LEGION network, which yields intrinsic oscillations and organizes an input scene into different segments that correspond to synchronous oscillations. Other parts of the system exhibit driven oscillations and synchrony because of the oscillatory input from the segmentation layer. Our phase space mechanism provides a way to unify bottom-up and top-down analyses in neural networks.

There is a growing body of neurobiological evidence that supports synchronous oscillations as a neural mechanism for binding sensory features into perceptual objects [11], [17], [29]. In addition, recent psychophysical results lend strong support to the idea that phase relations among sensory features give rise to perceptual organization [5], [13], [34].

Our system has a number of limitations. Although it provides a solution to the alignment problem and translation invariance, it does not address other forms of invariance in pattern recognition, such as size, rotation, or distortion. To deal with size and rotation invariance, a promising extension would allow a systematic way of varying the spatial scale and orientation of a segment from the segmentation layer, in addition to shifting (see also [1]). Such variation should be limited to a plausible range. To deal with distortion invariance, the current recall method, which yields a successful match only when a segment is a proper part of a stored pattern, should be extended so that a substantial match with a stored pattern is sufficient to produce a recall. Other challenging recognition issues, such as recognizing nonrigid 3-D objects (e.g., a human body) from two-dimensional views, are clearly beyond the capability of our exceedingly simple memory system. As acknowledged earlier, our design of the STM model does not allow a single segment to recall more than one pattern. On the other hand, visual perception reveals that there are genuinely ambiguous figures, where appropriate processing should output multiple interpretations instead of a common subset among these interpretations [23]. Another limitation occurs when a segment recalls the same pattern multiple times at different positions. Again in this situation, the STM layer finds the common part of the multiple occurrences of the same pattern. If the segment does not recall any other pattern in the memory layer, the appropriate output from the STM layer should be just that pattern. These limitations notwithstanding, the issues dealt with in this paper, such as memory-based grouping and segmentation, must be addressed in any comprehensive scene-analysis system.

To conclude, we have proposed a scene-analysis network that integrates primitive segmentation and associative memory, and the integration is achieved in phase space. When a scene is presented to the system, it is first segmented by a LEGION network, which produces alternating segments. At any time, at most one segment interacts with the memory layer and the common part from multiple recalls is registered in the STM layer. Through the top-down projections from STM, the segmentation layer performs further grouping (pattern completion) and segmentation. As a result, the integrated system exhibits complete location invariance as well as context sensitivity in memory recall and handles object occlusion properly, overcoming several hurdles in neural computing. We have demonstrated that memory-based organization significantly improves the scene-analysis performance.

Acknowledgment

The authors thank the three anonymous referees whose comments improved the presentation of this paper.

References

[1] C. H. Anderson and D. C. van Essen, "Shifter circuits: A computational strategy for dynamic aspects of visual processing," in *Proc. Nat. Acad. Sci. USA*, vol. 84, 1987, pp. 6297–6301.

[2] A. S. Bregman, *Auditory Scene Analysis*. Cambridge, MA: MIT Press, 1990.

[3] M. D. Byrne, "ACT-R/PM and menu selection: Applying a cognitive architecture to HCI," *Int. J. Human-Comp. Stud.*, 2001.

[4] J. Driver and G. C. Baylis, "Attention and visual object recognition," in *The Attentive Brain*, R. Parasuraman, Ed. Cambridge, MA: MIT Press, 1998, pp. 299–326.

[5] M. A. Elliott and H. J. Müller, "Synchronous information presented in 40-Hz flicker enhances visual feature binding," *Psychol. Sci.*, vol. 9, pp. 277–283, 1998.

[6] S. Grossberg and L. Wyse, "A neural network architecture for figure-ground separation of connected scenic figures," *Neural Networks*, vol. 4, pp. 723–742, 1991.

[7] J. J. Hopfield, "Neural networks and physical systems with emergent collective computational abilities," *Proc. Nat. Acad. Sci. USA*, vol. 79, pp. 2554–2558, 1982.

[8] D. Horn and M. Usher, "Parallel activation of memories in an oscillatory neural network," *Neural Comput.*, vol. 3, pp. 31–44, 1991.

[9] M. A. Just and P. A. Carpenter, "A capacity theory of comprehension: Individual differences in working memory," *Psychol. Rev.*, vol. 99, pp. 122–149, 1992.

[10] M. A. Just, P. A. Carpenter, and S. Varma, "Computational modeling of high-level cognition and brain function," *Human Brain Map.*, vol. 8, pp. 128–136, 1999.
[11] A. Keil, M. M. Mueller, W. J. Ray, T. Gruber, and T. Elbert, "Human gamma band activity and perception of a gestalt," *J. Neurosci.*, vol. 19, pp. 7152–7161, 1999.
[12] C. Koch and S. Ullman, "Shifts in selective visual attention: toward the underlying neural circuitry," *Human Neurobiol.*, vol. 4, pp. 219–227, 1985.
[13] S. H. Lee and R. Blake, "Visual form created solely from temporal structure," *Science*, vol. 284, pp. 1165–1168, 1999.
[14] T. Lindeberg, "Feature detection with automatic scale selection," *Int. J. Comput. Vis.*, vol. 30, pp. 79–116, 1998.
[15] P. S. Linsay and D. L. Wang, "Fast numerical integration of relaxation oscillator networks based on singular limit solutions," *IEEE Trans. Neural Networks*, vol. 9, pp. 523–532, 1998.
[16] X. W. Liu, D. L. Wang, and J. R. Ramirez, "Boundary detection by contextual nonlinear smoothing," *Pattern Recognit.*, vol. 33, pp. 263–280, 2000.
[17] M. S. Livingstone, "Oscillatory firing and interneuronal correlations in squirrel monkey striate cortex," *J. Neurophysiol.*, vol. 75, pp. 2467–2485, 1996.
[18] C. Lourenco, A. Babloyantz, and M. Hougardy, "Pattern segmentation in a binary/analog world: unsupervised learning versus memory storing," *Neural Networks*, vol. 13, pp. 71–89, 2000.
[19] Q. Ma, "Adaptive associative memories capable of pattern segmentation," *IEEE Trans. Neural Networks*, vol. 7, pp. 1439–1449, 1996.
[20] J. B. Mattingley, G. Davis, and J. Driver, "Preattentive filling-in of visual surfaces in parietal extinction," *Science*, vol. 275, pp. 671–674, 1997.
[21] D. E. Meyer and D. E. Kieras, "A computational theory of executive cognitive processes and multiple-task performance—Part I: Basic mechanisms," *Psychol. Rev.*, vol. 104, pp. 3–65, 1997.
[22] K. Nakayama, Z. J. He, and S. Shimojo, "Visual surface representation: A critical link between lower-level and higher-level vision," in *An Invitation to Cognitive Science*, S. M. Kosslyn and D. N. Osherson, Eds. Cambridge, MA: MIT Press, 1995, pp. 1–70.
[23] S. E. Palmer, *Vision Science*. Cambridge, MA: MIT Press, 1999.
[24] R. Parasuraman, *The Attentive Brain*, R. Parasuraman, Ed. Cambridge, MA: MIT Press, 1998.
[25] L. Parida, D. Geiger, and R. Hummel, "Junctions: Detection, classification and reconstruction," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 20, pp. 687–698, 1998.
[26] H. E. Pashler, *The Psychology of Attention*. Cambridge, MA: MIT Press, 1998.
[27] B. D. Ripley, *Pattern Recognition and Neural Networks*. Cambridge, U.K.: Cambridge Univ. Press, 1996.
[28] N. Shareef, D. L. Wang, and R. Yagel, "Segmentation of medical images using legion," *IEEE Trans. Med. Imag.*, vol. 18, pp. 74–91, 1999.
[29] W. Singer and C. M. Gray, "Visual feature integration and the temporal correlation hypothesis," *Ann. Rev. Neurosci.*, vol. 18, pp. 555–586, 1995.
[30] H. Sompolinsky and M. Tsodyks, "Segmentation by a network of oscillators with stored memories," *Neural Comput.*, vol. 6, no. 4, pp. 642–657, 1994.
[31] M. Sonka, V. Hlavac, and R. Boyle, *Image Processing, Analysis and Machine Vision*, 2nd ed. Pacific Grove, CA: PWS, 1999.
[32] L. Stark and K. Bowyer, "Achieving generalized object recognition through reasoning about association of function to structure," *IEEE Trans. Pattern Anal. Machine Intell.*, vol. 13, pp. 1097–1104, 1991.
[33] D. Terman and D. L. Wang, "Global competition and local cooperation in a network of neural oscillators," *Phys. D*, vol. 81, pp. 148–176, 1995.
[34] M. Usher and N. Donnelly, "Visual synchrony affects binding and segmentation in perception," *Nature*, vol. 394, pp. 179–182, 1998.
[35] B. van der Pol, "On 'relaxation oscillations'," *Philos. Mag.*, vol. 2, no. 11, pp. 978–992, 1926.
[36] C. van der Malsburg, "The correlation theory of brain function," Max-Planck-Institute for Biophysical Chemistry, Int. Rep. 81-2, , 1981.
[37] C. van der Malsburg and W. Schneider, "A neural cocktail-party processor," *Biol. Cybern.*, vol. 54, pp. 29–40, 1986.
[38] D. L. Wang, "Pattern recognition: Neural networks in perspective," *IEEE Expert*, vol. 8, pp. 52–60, Aug. 1993.
[39] ——, "Object selection based on oscillatory correlation," *Neural Networks*, vol. 12, pp. 579–592, 1999.
[40] ——, "Relaxation oscillators and networks," in *Encyclopedia of Electrical and Electronic Engineers*, J. Webster, Ed. New York: Wiley, 1999, pp. 396–405.
[41] ——, "Visual scene segmentation: Neural networks," in *Handbook of Brain Theory and Neural Networks*, 2nd ed, M. A. Arbib, Ed. Cambridge, MA: MIT Press, 2002.
[42] D. L. Wang, J. Buhmann, and C. M. von der, "Pattern segmentation in associative memory," *Neural Comput.*, vol. 2, pp. 95–107, 1990.
[43] D. L. Wang and D. Terman, "Locally excitatory globally inhibitory oscillator networks," *IEEE Trans. Neural Networks*, vol. 6, no. 1, pp. 283–286, 1995.
[44] ——, "Image segmentation based on oscillatory correlation," *Neural Comput.*, pp. 805–836, 1997.
[45] J. M. Wolfe, "Guided search 2.0: A revised model of visual search," *Psychol. Bull. Rev.*, vol. 1, pp. 202–238, 1994.
[46] S.-C. Yen and L. H. Finkel, "Extraction of perceptually salient contours by striate cortical networks," *Vis. Res.*, vol. 38, pp. 719–741, 1998.

DeLiang Wang (M'90–SM'01) received the B.S. and M.S. degrees from the Peking University, Beijing, China, in 1983 and 1986, respectively, and the Ph.D. degree from the University of Southern California, Los Angeles, in 1991, all in computer science.

From July 1986 to December 1987, he was with the Institute of Computing Technology, Academia Sinica, Beijing. Since 1991, he has been with the Department of Computer and Information Science and the Center for Cognitive Science at The Ohio State University, Columbus, where he is currently a Professor. From October 1998 to September 1999, he was a Visiting Scholar in the Vision Sciences Laboratory at Harvard University, Cambridge, MA. His present research interests include neural networks for perception, neurodynamics, neuroengineering, and computational neuroscience.

Dr. Wang's is a recipient of the 1996 U.S. Office of Naval Research Young Investigator Award.

Xiuwen Liu (S'97–A'99) received the B.Eng. degree in computer science from Tsinghua University, Beijing, China, in 1989, and M.S. degrees in geodetic science and surveying and computer and information science in 1995 and 1996, respectively, and the Ph.D. degree in computer and information science in 1999, all from The Ohio State University, Columbus.

Currently, he is an Assistant Professor in the Department of Computer Science, Florida State University,Tallahassee, leading the Florida State Vision Group. His current research interests include low-dimensional representations of images, image classification and segmentation, statistical computer vision, neural networks, and computational models of vision.