Working Memory Networks for Learning Temporal Order, with Application to 3-D Visual Object Recognition
WORKING MEMORY NETWORKS FOR LEARNING TEMPORAL ORDER, WITH APPLICATION TO 3-D VISUAL OBJECT RECOGNITION

Gary Bradski, Gail A. Carpenter, and Stephen Grossberg

April, 1991

WORKING MEMORY NETWORKS FOR LEARNING TEMPORAL ORDER, WITH APPLICATION TO 3-D VISUAL OBJECT RECOGNITION

Gary Bradski, Gail A. Carpenter, and Stephen Grossberg
Center for Adaptive Systems
and
Department of Cognitive and Neural Systems
Boston University
111 Cummington Street
Boston, MA 02215

Technical Report CAS/CNS-TR-91-014

April, 1991

Requests for reprints should be sent to:
Stephen Grossberg
Center for Adaptive Systems
Boston University
111 Cummington Street
Boston, MA 02215

† Supported by DARPA (AFOSR 90-0083).
‡ Supported in part by British Petroleum (89-A-1204), DARPA (AFOSR 90-0083) and the National Science Foundation (NSF IRI 90-00530).
§ Supported in part by the Air Force Office of Scientific Research (AFOSR 90-128, AFOSR 90-0175), DARPA (AFOSR 90-0083) and the National Science Foundation (NSF IRI 87-16960).

Acknowledgements: The authors wish to thank Kelly Dumont, Diana Meyers, and Carol Yananakis Jefferson for their valuable assistance in the preparation of the manuscript.
ABSTRACT

Working memory neural networks are characterized which encode the invariant temporal order of sequential events. Inputs to the networks, called Sustained Temporal Order REcurrent (STORE) models, may be presented at widely differing speeds, durations, and interstimulus intervals. The STORE temporal order code is designed to enable all emergent groupings of sequential events to be stably learned and remembered in real time, even as new events perturb the system. Such a competence is needed in neural architectures which self-organize learned codes for variable-rate speech perception, sensory-motor planning, or 3-D visual object recognition. Using such a working memory, a self-organizing architecture for invariant 3-D visual object recognition is described. The new model is based on the model of Seibert and Waxman (1990a), which builds a 3-D representation of an object from a temporally ordered sequence of its 2-D aspect graphs. The new model, called an ARTSTORE model, consists of the following cascade of processing modules: Invariant Preprocessor → ART 2 → STORE Model → ART 2 → Outstar Network.
1. INTRODUCTION

Working memory is the type of memory whereby a telephone number, or other novel temporally ordered sequence of events, can be temporarily stored and then performed (Bradeley, 1976). Working memory, a kind of short term memory (STM), can be quickly erased by a distracting event, unlike long term memory (LTM). There is a large experimental literature about working memory, as well as a variety of models (Atkinson and Shiffrin, 1971; Cohen and Grossberg, 1987; Cohen, Grossberg, and Stork, 1987; Elman, 1990; Grossberg, 1970, 1978a, 1978b; Grossberg and Pepe, 1971; Grossberg and Stone, 1986; Gutfriend and Mezard, 1988; Guyon, Personnaz, Nadal, and Dreyfus, 1988; Jordan, 1986; Reeves and Sperling, 1986; Schreter and Pfeifer, 1989; Seibert, 1991; Seibert and Waxman, 1990; Wang and Arbib, 1990). The present article describes working memory models for the storage of temporal order information across a series of event representations.

The present class of models, called STORE (Sustained Temporal Order REcurrent) models, exhibit properties that have heretofore not been available in a dynamically defined working memory. In particular, STORE working memories are designed to encode the invariant temporal order of sequential events that may be presented at widely differing speeds, durations, and interstimulus intervals. Moreover, despite these variations, the temporal order code is designed to enable all possible groupings of sequential events to be stably learned and remembered in real time, even as new events perturb the system. In other words, these working memories enable chunks (also called compressed, categorical, or unitized representations) of variable size to be learned in a manner that is not destabilized by the continuous barrage of new inputs to the working memory. Such chunks are used to represent the most informative combinations of recently stored items, and to predict and control future behavior based upon these representations.

Working memories with these properties are important in many applications wherein properties of behavioral self-organization are needed. Three important applications are real-time self-organization of codes for variable-rate speech perception, sensory-motor planning, and 3-D visual object recognition. Architectures for the first two types of application are described in Cohen, Grossberg and Stork (1987) and Grossberg and Kuperstein (1989). Herein we outline how such a working memory can both simplify and extend the capabilities of the Siebert and Waxman model for 3-D visual object recognition (Seibert and Waxman, 1990; Seibert, 1991).

2. INVARIANCE PRINCIPLE AND PARTIAL NORMALIZATION

The STORE neural network working memories are based upon algebraically characterized working memories that were introduced by Grossberg (1978a, 1978b). These algebraic working memories were designed to explain a variety of challenging psychological data concerning working memory storage and recall. In these models, compressed representations of individual events are stored in working memory in such a way that the pattern of STM activity across event representations encodes both the events that have occurred and the temporal order in which they have occurred. The models also include a mechanism for reading out events in the stored temporal order. An event sequence can hereby be performed from STM even if it is not yet incorporated through learning into LTM.

The large data base on working memory shows that storage and performance of temporal order information from working memory is not always veridical (Atkinson and Shiffrin, 1971;
Baddeley, 1978; Reeves and Sperling, 1986) These deviations from veridical temporal order in STM could be explained by the algebraic working memory model as consequences of two design principles that have clear adaptive value. These principles are called the Invariance Principle and Partial Normalization (Grossberg, 1978b).

**Invariance Principle:** The spatial patterns of STM activation across the event representations of a working memory are stored and reset in response to sequentially presented events in such a way as to leave the temporal order codes of all past event groupings invariant.

In particular, a temporal list of events is encoded in STM in a way that preserves the stability of previously learned LTM codes for familiar sublists of the list.

The Invariance Principle can be algebraically realized as follows, provided that no list items are repeated. Assume that the $i^{th}$ list item activates a compressed recognition code that, in turn, sends an input $I_i$ to a field $F_1$. The activity pattern across $F_1$ represents a working memory with activity $x_i$ at the $i^{th}$ event representation, also called an item representation. At time $t_i$, after the $i^{th}$ input has been presented,

$$x_k(t_i) = \begin{cases} 
0 & \text{if } k > i \\
\mu_i & \text{if } k = i \\
\omega_i x_k(t_{i-1}) & \text{if } k < i 
\end{cases} \quad (1)$$

At time $t_i$, the pattern $(x_1, x_2, \ldots, x_{i-1})$ of already stored STM activities is multiplied by a common factor $\omega_i$ as the $i^{th}$ item is instated with some activity $\mu_i$. The algebraic working memory defined by (1) leaves invariant previously learned codes for sublists of a list as new items are added to the list, as follows. Suppose that sublists are being learned by a competitive learning network (Grossberg, 1976) in which $F_1$ sends signals to a field $F_2$ via an adaptive filter. The total input to the $j^{th} F_2$ node is $\Sigma_k x_k z_{kj}$, where $z_{kj}$ denotes the LTM trace, or adaptive weight, in the path from the $k^{th} F_1$ node to the $j^{th} F_2$ node. In psychological terms, each $F_2$ node represents a chunk, or compressed representation, of the $F_1$ activity pattern. When the $j^{th} F_2$ node is active, the LTM weights $z_{kj}$ converge toward $x_k$; in other words, the weight vector becomes parallel to the $F_1$ activity vector. When a new item is added to the list being learned by node $j$, the STM activity pattern at $F_1$ is altered. The Invariance Principle implies that when the $i^{th}$ item is added, the previously learned weights will simply be multiplied by a common factor. Hence the temporal order of items in the sublist, encoded as relative sizes of both the STM and the LTM variables, remains invariant.

**Partial Normalization:** The Partial Normalization principle algebraically instates the classical property of limited capacity of STM (Atkinson and Shiffrin, 1971). A convenient statement of this property is given by the equation

$$S_i = \sum_k x_k(t_i) = \mu_1 \theta_i + S(1 - \theta_i), \quad (2)$$

where $\theta_1 = 1$ and $\theta_i$ decreases towards 0 as $i$ increases. For example, let $\theta_i = \theta^{i-1}$, with $0 < \theta < 1$. Total activity $S_i$ increases toward an asymptote, $S$, as new items are presented. By (1) and (2), the STM reset parameters $\omega_i$ can be expressed as a function of the parameters
\( \mu_i \) that determine initial storage strength of each activity \( x_i \), and the parameters \( S \) and \( \theta_i \) that determine the growth of total STM activity. Using these properties, it was proved that the rate at which \( S \) approaches asymptote helps determine the nature of the STM activity pattern. In fact, the pattern \( (x_i, \ldots, x_i) \) can exhibit primacy (all \( x_{k-1} > x_k \)), recency (all \( x_{k-1} < x_k \)), or bowing, which combines primacy for early items with recency for later items (Grossberg, 1978a). These various patterns reflect anomalies of STM storage by human subjects.

The multiplicative gating in (1) and the partial normalization in (2) are algebraic versions of the types of properties which are found in a general form in shunting competitive feedback networks (Grossberg, 1973). A task of the present research was to discover specialized shunting nets which realize equations (1) and (2) in detail. The \textit{STORE} model is a real-time shunting network, defined below, which exhibits the desired emergent properties. In particular, the \textit{STORE} system moves from primacy to bowing to recency as a single model parameter is increased.

### 3. WORKING MEMORIES INVARIANT UNDER VARIABLE INPUT SPEED, DURATION, AND INTERSTIMULUS INTERVAL

Two types of real-time working memories, transient models and sustained models, can realize invariance and partial normalization. In a transient model, presentation of items of different durations can alter the previously stored pattern of temporal order information. Transient memory models can still accurately represent temporal order if input durations are controlled by a preprocessing stage. Sustained models allow input durations and interitem intervals to be essentially arbitrary: so long as these intervals are not too short, temporal input fluctuations have no effect on patterns stored in memory. A sustained neural network model is defined below. This is an elementary example of two-level networks called \textit{STORE} (Sustained Temporal Order REcurrent) models. The elementary \textit{STORE} model codes lists of distinct items. A variant of the \textit{STORE} model design, to be discussed in a subsequent article, can encode temporal order of lists in which each item may occur multiple times.

The first level of the elementary \textit{STORE} model (Figure 1a) consists of nodes with STM activity \( x_i \). The \( i \)th item is assumed to send a unit input \( I_i \) to the \( i \)th node for a time interval of length \( \alpha_i \). After an interstimulus interval of length \( \beta_i \), the \((i+1)\)st item sends an input to the \((i+1)\)st node, and so on. Each STM node also receives nonspecific shunting inhibition proportional to the total STM activity. The second \textit{STORE} level consists of excitatory interneurons whose activity \( y_i \) tracks \( x_i \). A critical additional factor in the model is gain control that enables changes in \( x_i \) to occur only when an input is present and enables changes in \( y_i \) to occur only when no input is present.

This alternation allows feedback from \( y_k \) to \( x_k \) (\( k < i \)) to preserve previously stored patterns even when a new input \( I_i \) is on for a long time interval.

**STORE Model Equations**

The elementary \textit{STORE} model is defined by the dimensionless equations

\[
\frac{dx_i}{dt} = [AI_i + y_i - x_i] I
\]
Figure 1. (a) Elementary STORE model: STM activity $x_i$ at level 1 registers the item input $I_i$, nonspecific shunting inhibition $z$, and level 2 STM $y_i$. STM activity $y_i$ at level 2 registers $x_i$. Complementary input-driven gain signals $I$ and $I^c$ control STM processing at levels 1 and 2. (b) Input $I_i(t)$ equals 1 for $t_i - \alpha_i < t \leq t_i$. When all inputs are off ($t_i < t \leq t_i + \beta_i$) level 2 variables $y_k$ relax to level 1 values $x_k(t_i)$.

$$\frac{dy_i}{dt} = [x_i - y_i]I^c,$$

where

$$x = \sum_k x_k,$$

$$I = \sum_k I_k,$$

$$I^c \equiv 1 - I,$$

and

$$x_i(0) = y_i(0) = 0.$$

The input sequence $I_i$ is given by

$$I_i(t) = \begin{cases} 1 & \text{if } t_i - \alpha_i < t < t_i \\ 0 & \text{otherwise.} \end{cases}$$

(Figure 1b). The input durations ($\alpha_i$) and the interstimulus intervals ($\beta_i = t_i - \alpha_i - t_{i-1}$) are assumed to be large relative to the dimensionless relaxation times of $x_i$ and $y_i$, set equal to 1 in (4) and (5). Thus each $x_i$ reaches steady state when inputs are on and each $y_i$ reaches
when inputs are off. Otherwise, \( t_i \) and \( \alpha_i \) can be arbitrary, and their values have no effect on patterns of memory storage.

**Temporal Order Patterns**

We will now examine how system properties vary as a function of the single free parameter, \( A_1 \) in (3). We will see that, in all cases, patterns of past activities remain invariant as new inputs perturb the system, and partial normalization obtains. In addition, the STM pattern \((x_1 \ldots x_i)\) exhibits primacy for small \( A \), recency for \( A > 1 \), and bowing for intermediate values of \( A \), as follows.

By (3), (8), and (9), when the \( i \)th input is presented, \( I_i = 1 \), \( y_i = 0 \), and

\[
x_i \rightarrow \frac{A}{x}.
\]  

For \( k < i \), \( I_k = 0 \) and

\[
x_k \rightarrow \frac{y_k}{x} = \frac{x_k(t_{i-1})}{x}.
\]  

Thus the pattern \((x_1 \ldots x_{i-1})\) is preserved when \( x_i \) becomes active. Amplitudes increase uniformly if total activity \( x < 1 \), and decrease uniformly if \( x > 1 \). Equations (3) and (4) imply that the variable \( x \) obeys the equations.

\[
\frac{dx}{dt} = [A + y - x^2] I
\]  

(12)

\[
\frac{dy}{dt} = [x - y] I^c,
\]  

(13)

where

\[
y = \sum_k y_k.
\]  

(14)

By (12), \( S_i \equiv x(t_i) = \sqrt{A} \) and \( S_i \equiv x(t_i) = \sqrt{A + S_{i-1}} \). As the number of items increases, \( y(t) \approx x(t) \) and \( x(t) \) approaches the limit \( S = .5[1 + \sqrt{1 + 4A}] \). For large \( A \), therefore, \( \frac{S_i}{S} \approx 1 \); for small \( A \), \( \frac{S_i}{S} \approx \sqrt{A} \ll 1 \). Thus, for large \( A \), STM is approximately normalized at all times. Since the size of parameter \( A \) in (3) reflects the degree to which the input \( I_i \) influences the STM pattern, recency for large \( A \) (present input dominates) and primacy for small \( A \) (past activities dominate) would be predicted. In fact, for large \( A \), the pattern of STM activity \((x_1, \ldots, x_i)\) always shows a recency gradient (Figure 2). For small \( A \), total STM activity \( S_i \) grows sharply at first, and initial STM patterns show a primacy gradient. Specifically, by (10) and (11),

\[
x_i(t_i) = \frac{A}{x(t_i)}
\]  

(15)

and
Figure 2. STORE model simulations for decreasing values of the input parameter $A$. The STM patterns $(x_1(t_1), \ldots, x_i(t_i))$ show recency for large $A$, bowing for intermediate values of $A$, and primacy for small values of $A$. Total activity $x(t_i) = S_i$ grows toward the asymptote $S$ as $i$ increases. When a new input $I_i$ is stored, the previous pattern vector $(x_1, \ldots, x_{i-1})$ is amplified if $S_i \equiv S_i > 1$; or depressed if $S_i \equiv S_i < 1$; but the pattern of relative activities is preserved. For these simulations, input durations $\alpha_i$ were varied randomly between 10 and 40, with the intervals $(t_i - t_{i-1})$ set equal to 50.

\[
x_{i-1}(t_i) = \frac{x_{i-1}(t_{i-1})}{x(t_i)}.
\]

Thus at time $t_i$, just after $I_i$ has been presented,

\[
x_{i-1} > x_i \text{ iff } x_{i-1}(t_{i-1}) > A.
\]

Thus if $x_1(t_1) > A$, $(x_1, \ldots, x_i)$ will show a primacy gradient until $x_i(t_i) \leq A$. At that point, the STM pattern will bow. If $x_1(t_1) < A$, the STM pattern will always exhibit recency. Since $x_1(t_1) = \sqrt{A}$, recency occurs whenever $A \geq 1$, while small $A$ values lead to long primacy intervals. Exact bow position can be calculated iteratively. For example, the bow will occur at position $i = 2$ if $1 > A \geq .5(3 - \sqrt{5}) \approx 0.382$.

4. A SELF-ORGANIZING ARCHITECTURE FOR INVARIANT 3-D VISUAL OBJECT RECOGNITION

Seibert and Waxman (1990a, 1990b; Seibert, 1991) have developed a novel self-organizing neural network architecture for invariant 3-D visual object recognition. In response to moving objects in space, an Invariant Preprocessor in the architecture automatically generate 2-D patterns that are invariant under changes in object position, size, and orientation. These patterns form the inputs to an ART 2 network (Carpenter and Grossberg, 1987) that self-organizes learned category representations of the invariant patterns. Each category encodes a 2-D “aspect” of the object; that is, it forms a single compressed representation of a collection of similar 2-D views of the object. The ART 2 vigilance parameter controls how similar these 2-D views must be in order to activate the same category node $J$. 
Figure 3. The Aspect Network of Seibert and Waxman detects temporal order properties by computing the temporal overlap of pairs \( x_i \) and \( x_j \) of activities at distinct locations \((i, j)\) and then learning the pattern of overlapping traces to code transitions between 2-D aspects. (a) A single-object Aspect Network. (b) A complete multi-object Aspect Network in which each 2-D Aspect Node fans out to contact the Aspect Networks corresponding to all 3-D Object Nodes, which compete among themselves according to winner-take-all competitive learning rules. Reprinted with permission (Seibert, 1991).

Seibert and Waxman have successfully applied their system to the recognition of real 3-D objects. As the object moves with respect to the camera, a temporally ordered sequence \( J_1, J_2, \ldots, J_m \) of category nodes is activated. These nodes and their transitions implicitly represent invariant 3-D properties of the object, in much the same manner as an "aspect graph" (Koenderink and van Doorn, 1979). Seibert and Waxman developed a second-order network, called an Aspect Network, to represent the transitions between 2-D aspects (Figure 3). The output of the second-order network forms the input to a competitive learning network, each of whose 3-D Object Nodes learns to fire only when an unambiguous sequence of 2-D aspects is activated. An important feature of this model is its ability to recognize novel sequences composed of learned transitions and to move from one sequence to another sequence that intersects it. This approach to synthesizing 3-D recognition from combinations of distinct 2-D views is consistent with data of Perrett et al. (1987) about cells in temporal cortex that are sensitive to different 2-D views of a face.

Despite its many appealing features, the Seibert and Waxman model could face two types of limitations if used in a more general context: proliferation of connections and sensitivity to input timing. As in all higher order networks proliferation of connections could occur in the Aspect Graph, although this problem does not occur in the application domain considered by Siebert and Waxman in general, each possible temporal order could use a different Aspect Network to compute products of the temporally overlapping STM traces of all successive input pairs (Figure 3a). The spatial loci of these intersecting quantities project through an adaptive filter to the corresponding 3-D Object Node. In order to compute all possible objects that can be represented by \( M \) distinct (and non-repeated) 2-D Aspect Nodes \( N_i \), one needs to represent \( M! \) temporal orderings by \( M! \) Aspect Networks (Figure 3b). Each Aspect
Figure 4. Processing stages of an ARTSTORE model for invariant 3-D object recognition.

Network computes $O(M^2)$ products which requires $O(M^2)$ weighted pathways to each 3-D Object Node. In our modified architecture, called an ARTSTORE model,

only one STORE model is needed with $M$ nodes to represent all $M!$ temporal orders, no Aspect Network is needed, and only $O(M)$ weighted pathways are needed to each 3-D Object Node (Figure 4). The reduction in system complexity is due to the use of analog STM and LTM quantities to represent order, as opposed to binary representations, which require second order connections.

The Seibert-Waxman scheme for computing aspect transitions uses, by using products of successive STM traces, is also sensitive to changes in input speed, duration, and interstimulus interval. They partially overcome this problem by using a specialized LTM law whose adaptive weight converges to 1 if the corresponding product exceeds a threshold, and zero otherwise (Figure 3a). Such an approach cannot deal with long interstimulus intervals $\beta_i$. This problem also does not arise if a sustained working memory, such as a STORE model, is used instead of the Aspect Networks. Each node of the STORE model is a 2-D Aspect Node. Different analog patterns of activation across the STORE model encode different invariant temporal orderings of the object’s 2-D aspects. This temporal order code of a STORE model can input to an ART 2 network, which can automatically learn to select different 3-D Object Nodes in response to different analog patterns of temporal order information over the same fixed set of working memory nodes. The STORE model learns transitions if network dynamics are adjusted to cause all sequences stored in STM to have length 2.

We also augment the Seibert-Waxman scheme by letting 3-D Object Nodes be used to learn arbitrary output names via outstar learning (Grossberg, 1968, 1978b). Each 3-D Object Node is the source cell of an outstar. All the outstars converge on the same outstar border where an output name can be represented in an arbitrary format by an external teacher. Thus, in response to a 3-D object moving with respect to the Invariant Preprocessor, the architecture outputs an object name when enough information about the object’s 2-D aspects and their temporal order have accumulated. The total self-organizing
system uses the following cascade of processing stages: Invariant Preprocessor → ART 2 (2-D Aspects) → STORE model (Invariant Working Memory) → ART 2 (3-D Objects) → (Outstar Network). This is a self-organizing multi-level instar-outstar map specialized for invariant 3-D object recognition (Carpenter and Grossberg, 1991).

5. CONTROL OF WORKING MEMORY AND TEMPORAL LEARNING.

Reset of the working memory can be autonomously controlled by the object tracking system that Seibert and Waxman have incorporated into their Invariant Preprocessor. This system enables the architecture's camera to continuously track a moving object. As continuous tracking occurs, a sequence of 2-D aspects is learned and encoded in working memory, after which a ballistic camera movement focuses on a new object. We assume that working memory is reset, and thereby cleared, when a ballistic movement occurs; for example, by reducing the gain of the recurrent interactions between $y_i$ and $x_i$ in the STORE model. As a result, each sequence of simultaneously stored 2-D aspects represents the same 3-D object with high likelihood.

ART 2 learning of each working memory pattern may be controlled in either of two ways: (1) Unsupervised learning: Here each new entry into working memory causes ART 2 to choose and learn a new category. Each subsequence $(J_1), (J_1, J_2), (J_1, J_2, J_3), \ldots$ of 2-D aspect nodes can then learn to activate its own ART 2 node. Only those subsequences which are associated with names of 3-D objects generate output predictions. (2) Supervised learning: Here an ART 2 learning gate is opened only when a teaching input to an outstar occurs. Consequently, only those sequences $(J_1, J_2, \ldots)$ that generate 3-D object predictions will learn to activate ART 2 categories and their outstar predictions. The number of learned ART 2 categories is hereby minimized. In either case, the ART 2 module can learn to select those combinations of item and order information that are predictive of an object by using its top-down expectation and vigilance properties (Carpenter and Grossberg, 1987).

6. CONCLUDING REMARKS

The present model illustrates how a hierarchically-organized neural architecture can self-organize a higher-order type of invariant recognition by cascading together a combination of self-organizing modules, each of which computes a simpler invariant property. The Invariant Preprocessor computes a position/size/rotation invariant; the first ART 2 computes a self-calibrating similarity invariant of 2-D aspects; the STORE model computes a temporal order invariant; and the second ART 2 computes a self-calibrating similarity invariant of 3-D objects. In particular, the self-calibrating similarity invariant of 2-D aspects needs the temporal invariance of working memory to gain full effectiveness. This is so because the timing of individual outputs from the 2-D aspect nodes can depend in a complex way on the 3-D shape of an object and its relative motion with respect to the camera or other observer.
REFERENCES


