A Spatiotemporal Memory that Separates "What" and "When"

Wee Kheng Leow
School of Computing
National University of Singapore
Lower Kent Ridge Road, Singapore 119260
email: leowwk@iscs.nus.edu.sg

Abstract

To process a spatiotemporal pattern, a short-term memory (STM) is needed to maintain the pattern over time. In general, a STM for storing a temporal sequence of input items should possess characteristics such as acting as a first-in-first-out buffer, storing multiple items of the same type, and encoding interval-invariant and rate-invariant representations of the input items. Some of these characteristics are lacking in existing STM models. This article presents a spatiotemporal STM that possesses all the above characteristics. It is achieved using a novel method of organizing the STM by separating the storage of spatial and temporal information.

1 Introduction

Many intelligent behaviors such as hearing, speech, reading, and vision involve the processing of spatiotemporal patterns. To process a pattern that changes over time, a short-term memory (STM) is needed to maintain the changing pattern for some specific amount of time. Some existing models of STM store raw inputs sampled at fixed time intervals, whereas others store specific items such as the letters '*b*', '*e*', '*e*', and '*t*' in the word *beet*. This article focuses on STM of the latter type.

A spatiotemporal STM for storing items should have the following characteristics:

1. The STM should act as a first-in-first-out buffer.
2. It can store a list of items, including multiple items of the same type, for example, the two '*e*'s in the word *beet*.
3. It should preserve the items' temporal order regardless of their durations (*intra-stimulus durations*) and the time intervals between successive items (*inter-stimulus intervals*). This characteristic is know as *interval invariance* [6], and is useful, for instance, for memorizing a 7-digit telephone number regardless how fast or slowly are the digits spoken.
4. The stored patterns should be *rate-invariant* [6] which means that they are invariant to a scaling of the intra-stimulus durations and inter-stimulus intervals. This property is useful, for example, for distinguishing between *beet* and *bit* since the phoneme of '*ee*' is longer than that of '*i*' in comparison to the whole word. The distinguishing criterion is dependent on the relative duration and interval rather than the absolute ones.

2 Related Works

Existing models of STM can be grouped into two basic categories: tapped delay line and reduced trace (refer to [6] for a more comprehensive review). A tapped delay line works like a shift register [3, 4]. An input pattern passing through a delay line produces an output at each successive delay tap. It transforms a spatiotemporal pattern into a set of spatial patterns whose temporal order is given by the sequence of the taps in the delay line. It is, thus, a first-in-first-out buffer which can store as many items as there are taps in the delay line. However, since the time delays are usually fixed, a tapped delay line is neither interval-invariant nor rate-invariant. A special kind of tapped delay line with variable propagation speed has been proposed by Yamauchi et al. [7]. It uses additional complex networks to examine the input over fixed time intervals to determine the appropriate propagation speed of the delay lines. With the additional networks, the patterns stored in the tapped delay lines are rate-invariant, but not interval-invariant.

The reduced trace models represent an item type by a single distinct unit and encode temporal order in the units' activities [5, 1]: the smaller the activity, the older is the item in the memory. To store more than one, say, N items of each type, N duplicate units are

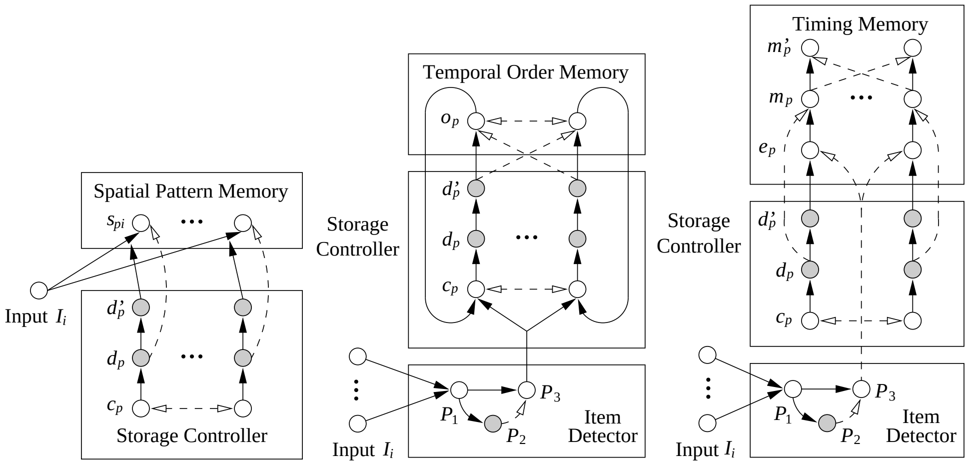


Figure 1: SSSTEM consists of five main modules: Item Detector, Storage Controller, Spatial Pattern Memory, Temporal Order Memory, and Timing Memory. Shaded units introduce delay in the propagation of unit activity. Solid lines denote excitatory connections and dashed lines denote inhibitory connections.

required for each item type. These STM models can store at most N items of the same type even when the memory units of the other types are completely empty. Therefore, their' usage of memory units is far from optimal. These models are typically interval-invariant because the storage of an item is triggered by the detection of the item rather than triggered at fixed time intervals. However, since they do not encode inter-stimulus intervals, the patterns stored in these models are not rate-invariant.

This article presents a model of spatiotemporal STM called SSSTEM (Separate Storage SpatioTEmporal Memory) that combines the strength of the delay line and the reduced trace models. SSSTEM is a first-in-first-out buffer and it can store as many items, whether of the same or different types, as there are memory slots in the network. It can also encode interval- and rate-invariant representations of the input patterns. SSSTEM achieves these desired characteristics by separating the storage for spatial and temporal information.

3 Network Architecture and Operation

SSSTEM consists of five main modules (Fig. 1): Item Detector, Storage Controller, Spatial Pattern Memory, Temporal Order Memory, and Timing Memory. The Spatial Pattern Memory consists of a matrix of units s_{pi}, each row of which is used to store a spatial pattern $\mathbf{s}_p$. The Temporal Order Memory unit o_p stores the temporal order of spatial pattern $\mathbf{s}_p$: the larger the o_p, the more recent is the corresponding pattern $\mathbf{s}_p$. The Timing Memory unit m_p records the time interval between pattern $\mathbf{s}_p$ and the next pattern. The Item Detector and the Storage Controller coordinate the storage of spatial patterns and the corresponding temporal order and time intervals. Let us look at how SSSTEM operates.

3.1 Storing Spatiotemporal Patterns

The Item Detector is a network that examines the inputs to detect the arrival of spatiotemporal patterns. In general applications, it can be quite a complex network. In this article, a simple item detector is illustrated as an example.

SSSTEM's Item Detector consists of three units: P_1, P_2, and P_3. Unit P_1 examines the inputs I_i and produces a large output when any of the inputs I_i is larger than the positive threshold Γ_P:

$$\dot{P}_1 = -\alpha_P P_1 + (\Psi_P - P_1)\beta_P \max_i f(I_i - \Gamma_P)\,. \quad (1)$$

The activation function f is a sigmoid-like function:

$$f(x) = \begin{cases} 1 & \text{if } x > 1 \\ x & \text{if } 0 \le x \le 1 \\ 0 & \text{otherwise.} \end{cases} \quad (2)$$

The parameter α_P determines the rate of decay, β_P the rate of increase, and Ψ_P the maximum value of P_1. Unit P_2 is a delay unit which feeds a delayed copy of P_1 to P_3. By summing P_1 and P_2, unit P_3 generates an output *event pulse* wherever an input item is detected (Fig. 2a):

$$P_3(t) = P_1(t) - P_2(t) = P_1(t) - P_1(t - \tau_P) \quad (3)$$

where τ_P is a positive time delay. Together, the parameters α_P, β_P, Ψ_P, and τ_P determine the event pulse's height and width (i.e., duration). The pulse height is arbitrary but should be large enough. The pulse width must be much smaller than the duration of the input pattern so that SSSTEM can complete the storage of the pattern before it vanishes. In the current implementation, the input values I_i are set to range from 0 to 1, and each input pattern is set to last for 4 time units. The values of the above parameters (Appendix) are then chosen to produce event pulses with a height of 0.8 and a width of 0.5 time steps. The P_3 pulse is propagated to the Storage Controller to coordinate the storage of input patterns.

In the Storage Controller (Fig. 1), a loser-take-all network c_p determines which memory slot p to use to store the input pattern. It looks at the Temporal Order Memory for a unit o_p with the smallest activation value:

$$c_p = \begin{cases} P_3 & \text{if } o_p = \min\limits_q o_q \\ 0 & \text{otherwise.} \end{cases} \quad (4)$$

The least active unit, say, o_k represents the oldest memory slot k which will be used to store the new input pattern. The P_3 pulse is propagated along the single selected pathway k to the delay units d_k and d'_k (Fig. 2a):

$$d_p(t) = c_p(t - \tau) d'_p(t) = d_p(t - \tau)\,. \quad (5)$$

The delay units introduce further delays of duration τ to the P_3 pulse to coordinate the clearing and storing of memory. In particular, the d_k pulse along the selected pathway k clears memory units s_{ki} to prepare them for storing a new pattern, and the d'_k pulse stores the input pattern I_i into units s_{ki} (Fig. 2a):

$$\dot{s}_{pi} = -\lambda d_p + \beta_s d'_p I_i\,. \quad (6)$$

The parameter λ has a large positive value which ensures fast clearing of s_{pi} to 0. The parameter β_s determines the storage speed of units s_{pi}. Appropriate choice of β_s (Appendix) ensures that the stored value is equal to the input value (Fig. 2b).

3.2 Storing Temporal Order

As input patterns are stored in the Spatial Pattern Memory, the Temporal Order Memory units o_p (Fig. 1) are updated to encode the patterns' temporal order:

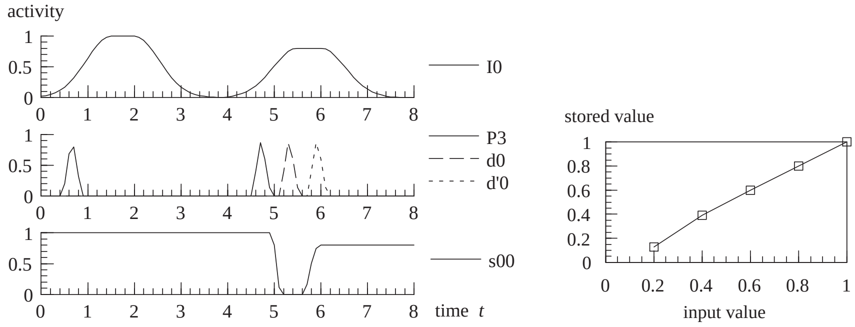
$$\dot{o}_p = -o_p \sum_q (o_q + \beta_o d'_q) + o_p + \beta_o d'_p \quad (7)$$

where β_o is a positive parameter that determines the rate of increase of o_p. When an input pattern is stored into, say, memory slot k, the Temporal Order Memory unit o_k will integrate its activity for the duration of the d'_k pulse. It has been shown in Leow [2] that Eq. 7 always normalizes the total network activity $\sum_p o_p$ to the value 1. As a result, as o_k increases, the activities of other o_p units decrease. The parameter β_o is set to an appropriate value (Appendix) such that at the end of the integration period, o_k will have the largest value, indicating that the most recent input pattern is now stored into Spatial Pattern Memory units s_{ki}.

Table 1 illustrates an example of Temporal Order Memory's operation. Initially, the units o_p are initialized to the proportional values $p/\sum_p 1$ to encode the initial sequence (Table 1, row 0). Unit o_0 has the smallest value which represents the oldest memory slot, and unit o_4 has the largest value which encodes the most recent slot. When pattern 1 arrives, it is stored into memory slot 0. Unit o_0 is now updated to represent the most recent memory slot. Due to total activity normalization, unit o_1 now has the smallest value, indicating that it now represents the oldest memory slot. As new patterns are stored into the memory, units o_p are updated to encode the new temporal order. Successive input patterns are stored in successive memory slots regardless of whether they

Table 1: Operation of Temporal Order Memory. The memory units o_p are initialized to the values given in row 0. The unit with the largest value represents the most recent memory slot. As new patterns are stored, memory units o_p are updated to encode the new temporal order.

S/No.	o_0	o_1	o_2	o_3	o_4
0	0.067	0.133	0.200	0.267	**0.333**
1	**0.323**	0.095	0.144	0.194	0.243
2	0.236	**0.343**	0.104	0.140	0.177
3	0.172	0.251	**0.348**	0.101	0.128
4	0.125	0.183	0.255	**0.345**	0.093
5	0.090	0.133	0.186	0.254	**0.339**
6	**0.339**	0.095	0.134	0.184	0.247
7	0.249	**0.342**	0.096	0.133	0.180



(a) (b)

Figure 2: (a) At the arrival of an input pattern (only I_0 is shown here), SSSTEM generates three pulses P_3, d_p, and d'_p in sequence to coordinate the storage of the pattern. In this example, at time $t = 5$, unit s_{00} is selected for storing the new input I_0. The d_0 pulse clears the activity of s_{00} and the d'_0 pulse stores the new input into s_{00}. (b) By choosing appropriate values for network parameters, the value stored in the memory can be made equal to the input value (except when the input value is very close to the input threshold, in this example, 0.1).

are of the same or different types. Therefore, SSSTEM can store as many items of the same type as the number of memory slots, and it uses memory units more efficiently than do existing reduce trace models.

3.3 Counting Time

SSSTEM records time intervals in the Timing Memory, which consists of three sets of units (Fig. 1). The timing units m_p integrate their activities over time thereby keepint track of time intervals. The activator units e_p keep the timing units m_p integrating over the time intervals between two successive input patterns. The output units m'_p normalizes the activities of m_p to encode rate-invariant representations. Let us examine each of these units more closely.

After a memory slot k is selected in the Storage Controller, d_k pulse arrives at the Timing Memory to clear unit m_k to prepare it for recording a new time count:

$$\dot{m}_p = -\lambda d_p + \beta_m e_p \, . \tag{8}$$

Subsequently, the arrival of the d'_k pulse sets unit e_k to a positive value which starts the timing unit m_k integrating (Fig. 3a):

$$\dot{e}_p = -\lambda P_3 + \beta_e d'_p \, . \tag{9}$$

When the Item Detector detects the arrival of a new input pattern, it generates a P_3 pulse which clears all the activator units e_p and stops the time counting (Eqs. 8, 9, Fig. 3a). Therefore, timing units m_p can keep track of the time intervals between the arrival of successive input patterns.

The product of parameter β_m (Eq. 8) and the activity e_p determine the rate of increase of m_p. In the current implementation, β_e (Eq. 9) is chosen such that unit e_p maintains a sustained output of 1 when it is activated. The value of β_m is arbitrarily chosen as 0.1. Thus, m_p increases at a rate of 0.1 per unit time (Fig. 3b).

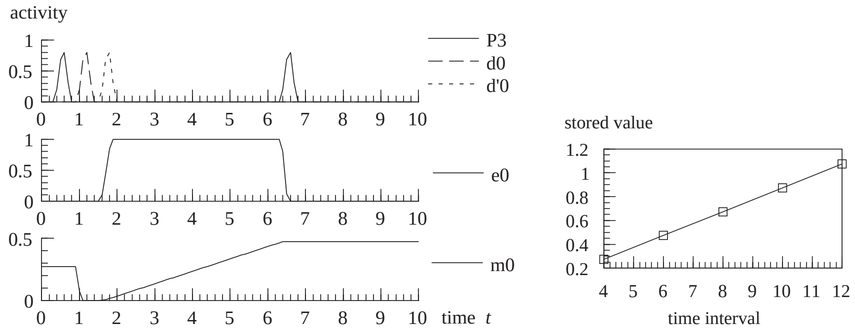
At the final stage of the Timing Memory, the stored values of units m_p are normalized by units m'_p (Fig. 1):

$$\dot{m}'_p = -m'_p \sum_q m_q + m_p \, . \tag{10}$$

It has been proved in Leow [2] that Eq. 10 always normalizes the total activity $\sum_p m'_p$ to 1 and the individual activities m'_p to $m_p / \sum_q m_q$. In other words, output m'_p depends only on the relative amounts of the time intervals instead of the absolute amounts. Therefore, the temporal information recorded in m'_p and the spatial pattern s_{pi} together encode a rate-invariant representation of the input items. As new items are stored into the memory replacing older ones, the timing units m_p and m'_p are continuously updated to encode the normalized time intervals between successive items stored in the memory.

4 Conclusions

This paper has presented a model of spatiotemporal memory called SSSTEM that stores spatial and temporal information in separate networks. This approach makes it easy for the model to achieve the desired characteristics of spatiotemporal memory listed in Section 1. In particular, SSSTEM acts as a first-in-first-out buffer, and it can store as many items as there



(a) (b)

Figure 3: (a) Operation of the Timing Memory. The P_3 pulse stops the time counting of all Timing Memory units. The d_0 pulse clears timing unit m_0 to 0 to prepare it for time counting. The d'_0 pulse sets unit e_0 to 1 which, in turn, starts the timing unit m_0 integrating until the arrival of the next P_3 pulse. (b) By choosing appropriate values for Timing Memory's parameters, the stored values can be made proportional to the time intervals between successive input patterns.

are memory slots regardless of whether the items are of the same or different types. It, thus, uses the memory units as efficiently as do tapped delay lines, and more efficiently than do reduce trace models. SSSTEM can also encode both interval-invariant and rate-invariant representations of the input items. Interval invariance is achieved, like reduced trace models, by detecting the arrival of input items and encoding the temporal order of stored items. Rate invariance is achieved by recording the normalized time intervals between successive items stored in the memory.

A Network Parameter Values

All the test examples illustrated in this article are obtained with the following network parameter values:

- For the Item Detector, $\alpha_P = \beta_P = \Psi_P = 10$, $\Gamma_P = 0.1$, and $\tau_P = 0.2$ time units.
- For the Storage Controller, $\tau = 0.6$ time units.
- For the Spatial Pattern Memory, $\beta_s = 5$.
- For the Temporal Order Memory, $\beta_o = 1.5$.
- For the Timing Memory, $\beta_e = 5$ and $\beta_m = 0.1$.
- For all three memory networks, $\lambda = 10$.

Acknowledgements

This research work is supported by NUS Academic Fund RP950656.

References

References

[1] G. Bradski, G. A. Carpenter, and S. Grossberg. STORE working memory networks for storage and recall of arbitrary temporal sequences. *Biological Cybernetics*, 71:469–480, 1994.

[2] Wee Kheng Leow. *VISOR: Learning Visual Schemas in Neural Networks for Object Recognition and Scene Analysis.* PhD thesis, The University of Texas at Austin, Austin, Texas, 1994.

[3] J. L. McClelland and J. L. Elman. Interactive processes in speech perception: The TRACE model. In J. L. McClelland and D. E. Rumelhart, editors, *Parallel Distributed Processing*, volume 2. MIT Press, Cambridge, Massachusetts, 1986.

[4] A. Waibel, T. Hanazawa, G. Hinton, K. Shikano, and K. Lang. Phoneme recognition using time-delay neural networks. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37:328–339, 1989.

[5] D. L. Wang and M. A. Arbib. Complex temporal sequence learning based on short-term memory. *Proceedings of IEEE*, 78:1536–1543, 1990.

[6] DeLiang Wang. Temporal pattern processing. In Michael A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks.* MIT Press, Cambridge, Massachusetts, 1995.

[7] K. Yamauchi, M. Fukuda, and K. Fukushima. Speed invariant speech recognition using variable velocity delay lines. *Neural Networks*, 8(2):167–177, 1995.