Limit Cycle Representation of Spatial Locations Using Self-Organizing Maps

Di-Wei Huang
Department of Computer Science
University of Maryland
College Park, MD 20742
dwh@cs.umd.edu

Rodolphe J. Gentili
Department of Kinesiology
University of Maryland
College Park, MD 20742
rodolphe@umd.edu

James A. Reggia
Department of Computer Science
University of Maryland
College Park, MD 20742
reggia@cs.umd.edu

***Abstract*—We use the term "neurocognitive architecture" here to refer to any artificially intelligent agent where cognitive functions are implemented using brain-inspired neurocomputational methods. Creating and studying neurocognitive architectures is a very active and increasing focus of research efforts. We have recently been exploring the use of neural activity limit cycles as representations of perceived external information in self-organizing maps (SOMs). Specifically, we have been examining limit cycle representations in terms of their compatibility with self-organizing map formation and as working memory encodings for cognitively-relevant stimuli (e.g., for images of objects and their corresponding names expressed as phoneme sequences [1]). Here we evaluate the use of limit cycle representations in a new context of relevance to any cognitive agent: representing a spatial location. We find that, following repeated exposure to external 2D coordinate input values, robust limit cycles occur in a network's map region, the limit cycles representing nearby locations in external space are close to one another in activity state space, and the limit cycles representing widely separated external locations are very different from one another. Further, and in spite of the continually varying activity patterns in the network (instead of the fixed activity patterns used in most SOM work), map formation based on the learned limit cycles still occurs. We believe that these results, along with those in our earlier work, make limit cycle representations potentially useful for encoding information in the working memory of neurocognitive architectures.**

I. Introduction

Interest in developing intelligent agents based on neurocognitive architectures has been accelerating during recent years. By *neurocognitive architectures* (also called biologically-inspired cognitive agents), we mean artificially intelligent systems where cognitive functions are implemented using brain-inspired neural networks. Often such systems try to create neuro-anatomically grounded simulations of all or major portions of human/mammalian brain structure and function, or at least major subsystems of the brain that span multiple cortical regions. Thus relevant recent work in this area has involved models that vary from extremely large networks of biologically-realistic spiking neurons to those that are more abstract, based on a higher level of components such as cortical columns, or are focused on simultaneously supporting cognitive functions [2]–[6]. Such work is likely to increase rapidly during the coming decade, in part due to major recent research funding initiatives (the Human Brain Project in Europe, the BRAIN Initiative in the US, etc. [7]).

Self-organizing maps (SOMs) have received relatively little attention so far in large-scale neurocognitive architectures, something that is surprising given the tremendous interest in neuroscience in "mapping" the brain. Self-organizing maps are artificial neural networks initially inspired by maps in biological neural systems [8], [9]. They learn, using unsupervised methods, to project vectors in a high-dimensional input space onto a low-dimensional (usually 2D) lattice of artificial neurons. This projection preserves topological relationships of the input space — nearby neurons in a trained map are typically sensitive to similar input patterns, although sudden jumps may also occur. SOMs are important neuocomputational tools in part due to their frequent use for clustering and visualizing complex high-dimensional data that may otherwise be difficult to understand [9]–[11]. They have been applied in a wide range of fields, such as robotic control, weather monitoring, genome analysis, and economic modeling, to name just a few examples. At the same time, SOMs are also important as brain models. Sensorimotor maps are a common phenomenon in the brain, most famously in the neocortex, and SOMs have been successful in modeling many biologically observed cortical properties, such as self-organization of somatosensory and visual cortical regions [12], [13], the alignment of multiple feature maps [14], the formation of mirror-symmetric topographic maps [15], and related cognitive phenomena [16].

A fundamental barrier to adopting SOMs in large-scale neurocognitive architectures, and one that is central to the work reported below, is that most past SOMs have used a *static representation* of information. By this we mean that each input pattern or sequence of patterns is typically represented by a single fixed activation pattern over the map layer, without considering how that activation pattern evolves with time. In conventional SOMs, activation patterns are driven directly by fixed input stimuli, and thus remain static until external control mechanisms alter or reset the map layer. This remains true even in SOMs processing temporal sequences: even though the map regions have changing activation patterns over time, the representation that encodes a whole input sequence is still a single spatial activity pattern that is chosen from the series of changing activation patterns (e.g., the activation pattern that occurs in response to the last element of an input sequence [17]). A sequence's representation is therefore still static. Importantly, such a static representation is typically transient — it occurs in a SOM for only a short time in-between updates of the SOM's activation. For a downstream neural component to access a transient representation like this, it has to either implement a precise timing control or artificially

"freeze" the SOM's activation state. In both cases, a priori knowledge about the timing of the transient representation is required.

In the context of building large-scale neurocognitive architectures, using static representations for SOMs can also introduce biological implausibility. For example, biological neural systems generally do not exist in static activation states, and they are unlikely to operate solely based on fixed spatial patterns. On the contrary, biological systems are clearly characterized by prominent ongoing rhythmic oscillatory activity [18], [19], and there is substantial evidence that cognitive functions such as memory are strongly related to this rhythmic oscillation of neural activities [20]. Computational models for non-fixed point neurodynamics has been intensively studied in several types of neural networks [21]–[25], including attractor networks, oscillator networks, reservoir networks, and so on. However, the oscillatory nature of this ongoing activity has rarely been accounted for in past work on SOMs, although there are exceptions. In [26], a SOM with explicit phase variables for each node is used to detect the periods of oscillatory input stimuli, while in [27], a conventional SOM is based on spiking neurons that exhibit periodic firing behaviors. However, the oscillatory activation in both of these studies is a direct result of the input stimuli being periodic, not of the map layer's activation dynamics. It is unclear if they would exhibit any oscillatory activation at all should the input become aperiodic or unavailable, and the issue of whether oscillations would continue or what they would represent if they did occur is not considered.

Motivated by the above issues, here we investigate a new *dynamic representation* for use in SOMs that involves not only spatial patterns but also temporal extension in the form of map limit cycle attractors. To our knowledge, our approach is the first attempt to encode external stimuli using activity limit cycles in SOMs. As a first step, our very recent work [1] studied this issue in situations where each input pattern/sequence (phonemes and images) is in the form of a binary-valued activation pattern. In this initial experiment, we found that short variable-length limit cycle attractors emerged in the state space of a multi-winner SOM. The limit cycles obtained were self-sustained and persisted after the input that triggered them terminated, and they can therefore be viewed as a candidate representation for working memory. We also showed that the resulting limit cycles possess properties that are appealing for representations, and that map formation, a defining phenomenon shared by most variants of SOMs, occurs despite using a limit cycle dynamics.

However, the generality of these initial results is unclear. For example, it is unclear how real-valued inputs, as opposed to binary inputs reported in our previous work, affect the dynamics of a SOM, or whether a map will even form under these conditions. Additionally, it is unknown how the SOM will respond to new inputs that have never been seen during training, since our previous work intentionally used the same data set for both training SOMs and evaluating resulting limit cycles. In this work, we study for the first time the limit cycle dynamics of a SOM using real-valued inputs, which are spatial coordinates of a 2D surface, and using separate training and evaluation data. This specific application (location in 2D space) is motivated in part by its obvious relevance to cognitive agents. The research question being asked is whether limit cycles can be obtained at all, as well as whether map formation will occur in the trained SOM, under these new conditions. If so, our goal is to evaluate the nature of the emerged limit cycles and their suitability to serve as internal representations for real-valued stimuli. In particular, using real-valued spatial locations as inputs allows us to look closely at how locations are organized in their representation space, which is composed of spatiotemporal limit cycle states. A central issue here is whether nearby locations are represented using similar limit cycles. This was not discussed in our previous work using binary features [1] partly because the distances between binary features are less continuous than real-valued inputs used here.

II. Methods

This study uses a one-shot multi-winner SOM with locally recurrent feedback connections based on [17]. Each fixed real-valued input pattern is presented to the SOM for a fixed number of time steps, after which the input pattern is removed while the SOM is allowed to continue running and adapting without external input. This continuation of activation and adaptation is a key component of our approach. As a result, the SOM generates a sparsely-coded activation pattern at each time step, retaining time-varying activity indefinitely after external input ends.

The artificial neurons (nodes) in the SOM are organized in a 2D rectangular grid as illustrated in Fig. 1. The neighborhood of a node i is the set of nodes that are located within a fixed radius r of the node i. Box distances are measured between nodes on the grid. Formally, let N_i denote the neighborhood of node i:

$$N_i = \{k \mid k \neq i \text{ and } d(i,k) \leq r\}, \quad (1)$$

$$d(i,k) = \max\left(|row_i - row_k|, |column_i - column_k|\right), \quad (2)$$

where the neighborhood radius is fixed as $r = 2$ throughout this study.

Each node i in the SOM receives two sets of connections (see Fig. 1 for an example). The first set consists of full connections from an external source of stimuli. These connections represent the afferent input of the SOM. Let $\boldsymbol{x}(t)$ denote the external input vector at time t, and $\boldsymbol{w}_i$ denote the weight values of the afferent connections of node i. The second set consists of topographical recurrent connections from neighboring nodes of node i, N_i. Let $\boldsymbol{u}_i$ denote the weight values of the recurrent connections. For generality, let $\boldsymbol{u}_i$ contain as many elements as the nodes in the SOM, only that each element is permanently set to zero if its corresponding node is not in N_i. Notice that by definition there is no self link from node i to itself, i.e., $u_{ii} = 0$ for each i. The net input for node i can then be expressed as

$$in_i(t) = \alpha_1 \boldsymbol{w}_i^{\mathsf{T}} \boldsymbol{x}(t) + \alpha_2 \boldsymbol{u}_i^{\mathsf{T}} \boldsymbol{a}(t-1), \quad (3)$$

where $\alpha_1 = 0.64$ and $\alpha_2 = 0.36$ are constant parameters specifying the relative weighting between afferent and recurrent connections (values are taken from [17]), and $\boldsymbol{a}(t-1)$ represents the activation vector of the SOM at the previous time step. It is assumed that the initial activation of the SOM is zero, i.e., $\boldsymbol{a}(t=-1) = \boldsymbol{0}$. A one-shot multi-winner activation

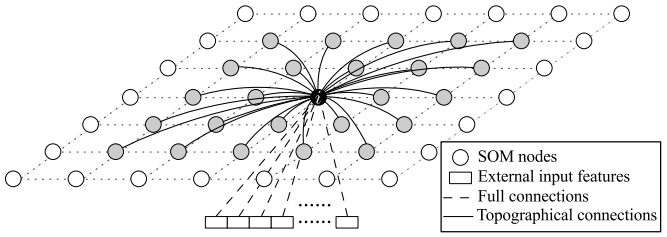

Fig. 1. A small example 7×7 SOM for illustrative purposes. Each node in the SOM has two sets of connections: fully-connected afferent connections and topographic recurrent connections (radius 2). For readability, only a small number of nodes are illustrated and incoming links for only node i are drawn. Shaded nodes indicate the neighborhood of node i (radius 2). Dotted grid lines do not represent connections but are drawn to indicate node adjacency.

rule is used to determine $\boldsymbol{a}(t)$, $t \geq 0$, from net input, where each $a_i(t) \in [0, 1]$ is calculated as:

$$winners(t) = \{k \mid in_k(t) > in_l(t), \forall l \in N_k\}, \quad (4)$$

$$a_i(t) = \min\left(1, \textstyle\sum_{k \in N_i \cap winners(t)} \gamma^{d(i,k)}\right). \quad (5)$$

Winners are nodes that have the highest net input in their local neighborhood. The activation level for each node is determined by how close it is to a winner node. The winners themselves are maximally activated, while their surrounding nodes have lower activation levels based on their distance from the winners. Together each winner node and its neighbors form a peak of activation centered at the winner. The shape of the peaks depends on the decay parameter γ ($0 \leq \gamma < 1$). A smaller γ makes steeper peaks.

The weight adaptation for the afferent connections follows competitive Hebbian learning:

$$\hat{\boldsymbol{w}}_i(t+1) = \boldsymbol{w}_i(t) + \mu_1 a_i(t) \boldsymbol{x}(t), \quad (6)$$

$$\boldsymbol{w}_i(t+1) = \hat{\boldsymbol{w}}_i(t+1) / \|\hat{\boldsymbol{w}}_i(t+1)\|_2, \quad (7)$$

where μ_1 is the learning rate.

The topographic recurrent connections are adapted using temporally asymmetric Hebbian learning [17], which is based on biological evidence that the efficacy of a synapse is strengthened if presynaptic firing precedes the postsynaptic firing in a 20 to 50 ms time window [28]–[30]. The weight value $u_{ik} \in \boldsymbol{u}_i$ for each connection from node k to node i is updated as:

$$g_i(t) = \max\left(0, a_i(t) - a_i(t-1)\right), \quad (8)$$

$$\hat{u_{ik}}(t+1) = u_{ik}(t) + \mu_2 a_k(t-1) g_i(t), \quad (9)$$

$$u_{ik}(t+1) = \hat{u_{ik}}(t+1) / \textstyle\sum_l \hat{u_{il}}(t+1), \quad (10)$$

where μ_2 is again the learning rate. The value of $g_i(t)$ specifies that the increase, rather than the value, of the activation level is taken into consideration.

For the specific spatial location application that we consider here, each external input represents the 2D coordinates (x, y) of a point within a 1×1 square, where $0 \leq x < 1, 0 \leq y < 1$. In practice, as in [17], each point (x, y) is projected to a unit sphere in order to obtain normalized vectors. The actual input vector becomes (a, b, c) where $a = x/\sqrt{2}$, $b = y/\sqrt{2}$, and $c = \sqrt{2 - x^2 - y^2}/\sqrt{2}$. The training data set is composed of 300 random points uniformly sampled within the 1×1 area. On the other hand, the evaluation, or testing, data set is composed of 100 grid points located at $x, y \in \{0, 0.1, 0.2, ..., 0.9\}$.

TABLE I. PARAMETERS FOR NONLINEARLY DECREASING FUNCTIONS USED DURING TRAINING

z	z_{init}	z_{fin}	z_{infl}	z_σ
μ_1	0.44	0	0.4	0.0001
μ_2	0.62	0	0.8	0.04
γ	0.37	0	0.2	0.16

The SOM is trained for 1000 epochs using the training data set. In each epoch, each vector in the training data set is presented to the SOM for 5 time steps (starting from $t = 0$), after which the SOM continues to update its activation states and adapt for another 5 time steps (determined empirically) without external input, i.e., with afferent activity $\boldsymbol{x}(t) = \boldsymbol{0}$. The activation and learning rules are applied accordingly throughout the 10 time steps for each input, although during the last 5 time steps, only recurrent weights are updated while the afferent weights are unchanged since the afferent input is all zeros (see Eq. 6). This continuation of activation and adaptation is critical to our approach, since it allows us to obtain and observe the on-going activation dynamics that naturally occurs *after* each external stimulus ends.

The learning rates μ_1, μ_2 and the peak activity decay parameter γ decrease nonlinearly as the epoch number progresses, according to the function $z = z_{\text{fin}} + (z_{\text{init}} - z_{\text{fin}})/(1 + \exp((\phi - z_{\text{infl}})/z_\sigma))$, where z is a parameter (μ_1, μ_2, or γ) and ϕ is the fraction of epochs completed. Relevant parameter values are listed in Table I. This function simulates the widely-adopted two phase training often used with SOMs, namely a rough organization phase with large learning rates and neighborhood sizes, followed by a fine-tuning phase with small learning rates and neighborhood sizes.

After training, all weights are fixed, and the peak activity decay parameter is set to $\gamma = 0$, i.e., all winners have activation level 1 and non-winners 0. Each vector in the evaluation data set is presented to the SOM for 5 time steps, and the activity attractor naturally occurring afterwards (from $t = 5$ on) in the SOM is designated to be the dynamic representation that encodes the input coordinate vector. The encoding attractors can be potentially qualitatively classified into three types: fixed point, limit cycle, and "complex". For brevity, the representation encoding a coordinate input (x, y) is denoted as $R_{(x,y)}$, which is an ordered list of spatial patterns. With a fixed point attractor, the dynamics eventually reach a state where further updating of the activation levels results in the same state persisting over time. In this case, the representation $R_{(x,y)}$ contains only this fixed-point state. For a limit cycle, the dynamics eventually result in a list of periodically repeating distinct states. This list of states is taken to be the representation $R_{(x,y)}$. Dynamics that do not show apparent regularity in 200 time steps are classified as being a complex attractor (including chaotic attractors and potentially very long limit cycles where no state has yet repeated during the observation window). In this study, we are more interested in the limit cycle attractors, since they exhibit oscillatory activities and have been shown to encode binary inputs representing phonemes and images quite well in our recent work [1].

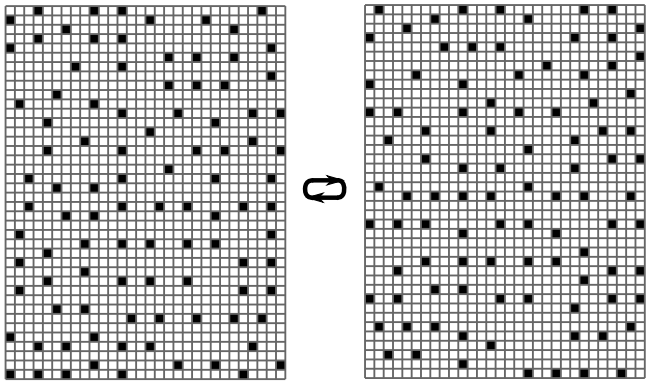

Fig. 2. A typical activity limit cycle $R_{(0.1,0.1)}$ encoding input coordinates (0.1, 0.1). This limit cycle contains 2 alternating states, each of which is a sparsely coded activation pattern. Each cell represents a node in the SOM. Dark cells represent activated nodes; light cells are inactive.

III. Results

The results below are obtained using a 40×30 SOM. A total of 20 independent simulations are performed with different initial random weights. Before training, the evaluation data set yields complex attractors and long limit cycles. After training, small limit cycles are detected for all input vectors in the evaluation data set. A majority of the limit cycles are of length 2, while in rare cases those of length 4 also occur, resulting in an average length of 2.016 ($SD = 0.024$). On average, a limit cycle occurs at 3.7 time steps ($SD = 0.2$) after each input stimulus is removed ($t = 5$). A typical activity limit cycle of length 2 representing coordinates $(x, y) = (0.1, 0.1)$, i.e., $R_{(0.1,0.1)}$, is depicted in Fig. 2.

Fig. 3 shows the differences between $R_{(0.1,0.1)}$ and $R_{(0.2,0.2)}$ ($R_{(0.9,0.9)}$). It can be observed that similar inputs, (0.1, 0.1) and (0.2, 0.2), yield similar limit cycles, where only a few nodes at the bottom-left corner have different activations (Fig. 3a). On the other hand, distant inputs, (0.1, 0.1) and (0.9, 0.9), yield quite different limit cycles (Fig. 3b).

To formally assess the similarity of limit cycles, we define the distance between two arbitrary limit cycles R and R' to be the minimum one-norm distance between their constituent states:

$$\mathrm{dist}(R, R') = \min_{\boldsymbol{p}\in R, \boldsymbol{q}\in R'} \|\boldsymbol{p} - \boldsymbol{q}\|_1, \tag{11}$$

where $\boldsymbol{p}, \boldsymbol{q}$ are any constituent activation states in R, R', respectively. This distance reflects the least number of nodes whose activation must be inverted (i.e., changing 0 to 1 or vice versa) to cause one limit cycle attractor to be converted into the other. This metric serves as a "conservative" lower-bound estimate of how far apart two limit cycles are from each other. Fig. 4 shows the distances between the limit cycles corresponding to all points in the evaluation data set and the limit cycles corresponding to the two selected points, (0.1, 0.1) and (0.5, 0.9). In both cases, the distances between limit cycles correspond quite well with the distances in the input locations. A gradient, although not perfect, is formed such that as input moves away from the selected points, the corresponding limit cycle gradually becomes more different (indicated by gradually lighter shades of the cell).

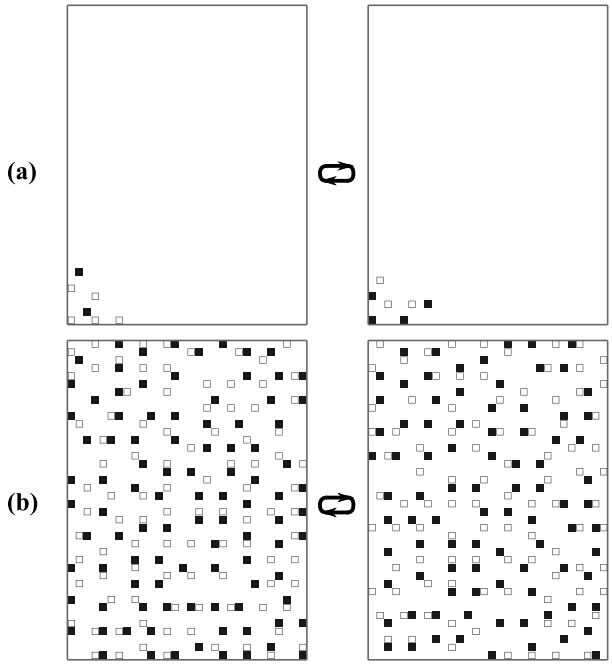

Fig. 3. Comparisons of limit cycles (a) $R_{(0.2,0.2)}$, which is the representation of an external location close to (0.1, 0.1), and (b) $R_{(0.9,0.9)}$, which is for an external location far from (0.1, 0.1), with $R_{(0.1,0.1)}$ (the latter was shown in Fig. 2). Only the differences of corresponding activation patterns are shown. Filled squares represent nodes that are activated in $R_{(0.2,0.2)}$ ($R_{(0.9,0.9)}$) but not in $R_{(0.1,0.1)}$; hollow squares represent nodes that are activated in $R_{(0.1,0.1)}$ but not in $R_{(0.2,0.2)}$ ($R_{(0.9,0.9)}$). The two states in $R_{(0.2,0.2)}$ and $R_{(0.9,0.9)}$ are aligned with those in $R_{(0.1,0.1)}$ such that the differences are minimized.

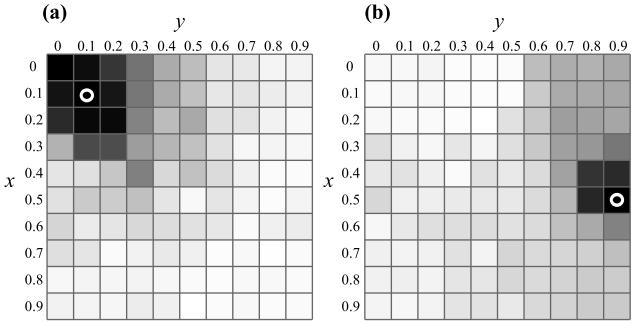

Fig. 4. The distances between the limit cycle for each point in the evaluation data set and the limit cycle for (a) the point (0.1, 0.1) and (b) the point (0.5, 0.9). Each cell represents a point in the evaluation data set. Lighter colors represent greater distances away from the selected points, (0.1, 0.1) or (0.5, 0.9). The two points, (0.1, 0.1) in (a) and (0.5, 0.9) in (b), are highlighted using white circles.

To further verify this phenomenon, we calculated the distance correlations [31], that is, the correlations between (1) the distance between any pair of input coordinates, measured using Euclidean distance, and (2) the distance between their corresponding limit cycles. Additionally, the average distance between any pair of limit cycles is also calculated. The average distance reflects the uniqueness of each limit cycle representation. A generally desirable property for encoding a set of items is the resulting representations being as unique as possible. The more unique a representation is, the less likely

TABLE II. DISTANCE CORRELATION AND AVERAGE DISTANCE OF LIMIT CYCLES BEFORE AND AFTER TRAINING

	Distance correlation	Average Distance
Before training	0.47 ($SD = 0.02$)	64.75 ($SD = 3.87$)
After training	0.89 ($SD = 0.01$)	156.29 ($SD = 1.55$)

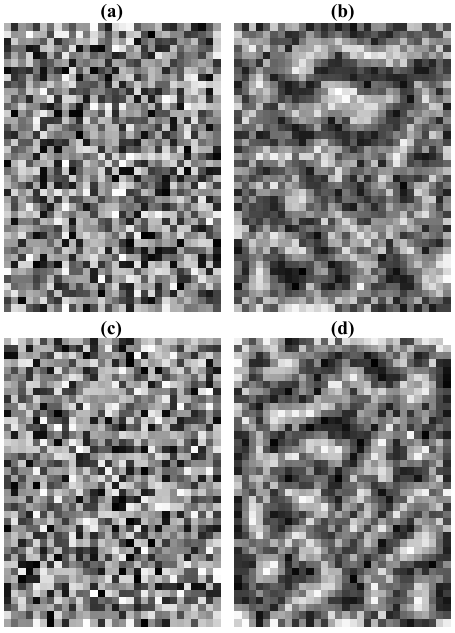

Fig. 5. Map formation (a, c) before and (b, d) after training. (a) and (b) show the weight values corresponding to the x component of input coordinates; (c) and (d) shows the weight values corresponding to the y component. Each cell corresponds to a node in the SOM. Brighter cells indicate higher weight values.

its corresponding item is to be confused with other items in subsequent processing, such as in a classification task. The distance metric in Eq. 11 is used here.

Table II summarizes the results concerning distance correlation and average distance between the limit cycles. After training, both metrics are significantly increased. For distance correlation, the value obtained after training indicates that distances in the input space are highly correlated with distances in the limit cycle state space (a value of 1 indicates two variables being almost surely dependent, while 0 indicates statistical independence). In other words, there is a strong tendency that similar inputs will result in similar limit cycles, and distinct inputs will results in distinct limit cycles. At the same time, the distance between any pair of limit cycles increases after training. This indicates that each limit cycle becomes more unique. These two properties together demonstrate that the limit cycles encode coordinate inputs quite well.

Finally, the weight values are visualized in Fig. 5. Map formation is obviously observed after training (right column), where a gradient is formed between clusters of bright cells (high weight values) and dark cells (low weight values). The existence of multiple bright (dark) clusters is caused by multi-winners-take-all activation, similar to the observations in [1].

IV. DISCUSSION

In our recent work [1], we established conditions for multi-winner SOMs to represent varied-length temporal sequences of binary stimuli (phoneme sequences of words and images of the objects they describe) using small limit cycles of variable lengths, through on-going activation and adaptation. These limit cycles were shown to be unique and resistant to perturbations, and can be associated with one another, such that one limit cycle in a SOM is able to trigger another limit cycle in another SOM while exact operation timing is not critical.

In this study, we have shown that small limit cycles emerged in response to 2D coordinate inputs. We believe that these limit cycles are well suited to serve as representations for the corresponding 2D locations, because they are found to be quite unique and simultaneously highly correlated with the input locations, in a way that similar inputs result in similar limit cycles and different inputs result in different limit cycles. This property was not obvious in our past work and is reported for the first time here. Further, despite using post-stimulus dynamics and limit cycle representations, the SOM still forms a smooth map — the weight changes between neighboring nodes become smoother after training and clustering occurs.

Compared with our recent work [1], this study uses real-valued coordinate inputs rather than binary inputs. Additionally, the training and evaluation data sets are separate in this study. The training data set is sampled randomly in the input space, while the evaluation data set consists of specific positions that form a grid in the input space. Under these new conditions, small limit cycles still emerged as a result of unsupervised learning, which are comparable to those reported previously. The results in this study suggest that real-valued inputs with randomized training data are likely to enhance the correlations between inputs and limit cycles, as compared with [1]. Another difference is that, in this study, the post-stimulus adaptation time required to obtain small limit cycles (5 time steps) is shorter than in our previous work ($\geq$ 15 times steps). The lengths of the resulting limit cycles are also less varied in this study (mostly 2), which may be the result of presenting fixed inputs for fixed durations, as opposed to varied-length temporal sequences in [1].

In conclusion, our initial studies show that limit cycles in SOMs have the potential to work in biologically-plausible neurocomputational models, and therefore they merit further investigation. In the future, large-scale statistical experiments and larger neural architectures with SOMs need to be explored to better support using this type of neurodynamics.

ACKNOWLEDGMENT

This work was supported by Office of Naval Research (ONR) award N000141310597, United States of America.

REFERENCES

[1] D.-W. Huang, R. J. Gentili, and J. A. Reggia, "Self-organizing maps based on limit cycle attractors," *submitted*, 2014.

[2] H. de Garis, C. Shuo, B. Goertzel, and L. Ruiting, "A world survey of artificial brain projects, part I: Large-scale brain simulations," *Neurocomputing*, vol. 74, no. 1–3, pp. 3 – 29, 2010.

[3] C. Eliasmith, T. C. Stewart, X. Choo, T. Bekolay, T. DeWolf, Y. Tang, and D. Rasmussen, "A large-scale model of the functioning brain," *Science*, vol. 338, no. 6111, pp. 1202–1205, 2012.

[4] S. Grossberg, "Adaptive resonance theory: How a brain learns to consciously attend, learn, and recognize a changing world," *Neural Networks*, vol. 37, pp. 1–47, 2013.

[5] S. A. Weems and J. A. Reggia, "Simulating single word processing in the classic aphasia syndromes based on the Wernicke–Lichtheim–Geschwind theory," *Brain and Language*, vol. 98, no. 3, pp. 291 – 309, 2006.

[6] R. Winder, C. R. Cortes, J. A. Reggia, and M.-A. Tagamets, "Functional connectivity in fMRI: A modeling approach for estimation and for relating to local circuits," *NeuroImage*, vol. 34, no. 3, pp. 1093 – 1107, 2007.

[7] A. Abbott, "Solving the brain," *Nature*, vol. 499, no. 7458, pp. 272–274, 2013.

[8] C. Malsburg, "Self-organization of orientation sensitive cells in the striate cortex," *Kybernetik*, vol. 14, no. 2, pp. 85–100, 1973.

[9] T. Kohonen, *Self-organizing maps*, 3rd ed., ser. Springer Series in Information Sciences. Springer, 2001, vol. 30.

[10] N. Manukyan, M. Eppstein, and D. M. Rizzo, "Data-driven cluster reinforcement and visualization in sparsely-matched self-organizing maps," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 5, pp. 846–852, 2012.

[11] C.-C. Hsu and S.-H. Lin, "Visualized analysis of mixed numeric and categorical data via extended self-organizing map," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 23, no. 1, pp. 72–86, 2012.

[12] R. Miikkulainen, J. A. Bednar, Y. Choe, and J. Sirosh, *Computational maps in the visual cortex*. Springer, 2005.

[13] G. G. Sutton, J. A. Reggia, S. L. Armentrout, and C. L. D'Autrechy, "Cortical map reorganization as a competitive process," *Neural Computation*, vol. 6, no. 1, pp. 1–13, 1994.

[14] Y. Chen and J. A. Reggia, "Alignment of coexisting cortical maps in a motor control model," *Neural Computation*, vol. 8, no. 4, pp. 731–755, 1996.

[15] J. Sylvester and J. Reggia, "Plasticity-induced symmetry relationships between adjacent self-organizing topographic maps," *Neural Computation*, vol. 21, no. 12, pp. 3429–3443, 2009.

[16] J. A. Bednar and R. Miikkulainen, "Tilt aftereffects in a self-organizing model of the primary visual cortex," *Neural Computation*, vol. 12, no. 7, pp. 1721–1740, 2000.

[17] R. Schulz and J. A. Reggia, "Temporally asymmetric learning supports sequence processing in multi-winner self-organizing maps," *Neural Computation*, vol. 16, no. 3, pp. 535–561, 2004.

[18] G. Buzsaki, *Rhythms of the Brain*. Oxford University Press, 2006.

[19] E. Niedermeyer and F. da Silva, *Electroencephalography: Basic Principles, Clinical Applications, and Related Fields*. Lippincott Williams & Wilkins, 2005.

[20] J. Fell and N. Axmacher, "The role of phase synchronization in memory processes," *Nat Rev Neurosci*, vol. 12, no. 2, pp. 105–118, 2011.

[21] D. J. Amit, *Modeling brain function: The world of attractor neural networks*. Cambridge University Press, 1992.

[22] W. J. Freeman, *Neurodynamics: An Exploration in Mesoscopic Brain Dynamics*. Springer, 2000.

[23] A. Samsonovich and B. L. McNaughton, "Path integration and cognitive mapping in a continuous attractor neural network model," *The Journal of neuroscience*, vol. 17, no. 15, pp. 5900–5920, 1997.

[24] D. Sussillo and L. F. Abbott, "Generating coherent patterns of activity from chaotic neural networks," *Neuron*, vol. 63, no. 4, pp. 544–557, 08 2009.

[25] D. Wang, "Emergent synchrony in locally coupled neural oscillators," *IEEE Transactions on Neural Networks*, vol. 6, no. 4, pp. 941–948, 1995.

[26] M. Kaipainen and T. Ilmonen, "Period detection and representation by recurrent oscillatory self-organizing map," *Neurocomputing*, vol. 55, no. 3–4, pp. 699 – 710, 2003.

[27] T. Rumbell, S. Denham, and T. Wennekers, "A spiking self-organizing map combining stdp, oscillations, and continuous learning," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 25, no. 5, pp. 894–907, 2014.

[28] G.-q. Bi and M.-m. Poo, "Synaptic modifications in cultured hippocampal neurons: Dependence on spike timing, synaptic strength, and postsynaptic cell type," *The Journal of Neuroscience*, vol. 18, no. 24, pp. 10 464–10 472, 1998.

[29] H. Markram, J. Lübke, M. Frotscher, and B. Sakmann, "Regulation of synaptic efficacy by coincidence of postsynaptic APs and EPSPs," *Science*, vol. 275, no. 5297, pp. 213–215, 1997.

[30] L. I. Zhang, H. W. Tao, C. E. Holt, W. A. Harris, and M.-m. Poo, "A critical window for cooperation and competition among developing retinotectal synapses," *Nature*, vol. 395, no. 6697, pp. 37–44, 1998.

[31] G. J. Székely, M. L. Rizzo, and N. K. Bakirov, "Measuring and testing dependence by correlation of distances," *The Annals of Statistics*, vol. 35, no. 6, pp. 2769–2794, 2007.