# An Instantaneous Topological Mapping Model for Correlated Stimuli

Ján Jockusch and Helge Ritter, Neuroinformatics Dept, University of Bielefeld
{ jan | helge }@techfak.uni-bielefeld.de

## Abstract

*Topology-representing networks, such as the SOM and the Growing Neural Gas (GNG) are powerful tools for the adaptive formation of maps of feature and state spaces for a broad range of applications. However, these algorithms suffer severe difficulties when their training inputs are strongly correlated. This makes them unsuitable for the on-line formation of maps of state spaces whose exploration occurs most naturally along trajectories, which is typical in many applications in the fields of robotics and process control. Based on investigations of the SOM and the GNG for these cases, we devise a new network model, the "Instantaneous Topological Map" (ITM) that is able to overcome these difficulties and form maps from strongly correlated stimulus sequences in a fast and robust manner. This makes the ITM highly suitable for mapping of state spaces in control tasks in general and especially in robotics, where workspace limitations are complex and probably more easily explored than analyzed and coded by hand.*

## Introduction

The enormous success of Kohonen's famous and beautifully simple self-organizing map [6][8] has inspired a large body of work towards even more powerful algorithms for the adaptive creation of topology-representing mappings of various feature and state spaces [3][1][10]. The common mathematical foundation of these approaches is usually of statistical nature, e.g., error minimization [5] or entropy maximization [2], with a few exceptions, like the PSOM, which is based on an interpolation approach [9].

While the SOM and their closer cousins are characterized by a pre-specified and fixed map topology (a 2d grid in most cases) which is then matched against the data, subsequent research has addressed the issue of identifying a suitable topology as part of the learning process. This ability is helpful when the correct topological structure of the data is unknown a-priori, such as, e.g., when creating maps of environments containing obstacles of unknown shape. We will use such a mapping task to evaluate neural networks throughout this paper.

Among the algorithms that have come into more widespread use, a particularly interesting proposal —related to the neural gas of Martinetz [7]— has been due to Fritzke [4]. His "Growing Neural Gas" (GNG) starts with a "gas" of nodes and builds the required links during learning, inserting new nodes in regular intervals and in regions where an averaged approximation error for the processed data is largest. An "aging mechanism" to remove links that have become obsolete due to the migration of nodes to new positions also endows the GNG with some capability to adapt to slowly changing topologies.

While the GNG thereby extends the adaptive capabilities of the SOM, the design of both algorithms is still heavily based on the assumption that the training stimuli are *statistically uncorrelated*. Below, we will consider the behavior of the SOM and the GNG when this assumption is violated. In particular, we will focus on the case when strong correlations are present due to stimuli that have been generated by exploration of the state or feature space along continuous trajectories, which is a frequent case in robotics and control applications. We will show that in this case both methods face severe degradation, which excludes their use for many on-line applications in the aforementioned fields.

Based on an analysis of the difficulties of the GNG when trained with trajectory data, we then propose a different set of adaptation rules, leading to a new algorithm which no longer relies on statistical independence of its training data and which is particularly well suited for rapidly forming topology-representing maps even for training data with a strong serial correlation. We call this algorithm "Instantaneous Topological Map" (ITM), since it no longer requires the maintenance of any averages accumulated over time.[1] Instead, it only uses information that is local both in space and in time, making it computationally extremely efficient without sacrificing the advantages of the GNG. We compare the performance of the ITM with the GNG and show that it leads to significantly faster adaptation without any significant loss in the quality of the generated maps. We then present our conclusions and discuss some possibilities for future work.

[1] Other than the main variables, i.e., the node positions and the inserted edges themselves.

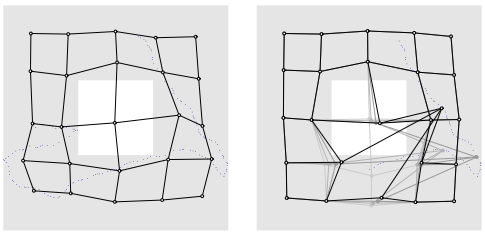

Figure 1: Stimulation of a SOM with a random walk trajectory. At low learning rates (left), the topology remains intact and the network will adapt correctly in the long term. If faster learning is desired, raising the learning rate (right) in this setting turns out to be a bad idea. Individual nodes "attach" to the trajectory, thereby destroying any already established topological integrity.

## Improving the GNG for Correlated Stimuli

Our main interest is in situations where exploration of the state or feature space occurs along continuous trajectories, possibly with some moderate amount of superimposed noise. As our data model to mimic that situation we consider a (discrete) random walk with small step size $d$, given by

$$\begin{aligned} \vec{x}(t+1) &= \vec{x}(t) + \vec{p}(d, \alpha(t)) \quad \text{and} \\ \alpha(t+1) &= \alpha(t) + \eta, \end{aligned} \tag{1}$$

where $\vec{p}(d, \alpha)$ is the polar coordinate representation of a step of length $d$ in the angular direction $\alpha$, and $\eta$ is a random variable. The step length $d$ remains constant while the angle $\alpha$ changes by uniformly distributed random amounts $\eta$. Workspace limits are implemented by simply forbidding steps that lead outside of the allowed area.

If we try to form a map of the state space explored by such sequences using the standard SOM algorithm, we observe a very strong dependence on the size of the learning rate. At low rates, the topology remains intact, but nodes ignore the trajectory information presented. At high rates, a single node "follows" the random walk path, disrupting the topology and destroying information that has already been accumulated from previous samples (see fig. 1).

A partial reduction of these difficulties can be achieved by providing a suitable "healing mechanism" for the topological disturbances that are introduced by the correlated stimulus motion. The ability of the GNG to adjust its topology to the situation can already fill this need to some extent, although we shall soon see that the healing effect is still too weak to allow a satisfactory performance.

However, first we wish to briefly recall the main ingredients of the GNG in order to provide the background for the following discussion and for the design of the ITM in the next section.

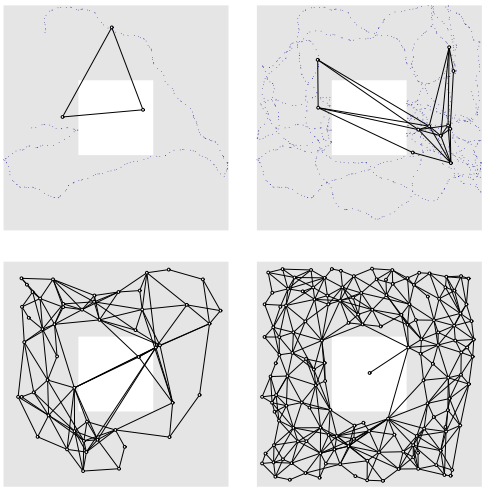

Figure 2: Adaptation of a standard GNG with correlated stimuli. The network has been parametrized in such a way that the final result approximately matches that of the enhanced GNG and the ITM. Especially in the starting phase, the standard GNG leaves large portions of the trajectory uncharted.

The basic GNG algorithm works on a set of nodes $i$, each represented by a weight vector $w_i$ and an accumulated error $e_i$, and a set of edges $j$ with an age value $a_j$. The adaptation with a new stimulus $\xi$ consists of four distinct steps.

**1. Matching:** Find the node $n$ nearest to the stimulus $\xi$ and the second-nearest node $s$.

**2. Reference vector adaptation:** Given adaptation rates $\epsilon_1$ and $\epsilon_2$, adapt the nearest node and its topological neighbors as follows: $\Delta w_n = \epsilon_1 (\xi - w_n)$, $\Delta w_i = \epsilon_2 (\xi - w_i) \quad \forall \quad i \in N(n)$, where $N(n)$ denotes the set of neighbors of $n$.

**3. Edge update:** *(i)* Create an edge connecting $n$ and $s$ if it does not already exist. Set that edge's age to zero. *(ii)* Increment the age of all other edges emanating from $n$ and delete any whose age surpasses a given $a_{\max}$. When deleting an edge, check the other referenced node for emanating edges; if there are none, remove that node as well.

**4. Node update:** *(i)* Increment the error measure of the nearest node: $\Delta e_n = \|\xi - w_n\|^2$. *(ii)* Add a new node every $\lambda$ adaptation steps by finding the node $q$ with maximum accumulated error and its neighbor $r$ with maximum accumulated error: $q = \arg\max_i e_i$, $r = \arg\max_{j \in N(q)} e_j$. Make a new unit $s$ with $w_s = \frac{1}{2}(w_q + w_r)$ and initialize its error with $e_q$. Decrease the errors of $q$, $r$, and $s$ by a given factor $\alpha$. *(iii)* Multiply the errors of all nodes with a decay factor $d$, so that they cannot grow indefinitely.

The familiar matching and reference vector adaptation steps are the heritage of Kohonen's SOM. We will discuss their

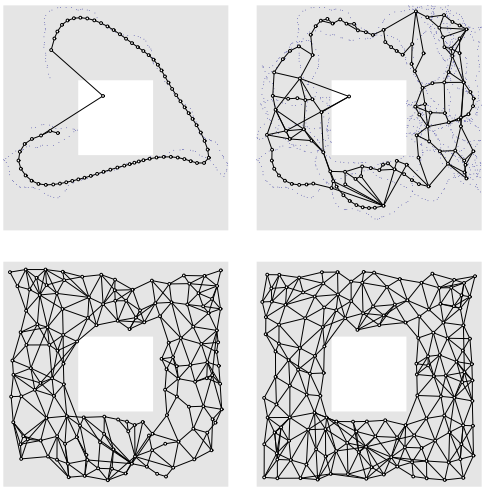

Figure 3: Improved GNG using an error threshold. Input is the same sequence used in fig. 2. In the startup phase, neurons are now created much faster to learn the trajectories traversed. As the error approaches the desired value, fewer new nodes are added.

changing role in the ITM later on. The topological adaptation steps, though, deserve a closer look.

The edge creation rule builds an edge of the Delaunay triangulation, given that the weight vectors of the nearest and second-nearest nodes are on opposite sides of the stimulus, because in this case, the nearest and second-nearest nodes share a Voronoi cell border. An edge created in this way may become obsolete if new nodes are created or the nodes move. The aging mechanism erases such edges eventually, but finding a suitable age limit can be difficult. The same limit may delete useful edges and still leave many useless ones untouched.

A noteworthy fact is that the construction of a Delaunay edge does not rely on the distribution of stimuli. Edges will be constructed in the fashion described even if stimuli touch only selected trajectories in input space. The triangulation is not guaranteed to be complete, but that is the case for both statistical and correlated series of stimuli: the Delaunay edges corresponding to shorter Voronoi borders are less likely to be constructed.

The node creation mechanism is less obvious and leaves more room for choice. The error accumulation provides a means of determining the optimum position of the next node. From a statistical point of view, this is a straightforward approach. In adapting to sequences of stimuli resembling trajectories, though, smarter node creation and deletion algorithms can solve the problem of topological disturbances created by single nodes following the trace of stimuli for long distances.

Our first approach to improving node creation works by defining a threshold value for the error, $e_{\max}$, with which the error measure of the nearest neuron, $e_n$, is compared. If it is larger, a new node is created between $n$ and the neighboring node with highest error count. This small design change alone gives the dramatic improvement shown in figures 2 and 3.

The modified algorithm proves to be more flexible than standard GNGs. Node creation now is a reaction to certain stimulus patterns, instead of being triggered by fixed external clock cycles. The main advantage comes from switching from a global method, i.e., finding the node with highest accumulated error, to a local method. Designing the algorithm to only use the neighborhood of a node for adaptation keeps the computational effort low even for very large networks.

## The Instantaneous Topological Map (ITM)

There are still two disadvantages to keeping an error measure and edge age count. *(i)* More parameters (error decay rate, error threshold, error distribution factor, maximal age) mean more hassle when optimizing a network. The parameter values are not very critical, but a wrong choice can still slow down the convergence considerably, or destroy it completely. *(ii)* Each slowly changing state variable (age and error count) introduced into the system produces some inertia, slowing down adaptation and defining a characteristic timescale which must be accounted for. The amount of time the network needs to react to changes in input stimulus pattern depends very much on the choice of the corresponding decay factors.

We therefore propose a new network type which does not need any edge aging or error accumulation to generate its map. In fact, it does not even require node adaptation. We call it ITM, for "Instantaneous Topological Map".

The ITM consists of a set of neurons $i$ with weight vectors $w_i$, and a set of undirected edges, represented implicitly by specifying a set of node neighbors $N(i)$ for each node $i$.[2] The network starts out with only two connected nodes. The adaptation triggered by a new stimulus $\xi$ consists of the following steps:

**1. Matching:** Find the nearest node $n$ and the second-nearest node $s$ (with respect to a given distance measure, e.g., the Euclidean distance): $n = \arg\min_i \|\xi - w_i\|$, $s = \arg\min_{j,j\neq n} \|\xi - w_j\|$.

**2. Reference vector adaptation:** Move the weight vector of the nearest node toward the stimulus by a small fraction $\epsilon$:

[2]The $N(i)$ are further constrained by the requirement that neighborhood relations between a pair of nodes shall always be symmetric.

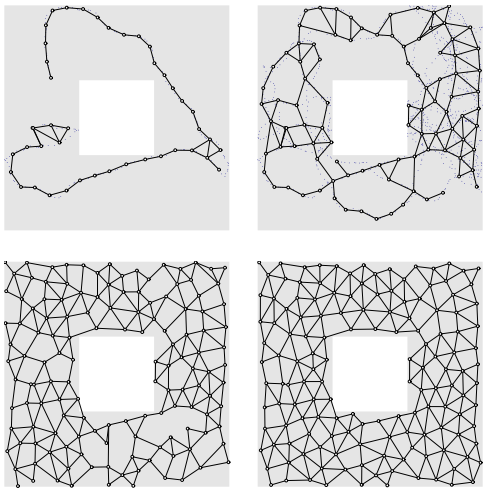

Figure 4: ITM network charting the same trajectory sequence as in fig. 2. The adaptation rate $\epsilon$ is not critical in this method, it can safely be set to zero.

$\Delta w_n = \epsilon\,(\xi - w_n)$. Below we will show that this step can even be omitted.

**3. Edge adaptation:** *(i)* Create an edge connecting $n$ and $s$ if it does not already exist. *(ii)* For each member $m$ of $N(n)$ check if $w_s$ lies inside the Thales sphere through $w_n$ and $w_m$. If that is the case, remove the edge connecting $n$ and $m$. When deleting an edge, check $m$ for emanating edges; if there are none, remove that node as well (see fig. 5).

**4. Node adaptation:** *(i)* If the stimulus $\xi$ lies outside the Thales sphere through $w_n$ and $w_s$, and outside a sphere around $w_n$ with a given radius $e_{\max}$, create a new node $y$ with $w_y = \xi$. Connect nodes $y$ and $n$. *(ii)* If $w_n$ and $w_s$ are closer than $\frac{1}{2}e_{\max}$, remove $s$ (see fig. 6).

In terms of **computational expense**, the matching step is the only step that scales with the number of neurons. Edge adaptation scales with the average number of neighbors, which is related to the local intrinsic dimensionality of the input data. All other operations are independent of the number of neurons involved. This means that the algorithm executes fast even for large networks.

Our experience with the algorithm indicates that **reference vector adaptation** (step 2) can even be omitted because nodes are created and deleted swiftly if the node distribution is found to be too sparse or too dense. The former learning rate $\epsilon$, which was essential to adjust the network to fit the input data, has now assumed the role of a smoothing parameter. Choosing small values of $\epsilon$ makes the nodes slowly assemble in a tidy arrangement with distances between nodes

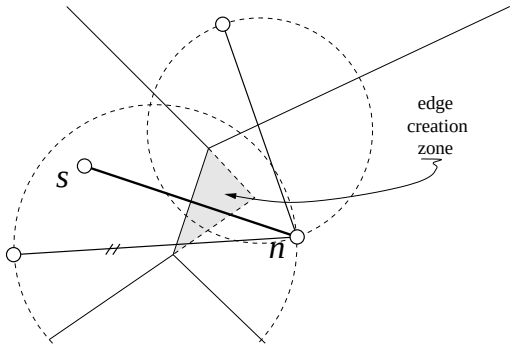


Figure 5: Edge addition is triggered when a stimulus hits the grey region where the Voronoi cell of $n$ intersects the Voronoi cell of $s$ if $n$ were not present. Removal of edges is triggered by the Thales sphere through $n$ and one of its neighbors. If $s$ lies inside that sphere, the corresponding edge is removed (the marked lower edge in the figure).

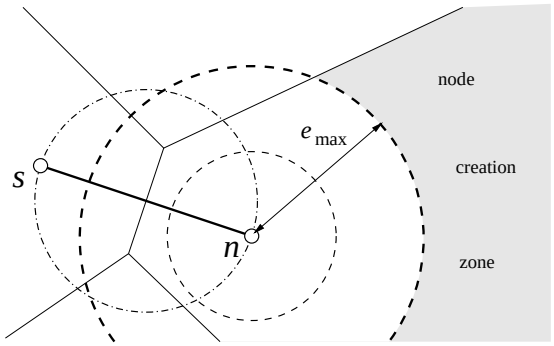


Figure 6: Node addition and removal in the ITM is guided by the Thales sphere through the nearest two nodes, $n$ and $s$, and spheres through $n$ of radius $e_{\max}$ and $\frac{1}{2}e_{\max}$.

approximately equal. The relaxation time of this process does not affect the network's overall performance.

**Edge creation** produces a valid Delaunay edge, as stated before. This edge is then used to verify the other edges emanating from the nearest node. Only those edges are kept which cross the corresponding Voronoi cell border. This eliminates all non-Delaunay edges and few Delaunay edges, mainly those belonging to the convex hull. The advantage of this method compared to former edge deletion techniques is that it destroys all non-Delaunay edges and that it does not rely on parameter tuning to do so. An exhaustive Delaunay test which detects even small eccentric Voronoi borders between connected nodes would be computationally much more expensive, as the amount of calculations needed would scale with the dimensionality of the data *and* the total number of nodes.

**Node creation** avoids putting new nodes inside the Thales sphere through nearest and second-nearest node, because

doing this would render useless the connection just made. If the stimulus lies farther away from the nearest node than a given threshold, a new node is created at the position of the stimulus. The threshold, $e_{\max}$, therefore has the meaning of a desired mapping resolution. This method is substantially different from providing a learning rate, as nodes are created at a maximum speed of one per stimulus if necessary, in which case the network stores the input data in weight vectors, and their order of arrival in its graph. Because nodes can still move by a small amount in this algorithm, a criterion is provided to remove nodes that are too close to each other. The threshold used is derived from $e_{\max}$.

**Configuring an ITM network** is exceedingly easy, since only at most two parameters need to be found: the desired resolution $e_{\max}$, and, optionally, the smoothing parameter $\epsilon$ (the former learning rate).

## Results

We use the random walk sequence of stimuli generated by equation (1) to measure and compare the performance of the three network models just described. This sequence is the simplest model of an autonomous robot driving randomly through a room with a square obstacle in the middle. The objective is to map this room using a neural network.

Figures 2–4 show four phases in the adaptation of the standard GNG, the enhanced GNG (error triggered node generation), and the ITM, respectively. The network parameters are chosen so that each network arrives at the same number of nodes at 15000 samples; in this way the intermediate phases can more easily be compared.

The ITM's normalized root mean square error (NRMSE) stays almost constant during training. This comes from its immediately creating nodes to achieve a desired precision. The slower error decay of the enhanced GNG originates from the inertia introduced by the error accumulators and the learning rate. The standard GNG is designed to slowly improve its mean error by adding nodes at regular intervals.

The edge creation and deletion algorithm's performance shows up when comparing the network's graph to the Delaunay triangulation. The edge aging mechanism of both GNG models is responsible for the high number of surplus edges, about ten to twenty percent, while the extremely strict immediate removal rule of the ITM lowers this number to almost zero. The small advantage of the enhanced GNG over the standard version can be explained with the better optimization of node placement. Each newly placed node makes some existing edges obsolete, and since standard GNG node creation frequency stays equally high throughout the experiment, the proportion of creating and deleting obsolete edges is less favorable.

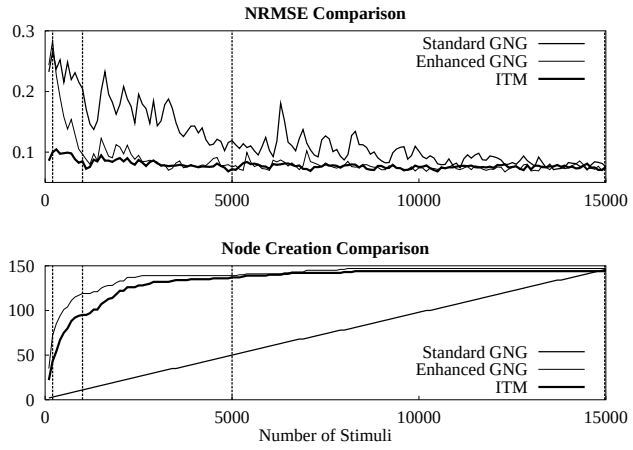


Figure 7: The graphs show measurements made on the three models discussed. The random walk stimulus pattern used can be seen in the figures 2–4, which show snapshots after 200, 1000, 5000, and 15000 samples (dashed lines in the graphs above). Although it creates nodes more slowly than the enhanced GNG, the ITM achieves the desired error from the start because it can create new nodes outside the current scope of the network.

The number of edges missing to complete the Delaunay triangulation —the bottom graph in fig. 8— cannot drop to zero in our experiment because of the obstacle in the middle of the imaginary room. Edge creation functions by the same principle in all network models, so they perform almost equally well in this respect, with a slight disadvantage for the ITM. This is because of the simple but very strict immediate edge deletion technique used, which sometimes erases even valid Delaunay edges.

## Conclusions

**Statistical Distributions:** Although we introduced and validated the ITM network model on the basis of trajectory-like series of stimuli, the ITM still performs very well in settings with statistically uncorrelated stimulus distribution. In these settings, too, the ITM can outperform the other network models in terms of convergence speed, because it has no inner state variables that can introduce inertial effects. Each adaptation step can produce and remove nodes and edges immediately, with no dependency on the network's past history.

**Architecture Comparison:** The ITM is in some respects complementary to the SOM. While the SOM has rigid topology and relies on learning rate and smoothness parameter annealing to adjust the nodes' positions and map the underlying topology, the ITM has rigid node positions and generates the topology with adaptation rules. While the SOM does not depend on changing topology to produce useful

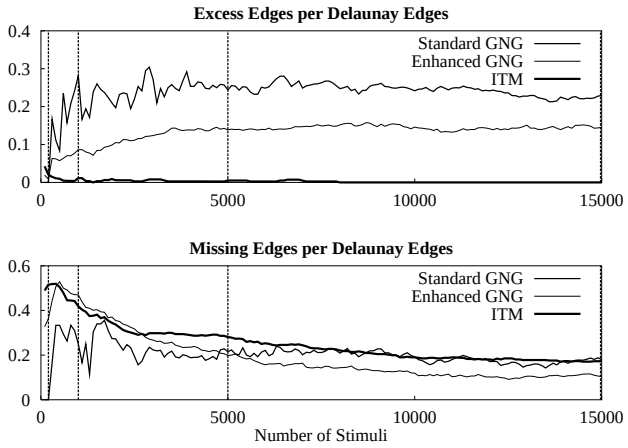


Figure 8: Measurements to gauge the quality of the connections created by the three network models discussed. The performance advantage of the ITM becomes apparent in the low number of excess edges which would carry no useful information for path finding. The underlying experiment is the same as in figure 7.

mappings, but can benefit from such additions (as in the GNG), the ITM does not depend on changing the nodes' positions, but can benefit from a small smoothness term (the former learning rate).

One aspect of using the precision parameter, $e_{\max}$, instead of a SOM-type learning rate, is that nodes are always approximately equally spaced. The ITM model is not suitable for applications where the network's node density needs to be a function of statistical stimulus density. This famous property of the GNG and the SOM is rooted in the adaptation of the nearest node and its topological neighbors; it cannot be replicated in the ITM.

**Dimensionality of Input Data:** Our experiment involves two-dimensional input vectors for clarity of the presentation. In our tests, the ITM has performed equally successful with higher intrinsic dimensionality of input data. With rising dimensionality, the ITM's advantage in swiftly deleting useless edges even grows, both in terms of mapping reliability and computing expense, because fewer stray emanating edges per node mean shorter edge testing loops.

**Applications:** The Instantaneous Topological Map produces reliable charts of trajectories without the need for special preparation of input samples. There are no delays in the construction of the map. These factors make the ITM especially useful in robotic control applications. A robot exploring its surroundings can store the topological data in an ITM, which then works like an associative memory device. The robot can then use the map literally to get from one location, represented by one node, to another: following the shortest path in the ITM's graph leads it to the target, automatically avoiding obstacles.

Many control processes involve finding an effective way of setting input values in order to reach a target output value with minimum effort. Using an ITM to map the state space while a simple controller is operating can turn up a more efficient pathway leading to the target position. Nodes along that pathway correspond to a series of target settings for the controller. Feeding that series instead of the final target values to the controller can lead to better overall performance.

## References


[1] I. Ahrns, J. Bruske, and G. Sommer. On-line learning with dynamic cell structures. In *Proceedings of ICANN'95*, volume 2, pages 141–146, 1995.

[2] C. M. Bishop, M. Svensén, and C. K. I. Williams. GTM: The generative topographic mapping. *Neural Computation*, 10(1):215–234, 1998.

[3] B. Fritzke. Growing cell structures — a self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7(9):1441–1460, 1994.

[4] B. Fritzke. A growing neural gas network learns topologies. *Advances in Neural Information Processing Systems*, 7:625–632, 1995.

[5] B. Fritzke. A self-organizing network that can follow nonstationary distributions. In *Proceedings of ICANN'97*, pages 613–618. Springer, 1997.

[6] T. Kohonen. Self-organized formation of topologically correct feature maps. *Biological Cybernetics*, 43:59–69, 1982.

[7] T. M. Martinetz and K. J. Schulten. Topology representing networks. *Neural Networks*, 7(3):507–522, 1994.

[8] H. Ritter, T. M. Martinetz, and K. J. Schulten. *Neural Computation and Self-Organizing Maps*. Addison-Wesley, 1992.

[9] H. J. Ritter. Parametrized self-organizing maps. In *Proceedings of ICANN'93*, pages 568–575. Springer, 1993.

[10] H. J. Ritter and T. Kohonen. Self-organizing semantic maps. *Biological Cybernetics*, 61, 1989.