A Hybrid Self-organizing Neural Gas Based Network

James Graham and Janusz A. Starzyk†

***Abstract*—This paper examines the neural gas networks proposed by Martinetz and Schulten [1] and Fritzke [2] in an effort to create a more biologically plausible hybrid version. The hybrid algorithm proposed in this work retains most of the advantages of the Growing Neural Gas (GNG) algorithm while adapting a reduced parameter and more biologically plausible design. It retains the ability to place nodes where needed, as in the GNG algorithm, without actually having to introduce new nodes. Also, by removing the weight and error adjusting parameters, the guesswork required to determine parameters is eliminated. When compared to Fritzke's algorithm, the hybrid algorithm performs admirably in terms of the quality of results it is slightly slower due to the greater computational overhead. However, it is more biologically feasible and somewhat more flexible due to its hybrid nature and lack of reliance on adjustment parameters.**

I. Introduction

The neural gas network is a biologically inspired ontogenic network. It is an unsupervised network that operates on a few central principles. A certain number of neurons or nodes (usually two) are selected by how far they are from an input signal. They are then adjusted according to a fixed algorithm/schedule determined by the algorithm's designer. The networks discussed in this paper are referred to as "neural gas" structures because of their similarity to the original neural gas network as it was first proposed by Martinetz and Schulten, in 1991 [1], and later expanded upon by Bernd Fritzke [2], [3]. Fritzke has produced several articles detailing many variations of neural gas networks and their properties.

Since then, there has been an increasing interest in neural gas related research as an alternative to Kohonen self-organizing maps [4] as evidenced by a number of papers that took advantage of the neural gas flexibility to fit various topologies. For example, Montes et. al. presented a hybrid ant-based clustering algorithm that made use of growing neural gas networks [5]. Also, Rodriguez presented an application of the growing neural gas model for automatically building statistical models of non-rigid objects such as hands [6]. Another example of research based on the growing neural gas network is the work done by Cheng and Zell [7] which presents a double growing neural gas network for disease diagnosis. In the double network, the network's growth is accelerated by doubling the number of neurons that are inserted during each insertion stage. Other examples of the applications of neural gas networks includes a method for training vector quantizers by Rovetta and Zunino [8], a method for assessing the stability of electric power systems from Rehtanz and Leder [9], and a hierarchical neural gas network for pattern recognition by Atukorale and Suganthan [10].

While the preceding articles give a good overview of the applications of growing neural gas networks, they do not offer comparisons to other neural network types. The paper by Heinke and Hamker [11] addresses this issue by comparing the growing neural gas network with such networks as the growing cell structures network (GCS), fuzzy ARTMAP (fuzzy predictive adaptive resonance theory) , and the standard multilayer perceptron network (MLP). They tested four different real world data sets with the four types of networks with varying results. No network type performed significantly better than the others on all four of the data sets. However, they concluded that when taking performance, convergence time, and dependence on parameters into account the networks could be ranked in the following order: GNG, GCS, MLP, and fuzzy ARTMAP. On the other hand, when there are linear boundaries between classes, fuzzy ARTMAP and MLP networks tend to perform better.

Included in this research are discussions on the topology finding properties of neural gas networks and their ability to automatically construct radial basis function networks [2,3,12]. The basis for the neural gas network is the traditional Self-Organizing Map (SOM) originally proposed by Kohonen [4]. However, it differs from the SOM in the way in which the neural network topology is established. While in a SOM each neuron's neighborhood is fixed, in a GNG both the number of neurons and their neighborhood change dynamically. This produces networks capable of fitting any distribution of the training data with great ease and accuracy. Yet, the resulting mechanism is not biologically feasible and the method requires setting a number of arbitrary parameters, which makes its effective use difficult.

This paper proposes a form of hybrid of the standard SOM and GNG networks. This is accomplished by taking

†Authors are with the School of Electrical Engineering and Computer Science, Ohio University, Athens, OH, USA. (email: James.Graham.1@ohio.edu and starzyk@bobcat.ent.ohiou.edu.
This work was supported in part by a grant form AFOSR

978-1-4244-1821-3/08/$25.00©2008 IEEE

the general structure of the SOM and adding properties of the neural gas network. Specifically, the ability to insert nodes/neurons where needed in the SOM. In Fritzke's neural gas network, the algorithm grows the network every *x* cycles by inserting a new node between the two nodes with the greatest error. The new algorithm proposed in this work operates in a similar manner. However, rather than adding a new neuron, it takes one from the region of least error, where it is least needed. This may have the effect of balancing the network and potentially reducing the training time. Furthermore, the proposed network adapts a less parameter dependent model similar to the one created by Berglund [13], although utilizing slightly different methods to achieve the lack of parameters.

The next section presents Fritzke's algorithm on which the proposed algorithm is based. This is followed by section three which presents the proposed hybrid algorithm and compares it to Fritzke's algorithm in detail. Further comparisons to existing algorithms are presented in section four, while section five presents testing results for the proposed algorithm. The remaining sections consist of a short discussion of future work and conclusions.

II. Background

Fritzke's model differs from the standard neural gas network largely because it grows itself. His algorithm produced good results and served as a basis for the neural gas algorithm proposed in this work. It was chosen due to its ability to dynamically adjust itself and the relative simplicity of its network structure. In general, the GNG is a useful network type because it can easily depict topological relations, and form separate clusters of data as needed. Furthermore, insertion criterion can be arbitrarily chosen, allowing for use with supervised learning applications.

To make comparisons between the two methods easier, Fritzke's algorithm is described first. In his paper [2], Fritzke described the growing neural gas method detailed as follows:

Fritzke's GNG Algorithm

1) Start with a set A of two units a and b at random positions w_a and w_b in R^n:
   $A = \{(a,b)\}$.
2) Generate an input signal ξ according to $P(\xi)$ (the probability density function of the training data).
3) Determine s_1 and s_2 (s_1, $s_2 \in A$), such that they are the two nearest neighbors to ξ.
4) If it does not already exist, insert a connection between s_1 and s_2. Regardless, set the age of the connection between the two to zero.
5) Determine the error delta for s_1 using:
   $\Delta E_{s_1} = \left\| w_{s_1} - \xi \right\|^2$
6) Move s_1 and its direct topological neighbors toward ξ by fractions ε_b and ε_b, with regard to the total distance.
   $\Delta w_{s_1} = \varepsilon_b \left(\xi - w_{s_1} \right)$
   $\Delta w_i = \varepsilon_n \left(\xi - w_i \right)$
7) Increment the age of all edges emanating from s_1
8) Remove edges with age larger than a_{max}. If this results in units having no edges, remove them as well.
9) If λ signal generation iterations have occurred execute this step:
   a. Determine the unit *q* with the maximum error.
   b. Place a new unit *r* half way between *q* and its nearest neighbor *f*.
   c. Decrease the error of *q* and *f*.
      $\Delta E_{q,f} = -\alpha E_{q,f}$
   d. Determine the error of *r* with:
      $E_r = 0.5 \left(E_q + E_f \right)$
10) Decrease the error variable of all units: $\Delta E_c = -\beta E_c$
11) If the stopping criterion is not yet fulfilled (number of iterations or some other performance measure) return to step 2.

Such an algorithm produces a network of neurons distributed over the field of training data, and has similar properties to a SOM network. Namely, such a network spreads its neurons over the areas that contain training data which facilitates data clustering, labeling, and classification tasks.

III. Hybrid Self-organizing NN Algorithm

Fritzke's algorithm was carefully examined and several potentially undesirable features were noted. The following algorithm was created as an attempt to address the aforementioned undesirable features. The actual changes to Fritzke's approach and the reasons behind them are discussed following the algorithm.

Proposed Hybrid Algorithm

1) Generate a randomly positioned & connected network of neurons of size λ, with η>=1 connections per neuron. Set all edge ages to 1. Set all node usage values to 0. Usage indicators keep track of when a node was last used.
2) Generate an input signal ξ according to $P(\xi)$.
3) Determine s_1 and s_2 (s_1, $s_2 \in A$) such that they are the two nearest neighbors to ξ.
4) If it does not already exist, insert a connection between s_1 and s_2. Regardless, increment the age

of the connection between the two, and set the usage indicator the current cycle. (If a new connection the age is one.)

a. If one of s_1's connections is more than twice the mean distance of the two shortest connections to s_1 then remove it.
b. If removing the long connection orphaned it, relocate it by moving it to the area with max error by executing steps 11-15.

5) Compute the relative force acting on all topological neighbors of s_1, N_{s1}.

$$F_i = \frac{1/\|w_i - \xi\|^2}{\sum_{N_{s_1}} 1/\|w_k - \xi\|^2}$$

The force is proportional to the weighted square of a distance between the data point and coordinates of the network neuron represented by the weight vector.

6) Move the neighborhood N_{s1} toward ξ by adjusting their position as: $\Delta w_i = \frac{F_i}{\log\left((mean(age(i))/2+1\right)}(\xi - w_i)$, where mean(age(i)) is the average of all edge ages connected to i.

7) Increase the total error for each node in the neighborhood by $\Delta E_i = F_i \|w_{s_1} - \xi\|^2$

8) For the neighborhood of s_1 Increment all edge ages, and update node usage indicators to the current cycle number, where usage indicator refers to the cycle in which a node was lasted updated.

9) Check all neurons' usage indicators. If a neuron's indicator has not been updated for u_{max} cycles then relocate it using step 10.

10) Every λ iterations of signal generation execute the following steps.

a. Remove unit s_k with the minimum error and adjust the errors of its topological neighbors by adding $\Delta E_i = E_{s_k} F_i$, where F_i is the relative force for the neighborhood.
b. Insert s_k near the neighborhood of the unit with maximum error, s_q using $w_r = \sum_{i \in N_{s_q}} \frac{\sqrt{E_i} w_i}{\sqrt{E_i}}$. Connect s_k *to* s_q (for s_k set usage indicator u_k to current cycle and age to 1)
c. Find pair wise distance between r,q, and N_q and remove the longest edge.
d. Compute relative force F_i where i is an element of N_k and compute error $E_k = \sum_{i \in N_k} F_i E_i / 2$, then adjust errors of the neighborhood by: $\Delta E_i = -\frac{E_i}{2} F_i$

11) Repeat the cycle, or return to the previous location

Discussion

The main differences between the two methods are:

1) The hybrid method uses a pre-generated network of a fixed size. This adheres to biologically observed development of the brain where learning results in modification of existing connections and pruning of connections over the fixed size network of neurons. Fig. 6 illustrates such reduction of connections taking place in the proposed algorithm.
2) Connections get "stiffer" with age in the hybrid method, making their weight harder to change.
3) Error is calculated after the node position updates rather than before.
4) Weight adjustment and error distribution is based on "Force" calculations.
5) Connections are removed only under the following conditions (instead of all the time).
   a. When a connection is added, and there is another connection of length 2x greater than the average connection length for the node.
   b. When a node is moved and the number of connections on the moved node is >1 (after attaching to its destination node) its longest connection is removed.

Although Fritzke's algorithm works exceptionally well and is computationally efficient, it has several features that are not particularly supportive of biologically based neuron learning. This includes the way the network is grown by inserting neurons and the conditions for breaking connections in the existing network. Specifically, in Fritzke's algorithm a neuron is removed if all of its edges have an age larger than a predefined maximum value (Fritzke-8). However, this happens when a node is used with high frequency by being close to many data points. We consider this unnatural, as connections in biological neurons frequently get stronger with use and only the neurons that are not used tend to loose their connections and die (be moved).

In the proposed method edges with larger age are less plastic, resisting a change of position (weight). The age of an edge is always increasing when a neuron's weight is adjusted (Hybrid-8), while in Fritzke's algorithm the age value is reset to zero each time associated neurons are the two nearest neighbors to a training data point (Fritzke-4). Another non-natural feature in Fritzke's algorithm is the unlimited growth of neurons in the network (Fritzke-9b).

While the algorithm (and the corresponding network growth) can be stopped if user specified criteria are matched (for instance the maximum number of iterations or the average error level), it is not always desirable to halt the network growth and learning at the same time.

However, in a biological network, resources for learning, expressed through the number of neurons, are pre-specified. Therefore, it is more desirable to find an algorithm that minimizes the average matching error by adjusting the interconnection strengths of the given set of neurons, rather than adding neurons every λ cycles of the algorithm. The proposed method implements this natural approach within a given network of neurons.

In Fritzke's method a new node is placed halfway between a node with the maximum error and its nearest neighbor (Fritzke-9d). This is done regardless of the amount of error in the general neighborhood of the node with maximum error. In particular, the nearest neighbor's error may not be significant so the reduction of its error value, as is done in Fritzke's method, would not be very useful. Other neighbors with a more significant error would benefit more from such reduction. Thus, in the proposed method the neuron being moved is placed in a location determined by a weighted average of the neighborhood's neurons and weights are determined by the amount of error in the neighboring neurons (Hybrid-10).

Finally, another aspect of the Fritzke's method that was not found to be particularly appealing is the number of parameters that need to be predefined, and somewhat arbitrary decisions that are made about the amount of error reduction. More specifically, the weight adjustment in the nearest neighbor is proportional to ε_b (Fritzke-6) with weights of its neighbors adjusted in proportion to ε_n; both of which are arbitrary. In the proposed method, weight adjustment in the neighborhood with the largest error is computed in proportion to the relative force acting on each node (Hybrid-5) and in inverse proportion to the age of the edges connecting the node being adjusted within the neighborhood (Hybrid-6). This weight adjustment seems to be naturally linked to "forces" that act on each neuron in the proposed algorithm, where the relative force is greater whenever the distance to the neighbor is smaller. In addition, the larger is the age of an edge, the smaller is its weight adjustment. Such a natural dependence on the number of times that a given edge was adjusted (reflected by its age) is in direct correspondence to edge stiffening in biological neurons that characterize mature (trained) connections. Mature biological links are less likely to change. This is in a striking contrast to Fritzke's algorithm where the "old" edges are removed first.

Similar arbitrarily set parameters are used in Fritzke's algorithm to adjust the amount of error in the two neurons closest to a data point (α) and amount of error in the inserted neuron (Fritzke-9). In the proposed method, error in the inserted node is computed based on the weighted average of the errors in the neighborhood and the amount of error in this node equals to the sum of the reduction in error in the nodes of the associated neighborhood (Hybrid-10d). In addition, Fritzke uses automatic reduction of errors in all nodes in proportion to yet another arbitrary factor β (Fritzke-10). No such reduction is used in the proposed method.

As previously mentioned the ability of the Hybrid algorithm to move neurons from one location to another, as needed, is a significant factor in its ability to adjust itself. This feature was taken from the GNG algorithm and adjusted to work in a non-growing network. For example, consider a network of 50 nodes. According to the hybrid algorithm, a network of 50 nodes will be examined every 50 iterations to find the nodes with the most and the least amount of accumulated error. To help illustrate this function of the algorithm, we ran the algorithm with 50 nodes and data structure as described in section 5. After the first 50 iterations the maximum error was 1.4481 for node 9 and 0 for node 6. The error of zero at this point reflects how young the network is (at the beginning of the training cycle). Node 6 has likely not been used yet. Normally, node 6 would have its error distributed to its neighbors, however, since it has no error it is moved into position around node 9 according to the error distribution of the node 9 and its neighborhood (Hybrid-10b). Next, node 9 takes on a portion of the error of each node in node 9's neighbor hood to reduce the error of the region (Hybrid-10d). Finally, node 6's old neighborhood connections are removed and it received a single connection to node 9. Node 6 will gain additional neighborhood connections later, as needed. This final step completes the moving process and the algorithm resumes its normal adjustment iterations.

IV. Comparison vs. Other Kohonen based networks

So far the proposed algorithm has only been considered in comparison to Fritke's GNG algorithm. However, to properly grasp its significance, it is necessary to compare this algorithm to other variants of the Self-Organizing Map. The following sections present a brief discussion of the pros and cons of various SOM based networks vs. this work.

A. Traditional SOM

The traditional SOM as produced by Kohonen [14] lacks all the modifications presented in the proposed algorithm as well as a multitude of others that have been derived from it. However, it retains one primary advantage, its overall simplicity. The standard SOM is simply a network of nodes that have multidimensional shared inputs whose weights are adjusted based on neighborhood proximity. In other words, nodes are chosen by proximity to an input vector and the

"neighborhood" surrounding said node is updated according to the algorithm used. In the traditional SOM, the neighborhood is usually defined by a function (possibly Gaussian). Weight adjustment within the neighborhood is handled by a time decreasing constant. There are several disadvantages of the traditional SOM which the proposed algorithm and numerous other variants of the SOM attempt to address. Specifically, nodes within the traditional SOM are incapable of moving outside the predefined lattice; the lattice is static and cannot create or remove connections. In addition, it is necessary to define parameters for the algorithm to operate. Furthermore, many SOM algorithms are also divided into train/test operations, often requiring a significant number of iterations to train the algorithm before evaluating the results. However, algorithms such as the proposed algorithm and Fritzke's GNG algorithm do not have this issue. Training occurs continuously, even during the testing phase.

B. Parameter-less SOM (PLSOM)

The parameter-less SOM is a variant on the standard SOM that eliminates several of the traditional parameters, specifically the ones associated with adjustments to weight and error. One example of the PLSOM is the work by Berglund [12]. Like the model presented in this paper, the only actual parameter used was the one associated with the network size. According to Berglund, the PLSOM's main advantage over the traditional SOM is its greater convergence rate. Tests within his paper show that Berglund's implementation of the PLSOM outperforms a Matlab implementation of the SOM in both adaptation time and accuracy. While the proposed algorithm and the PLSOM both share a predefined network size and a lack of weight and error parameters, there are also significant differences between the two. One such difference is that the Hybrid method actually picks up and moves nodes from one region to another; something that can have a major effect on the convergence of the network. Other differences are in the form of the error calculation and how neighborhoods are treated. The PLSOM treats neighborhoods in the standard SOM way, that is, using a neighborhood function to determine how nodes are updated about an input point. This is in contrast to both the Hybrid and Fritzke's methods which determine the neighborhood by generating connections between the two nearest neighbors to an input point. Thus, to move a neighborhood, the node closest to the input is selected and its neighbors are adjusted accordingly. In the case of the proposed hybrid algorithm the adjustment is based on a force function related to the neighborhoods' neurons distance from the input data.

C. Neural Gas

The neural gas algorithm is similar to the standard SOM, however, it lacks the rigid neighborhood connectivity present in the SOM. It was introduced by Matrinetz and Schulten [1], and differs from the SOM in that it lacks specific neighborhood connections altogether and computes neighbors on the fly in a method similar to the neighborhood function used by the traditional SOM. The neural gas network is named as such because of the gaseous way it expands to cover regions. Neural Gas (NG) algorithms have an advantage over the traditional SOM in that they can form clusters of nodes. Suppose that instead of a single rectangular region that is used in most SOM examples, one takes two regions separated by a space. The neural gas based algorithms are able to handle this type of problem with no difficulty; they simply divide into separate clusters. However, both the SOM and PLSOM algorithms have no mechanism for separating their grid space, meaning that edges and sometimes nodes exist in the "no-mans" land between valid input space regions. The proposed hybrid algorithm and the NG do not suffer from such a problem.

D. Growing Neural Gas

The Growing Neural Gas algorithm is based on the Neural Gas algorithm and differs in that nodes are added every λ input cycles rather than starting with a predefined number of randomly placed nodes,. Most of the work on GNG networks can be attributed to Fritzke [2,3,4]. The pros and cons of the GNG algorithm have already been discussed with regards to Fritzke's algorithm in Section 3. There also exist some similar algorithms, as in Lang and Warrick's work [15], which presented a Plastic SOM algorithm based on Fritzke's GNG algorithm. The main thrust of this algorithm was that it retains it's plasticity indefinitely and does not allow aging to affect network adjustment. It also tries to recognize different data sources and provide separate neuron clusters. Another example of work using the GNG algorithm as a basis was produced by Qin and Suganthan [16]. They created a robust algorithm based on the GNG but implemented several improvements, including an outlier resistant scheme, adaptive modulation of learning rates, and a cluster repulsion method.

V. Algorithm Testing Results

To analyze the performance of the proposed algorithm we tested against the performance of Fritzke's algorithm. Additionally, some of the traits of the proposed algorithm are examined in this section. The network organization is its most importance facet. It is important that the proposed algorithm perform as well as Fritzke's algorithm. We also tested the hybrid algorithm with several different initial connectivity settings and network sizes. In the following paragraphs both the minimum and maximum initial connectivities tested are also examined.

Figure 1 shows the organization that results from running Fritzke's algorithm for 40,000 iterations with the following constants: ε_b=0.05, ε_n=0.0006 , a_{max}=88, λ=200, α=.5, β=0.0005. Figure 2 shows the initial

structure of the hybrid model before training with 1 preset connection. Figures 3-4 show the connectivity networks resulting from running the algorithm for 40,000 iterations with 1 and 16 preset connections. Figures 5-6 give an idea of how the connectivity evolves between the nodes as the number of iterations progresses for the same set of preset connection numbers. As can be seen from the two figures, the neighborhood connectivity will eventually reach equilibrium at around 1,600 connections for the given conditions. A surprising property of the proposed algorithm is that the connectivity equilibrium (1,600 total connections for the presented example) is largely independent of the initial network connectivity. The equilibrium is largely dependant on the size of the network and the maximum connection age parameter of the network. In the instances shown, the connectivity approaches 8 connections per node.

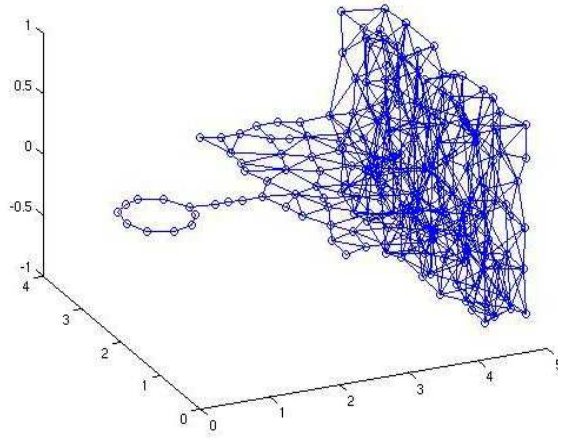

Fig. 1. Connectivity network from running Fritzke's growing neural gas algorithm

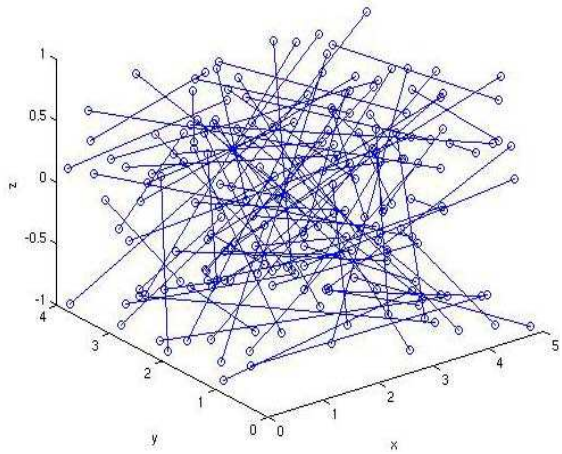

Fig. 2. Initial connectivity for untrained network with 1 random preset connection.

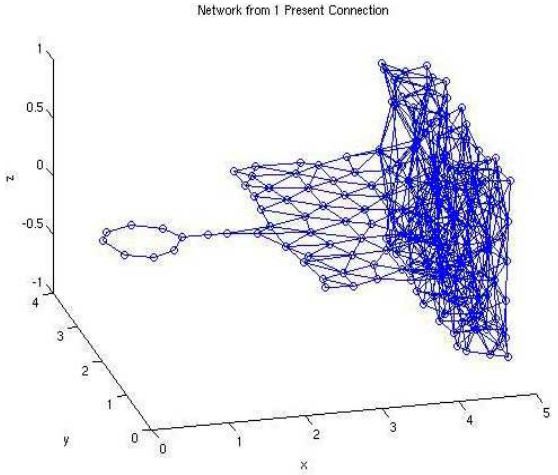

Fig. 3. Connectivity network resulting from 1 preset connection between nodes.

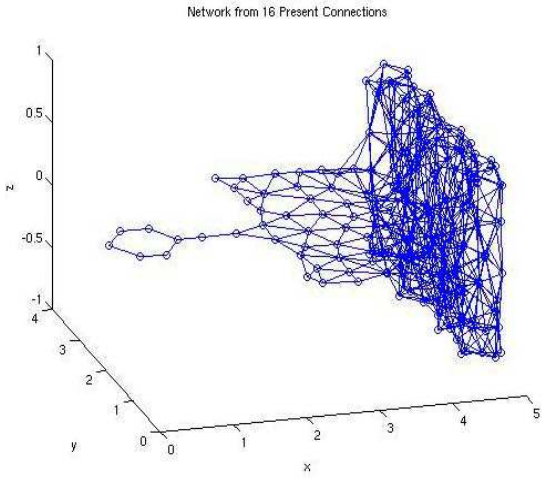

Fig. 4. Connectivity network resulting from 16 preset connections between nodes.

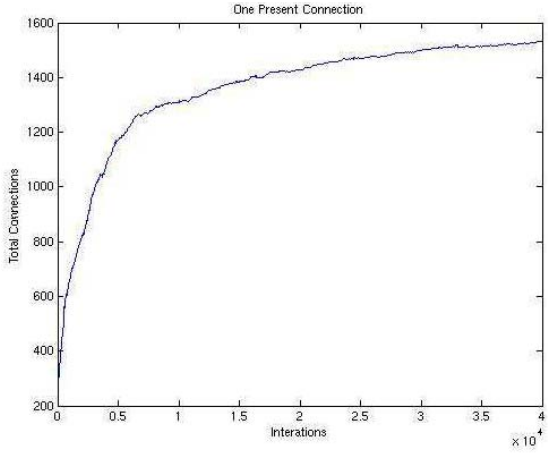

Fig. 5. The number of connections resulting from 1 preset connection between nodes.

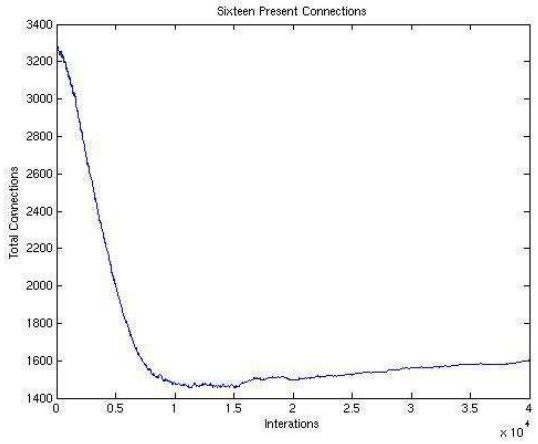

Fig. 6. The number of connections resulting from 16 preset connections between nodes.

Figure 7 presents another comparison of both Fritzke's and the Hybrid algorithm in a 2-D input space. It also includes a comparison with the standard SOM network. The Fritzke and Hybrid networks in Figure 7 use the same settings as in the earlier example, while the SOM network is the same size as the other two and ran on the same data for 25 epochs with the rest of its settings being the default for Matlab. Figures 1 and 3 and the comparison between the Fritzke and Hybrid networks in Figure 7 are sufficiently alike that we can say that the proposed Hybrid algorithm can create results of equivalent quality to Fritzke's algorithm. On the downside, however, the proposed algorithm is slower as a result of the greater number of computations needed to handle the parameterless calculations and because Fritzke's algorithm has significantly fewer nodes to account for, for a large portion of its run-time. Both algorithms, however, do a better job at covering the input region and do it faster than the basic SOM algorithm, even disregarding the nodes bridging the gaps.

VI. Further Work

Attempts are currently being made to adapt a form of the algorithm presented in this paper to a hierarchical network capable of feature extraction and shape recognition. It is believed that some of the features used in the proposed algorithm such as the GNG based node insertion, use of neighborhoods, force based weight and error adjustment, and other biologically plausible features could prove useful. Some optimization is needed, since it is believed that using cell array structures (Matlab data structure) is not as efficient as using regular matrices, despite making it easier to keep track of individual nodes. The eventual goal is to have a hierarchal network structured in an upside down pyramid structure, meaning subsequent layers have upwards of twice the number of nodes as those below them. Individual nodes will not have global input connectivity,

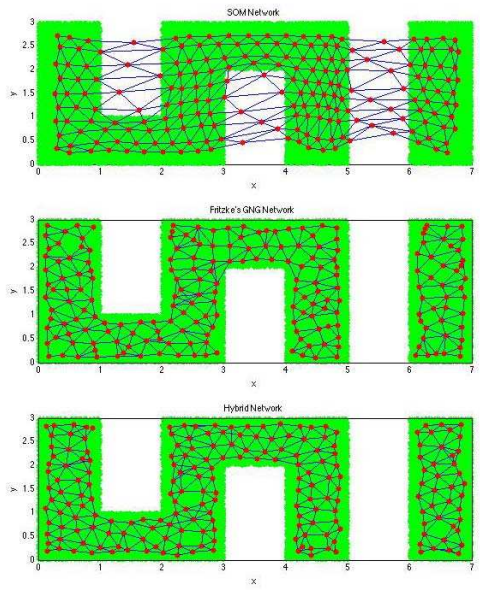

Fig. 7. 2-D comparison of SOM, Fritzke's GNG, and the Hybrid algorithm.

but rather local input connections based on the input data correlation.

VII. Conclusions

In the course of this work, Fritzke's growing neural gas algorithm was examined and altered into what is believed to be a more biologically plausible design. The changes make the new hybrid algorithm more similar to a standard self-organizing map algorithm rather than a neural gas algorithm. However, the presence of moving nodes allows for behavior very similar of that shown by Fritke's algorithm as can be seen from Figures 1, 3, and 4. The hybrid algorithm retains most of the advantages of the GNG while adapting a reduced number of parameters and more biologically plausible design. To be specific, it retains the ability to place nodes where needed, as does the GNG algorithm, without actually having to introduce new nodes. Also, by removing the weight and error adjusting parameters, the guesswork, required to determine the parameters in the first place, is eliminated. However, additional work is required to accurately characterize the proposed algorithm in comparison to the other algorithm types and variations mentioned herein. While the hybrid algorithm performs admirably in terms of the quality of results when compared to Fritzke's algorithm, it is slower and an actual quantifiable comparison has yet to be performed.

References

[1] T. M. Matrinetz and K.J. Schulten, "A 'neural-gas' network learns topologies," in T. Kohonen, K. Mäkisara, O. Simula, and J. Kangas editors, *Artificial Neural Networks*, North-Holland, Amsterdam, 1991, pp. 397-402.

[2] B. Fritzke, "A Growing Neural Gas Network Learns Topologies," *Advances in Neural Information Processing Systems 7*, G. Tesauro, D.S. Toretzky, and T.K. Leen, (eds.), MIT Press, Cambridge MA, 1995. .

[3] B. Fritzke, "Unsupervised ontogenetic networks", *Handbook of Neural Computation*, C2.4, IOP Publishing Ltd, 1997.

[4] T. Kohonen, "The Self-Organizing map", Proc. of IEEE, 78:1464-1480, 1990.

[5] M.A. Montes de Oca, L. Garrido, and J.L. Aguirre, "An hybridization of an ant-based clustering algorithm with growing neural gas networks for classification tasks," in Proc. of 2005 ACM Symposium on Applied Computing, Sante Fe, New Mexico, 13-17 March 2005, pp. 9-13.

[6] J.G. Rodriguez, A. Angelopoulou, and A. Psarrou, "Growing Neural Gas (GNG): A Soft Competitive Learning Method for *2D* Hand Modeling," IEICE Trans. INF. & SYST., Vol. E89–D(7), 2000, pp. 2124-2131.

[7] G. Cheng and A. Zell, "Double Growing Neural Gas for Disease Diagnosis," in Proc. of the Artificial Neural Networks in Medicine and Biology Conference (ANNIMAB-1), Goteborg, Sweden, 13-16 May 2000, Vol. 5, pp. 309-314.

[8] S. Rovetta and R. Zunino, "Efficient Training of Neural Gas Vector Quantizers with Analog Circuit Implementation," IEEE Trans. on Circuits and Systems-II: Analog and Digital Signal Processing, Vol. 46(6), 1999, pp. 688-698.

[9] C. Rehtanz and C. Leder, "Stability Assessment of Electric Power Systems using Growing Neural Gas and Self-Organizing Maps," in Proc of European Symposium on Artificial Neural Networks, Bruges, Belgium, 26-28 April 2000, pp. 401-406.

[10] A.S. Atukorale and P. N. Suganthan, "Hierarchical overlapped Neural-Gas network with application to pattern classification", Neurocomputing, November 2000.

[11] D. Heinke and F. H. Hamker, "Comparing Neural Networks: A Benchmark on Growing Neural Gas, Growing Cell Structures, and Fuzzy ARTMAP," IEEE Trans. On Neural Networks, Vol. 9(6), 1998, pp. 1279-1291.

[12] B. Fritzke, "Automatic construction of radial basis function networks with the growing neural gas model and its relevance for fuzzy logic," in Proc. of the 1996 ACM symposium on Applied Computing table of contents Philadelphia, Pennsylvania, 1996, pp. 624 – 627.

[13] E. Berglund and J. Sitte, "The Parameter-Less Self-Organizing Map algorithm," ITEE Trans. On Neural Networks, vol. 17, no. 2, pp. 305-316, March 2006.

[14] T. Kohonen, *Self-Organization and Associative Memory*, ser. Springer Series in Information Sciences. Berlin Heidelberg: Springer, 1984, vol. 8, 3rd ed. 1989.

[15] R. Lang and K. Warwick, "The plastic self organizing map," in Proc. Of the 2002 Int. Joint Conf. on Neural Networks, vol. 1, 2002, pp. 727-732.

[16] A. K. Qin and P. N. Suganthan, "Robust growing neural gas algorithm with application in cluster analysis", Neural Networks, Vol. 17, No. 8-9, 2004, pp. 1135-1148.