Approximation of Geometric Structures with Growing Cell Structures and Growing Neural Gas
A Performance Comparison

Hendrik Annuth and Christian-A. Bohn
Computer Graphics & Virtual Reality, Wedel University of Applied Sciences, Feldstr. 143, Wedel, Germany

Keywords: Growing Cell Structures, Growing Neural Gas, Smart Growing Cells, Surface Reconstruction.

Abstract: We compare Growing Cell Structures and Growing Neural Gas, which were introduced by Bernd Fritzke and which are famous for their facilities in classification, clustering, dimensionality reduction, data visualization, and approximation tasks. We practically test and analyze their capabilities in geometric approximation and focusing on the application of surface reconstruction from 3D point-data. Our focus is to work out the differences of the algorithms that are especially relevant concerning approximation purposes. We address the issue of suitable input data, their applied graphs, their topological properties, their run time complexities and we present a summary of suggested alternations to both approaches and evaluate our results.

1 INTRODUCTION

The approximation of a geometric structure $\mathcal{M}$ is usually based on measured data or data received from an analytically unknown function. The input data is a point set $\mathcal{P} = \{p_1 ... p_n\}$ with a certain dimension $n_\mathcal{P}$ which also defines the dimension of the target space of the approximation. Therefore $\mathcal{P} = \{p_1 ... p_n\} \subset \mathbb{R}^{n_\mathcal{P}}$. The dimension $n_\mathcal{M}$ of the resulting structure $\mathcal{M}$ usually is a subspace of $n_\mathcal{P}$. For surface reconstruction e.g. we have 3D Points $\mathcal{P} \subset \mathbb{R}^{n_\mathcal{P}=3}$ and our resulting structure is a 2D surface so $n_\mathcal{M} = 2$. With this definition given, a geometric approximation reconstructs a structure $\mathcal{M}$ of a certain dimension $n_\mathcal{M}$ greater zero from a set of points $\mathcal{P}$ contained in an $n_\mathcal{P}$ dimensional target space. With the steady development in scanning technologies the amount of data, the areas of application, and the related challenges constantly rises. The most common problems are noise, none uniform sample densities, holes in the sample data, and discontinuities of the tangent space like corners or creases which cause a huge demand for algorithms that can handle such problems.

2 PREVIOUS WORK

As Geometric approximations demand stability and robustness due to the presence of noisy data, many neural computation techniques have been applied to the problem as well. One of the most important works in that area of classification and clustering is (MacQueen, 1967). In the process k $n_\mathcal{P}$-dimensional reference vectors are placed in the input samples $\mathcal{P}$ such that they are means of those samples. When $\mathcal{M}$ is a graph $\mathcal{M} = \{v_1 ... v_n\}$ and $\mathcal{C} = \{c_1 ... c_n\} \wedge \mathcal{C} \subset \mathcal{M} \times \mathcal{M}$ where v is a reference vector and c a connection between two reference vectors. Kohonen's *self-organizing-map* (SOM) (Kohonen, 1982) and *Neural Gas* (Martinetz and Schulten, 1991; Martinetz and Schulten, 1994) are concepts that introduce $\mathcal{M}$ is as a graph. Since these methods produce topologies, they successfully can be used for reconstruction purposes. In (Barhak and Fischer, 2002; Yu, 1999; Hoffmann and Vrady, 1998) a SOM and in (Melato et al., 2008) the neural gas approach is used for surface reconstruction. As the resolution of the reference vectors is fixed the results strongly depend on the initial positions of reference vectors and the number of required reference vectors to properly represent the underlying geometry needs to be known in advance. Fritzke's *Growing Cell Structures* (GCS) approach (Fritzke, 1993) tracks the approximation error and adds reference vectors in areas of big approximation error and terminates when a certain error is reached. Thus the process does not require a previous choice of resolution. *Growing neural gas* (GNG) (Fritzke, 1995) also tracks the approximation error to add new vectors and also does not need a predetermined number of refer-

Annuth H. and Bohn C..
Approximation of Geometric Structures with Growing Cell Structures and Growing Neural Gas - A Performance Comparison.
DOI: 10.5220/0004157405520557
In *Proceedings of the 4th International Joint Conference on Computational Intelligence* (NCTA-2012), pages 552-557
ISBN: 978-989-8565-33-4

ence vectors. To establish a topology it uses competitive Hebbian learning. Both methods, GCS and GNG, have been applied to surface reconstruction successfully. (Vrady et al., 1999) suggested using GCS for surface reconstruction the first time. Ivrissimtzis presents modifications (Ivrissimtzis et al., 2003b) for mesh smoothing and for adding and deleting reference vectors with improved connectivity properties. With the *Smart Growing Cells* approach (SGC) (Annuth and Bohn, 2012) the GCS are enhanced with locally individual behavior to, first, account for surface curvature through the granularity of the underlying topology of the graph. Second, complex homeomorphisms can be reconstructed, and third, tangential discontinuities like corners and creases are reconstructed correctly. In (Holdstein and Fischer, 2008; Do Rêgo et al., 2010; Melato et al., 2008) GNG was used for surface reconstruction while in (do Rego et al., 2007) a hybrid approach using both, GCS and GNG, for surface reconstruction is presented.

3 PERFORMANCE COMPARISON

In this section we analyze similarities and differences of both approaches, and compare their strengths and weaknesses in the area of geometric approximation. We use surface reconstruction in our examples since it is a very common type of application of these algorithms.

3.1 Similarities of GCS and GNG

GCS and GNG both introduce a stochastic approach to the problem of geometric approximation. The sample points in $\mathcal{P}$ are accessed in a random series. This gives the method a big advantage. At any given time during the iterations only one data sample of $\mathcal{P}$ needs to be loaded. Therefore such approaches can process as many measured points as a modern hard drive is able to contain. Since both algorithms are descents from classical k-means clustering, they are both very strong against noisy data (see Fig. 1). Their algorithms are fairly simple and easy to understand. This makes these algorithms maintainable and as both concepts are actually based on data analysis they can robustly cope with any given point constellation $\mathcal{P}$. GCS and GNG are also very flexible. While $\mathcal{P}$ is processed, points could be added or deleted from $\mathcal{P}$, or regions of special interest in $\mathcal{P}$ could be set to a higher likelihood within the random sample selection, which would create a higher resolution in those areas (Holdstein and Fischer, 2008). The processes can also be stopped or continued at any given time during the iterations. On the downside both create $\mathcal{M}$ as a mesh and not as differentiable smooth approximation. Fritzke presented a GNG concept that uses radial basis functions (Fritzke, 1996), but this concept has not jet been used for geometric approximation purposes.

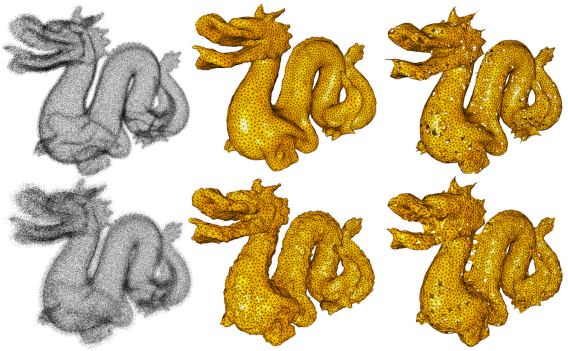

Figure 1: Both algorithms have a natural ability to deal with noise data. Point Cloud (left), GCS (middle), GNG (right), 2% Gaussian noise (top) and 4% Gaussian noise (bottom).

3.2 Sample Points $\mathcal{P}$

The approximation of a geometric structures from $\mathcal{P}$ basically exposes two kinds of problems. First, not all points in $\mathcal{P}$ might interpolate $\mathcal{M}$ due to noise or outliers. As mentioned above section 3.1 this kind of problems can be satisfactory handled by both algorithms. Second, the amount of points that is needed for an algorithm to create a sound surface — here GCS and GNG differ. The basis of existence for a structure in $\mathcal{M}$ and its associated topology $\mathcal{C}$ is carried at different places in the algorithms. In GCS the signal counters of a reference vector determines whether a reference vectors is deleted or not. In GNG the deletion is determined by the age of the connections between reference vectors. In order for both structures to remain they need to be hit by a sample point. As $|\mathcal{M}|$ is way smaller then $|\mathcal{C}|$, a GNG Graph simply needs more points to create a equivalent approximation (see Fig. 2). In an ideal situation for $\mathcal{M}$ every reference vectors has six connections. This is ideal, because every triangle can potentially be an equilateral triangle in such a constellation and the reference vectors are as evenly distributed as possible, which is desirable for most processes in computer graphics.
If every connection in $\mathcal{C}$ has two triangles on both sides, which means the surface has no boundaries, we can calculate the required points for both algorithms in an ideal case. For GCS any signal counter is associated with one reference vector, in order to get a signal at some time, $|\mathcal{M}|$ should not exceed the amount of points $|\mathcal{P}|$. For GNG in an ideal scenario were any

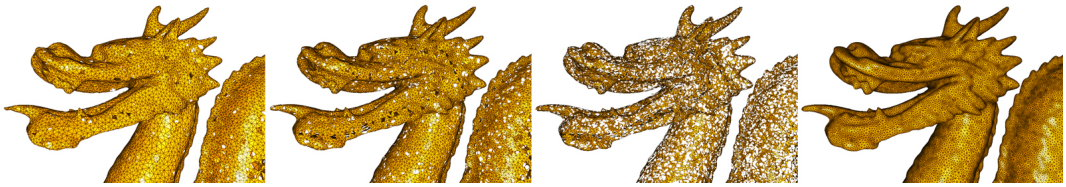

Figure 2: When the number of points in the GNG process in not sufficient to hit the connection in the graph, the structure starts to dissolve. From left to right approximations with different numbers of reference vectors: GNG Dragon 30K, GNG Dragon 60K, GNG Dragon 120K and GCS Dragon 120K.

vertex has six connections, which are shared by two reference vectors $|\mathcal{C}| = 3 \cdot |\mathcal{M}|$, so $|\mathcal{M}|$ can only be a third of $|\mathcal{P}|$ in an ideal scenario. In (Holdstein and Fischer, 2008) and in our tests we discovered by practical testing that the best results are created at 5% to 10% of $|\mathcal{P}|$.

3.3 Graph $\mathcal{M}$

For GCS and GNG $\mathcal{M}$ is a specific graph. GCS is only allowed to include simplices of the previous chosen dimension $n_{\mathcal{M}}$, while $\mathcal{M}$ in the case of GNG can contain any kind of simplex that does not exceed the dimension $n_{\mathcal{P}}$. If $n_{\mathcal{M}}$ is unknown or even needs to be a set of values, GCS is simply impractical, but in case of geometric approximation this is seldom if ever the case. The problem is usually the other way round. So $n_{\mathcal{M}}$ is known, in case e.g. of surface reconstruction it is 2D, but in GNG $\mathcal{M}$ is not guaranteed to consist of triangles. Therefore a post process needs to delete or transform all structures in $\mathcal{M}$ with a different dimension then 2D and those which infringe the criteria for a manifold. Which means that a connection $c \in \mathcal{C}$ can be connected to a maximum of two triangles. Since the process naturally creates lots of polygons with more then three sides, the process also needs to include a hole filling mechanism (see Fig. 3).

In our implementation we did not introduce such a method to give an impression of the result of the actual GNG algorithm, as such a method could be implemented in many different ways leading to very different results for $\mathcal{M}$. For our Figures we will show $\mathcal{M}$ as a graph of triangles, that do not have an orientation. Neither GCS nor GNG define an oriented surface in $\mathcal{M}$. In a real world computer graphics application however, a graph $\mathcal{M}$ needs to have oriented triangles. Even so this is not part of the actual GCS definition, all reconstruction algorithms we encountered in our research used a graph that already includes oriented triangles. As the surface is refined within the process, it is always build based on pre-existing surface. This way the orientation and gradient of a surface has a certain inertia, which enables the process to create sound

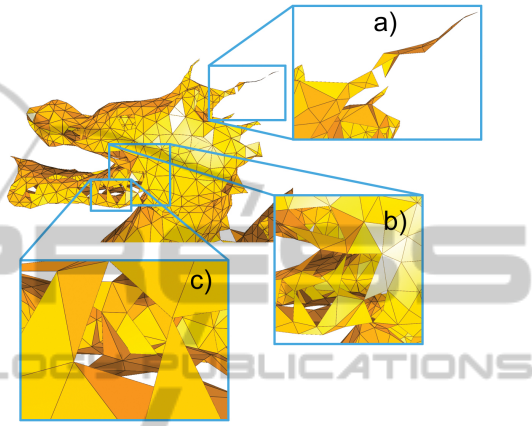

Figure 3: Different possible constellations in a GNG graph, like a none-manifold connection that is attached to three triangles a), many holes in the surface b) and connection that rather present a volume than a surface c).

surfaces even in very challenging areas of $\mathcal{P}$. If the surface is incorrect however this inertia might prevent surface from being corrected and an initial orientation, does not generally prevent the GCS process from creating twisted surface (see section 3.4).

Since $\mathcal{M}$ is not necessarily exclusively build of triangles in the GNG approach, $\mathcal{M}$ cannot be implemented as a graph consisting of oriented triangles throughout the process. Therefore refinements in $\mathcal{M}$ are not based on pre-existing surface. The topology generating method in the GNG process is not guided by previous surface stages. When the process is finished and the necessary cleaning mechanism has been performed on $\mathcal{M}$ the orientation of the encountered triangles need to be propagated across the graph, which can especially in noisy areas lead to twisted surface (see (Holdstein and Fischer, 2008)). As GCS uses a more sophisticated graph, a lot more information can be deduced within the process, depending the surface gradient or triangle properties like normal vectors. This extra information can be used to augment the algorithm (see section 3.6).

3.4 Topology $\mathcal{C}$

In the following section we will focus on the algo-

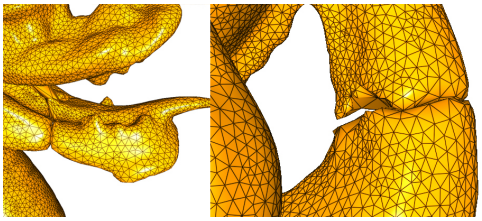

Figure 4: At the mouth of the Dragon the GCS process ran in a local minimum (left). The tail of the Dragon can not be approximated correctly, since the process was initialized with a tetrahedral, which has a different genus than the Dragon(right).

rithms topologies. In case of GNG every topology can be constructed, which is more a problem in the subsequent cleaning process. So without any modification GNG can actually produce any demanded topology. In case of GCS that is not the case. The surface can only create topologies equal to the initial topology. New added reference vectors always preserve the pre-existing surface (this can also be an advantage section 3.3). This creates two problems, first the surface can tangle up in a local minimum and second if $\mathcal{P}$ is based on a structure of different topology, it cannot be correctly build (see Fig. 4). To choose the demanded topology in advance, would not solve the problem, since the process would still run into local minima and for any approximation the topology would need to be known in advance. The standard GCS algorithm is practically unable to create approximations for topologies of different genus. This problem however has successfully been dealt with (see section 3.6).

3.5 Run Time Complexity

Without any aide of supporting algorithms the run time complex of both GCS and GNG is $O(n^2)$, where n dependents on the finial approximation resolution $|\mathcal{M}|$. The basic step that is performed depended on the approximation resolution $|\mathcal{M}|$ includes the search for the nearest neighbor in $\mathcal{M}$ to a given sample point, which naively implemented includes a distance check to all reference vectors in $\mathcal{M}$ ($O(n)$). This search can be improved by using an Octree, which reduces the complexity of this search to $O(log\,n)$. The basic step also contains decreasing the collected error of all reference vectors $O(n)$. And the adding and deletion process includes to find the highest or lowest error under all reference vectors $O(n)$. This has been addressed in (Annuth and Bohn, 2010b), where a data structure called Tumble-Tree reduces these complexities to $O(log\,n)$. So the overall run time complexity of both algorithms can be reduced to $O(n\,log\,n)$.

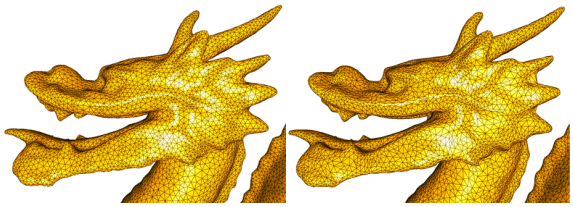

Figure 5: The GCS process with the smoothing operation (left) shows triangles that are closer to equilateral triangles, than the standard neighbor moving technique (right).

3.6 Modification Overview

In the following sections, we want to present some modifications that have been suggested to enhance both approaches.

GCS: For approximations of a certain size a constant error counter decreasing degree β is impractical, because all error counters start to converge to zero. Therefore (Ivrissimtzis et al., 2003a) introduce a dynamic β value. $\beta = 1 - \eta^{1/((\gamma=6)\cdot|\mathcal{M}|)}$ γ is the number of times a reference vector is allowed to be missed, before it falls under the deletion threshold η. In (Ivrissimtzis et al., 2003b) the moving of neighbors was exchanged for a smoothing mechanism presented by Taubin (Taubin, 1995). This increased the overall triangle quality, measured in equilateral triangles (see Fig. 5). In (Ivrissimtzis et al., 2003b) the way how new reference vectors are added and deleted is improved. The presented processes mostly create reference vectors with six connections which again creates more equilateral triangles. The GCS approximation creates a surface of simplices. In areas of curved surface a higher resolution of triangles is needed then in plane areas to reach the same approximation quality. This has been issued in (Jeong et al., 2003) where the change in normal directions has been tracked and to add more reference vectors in such area of much movement. In (Annuth and Bohn, 2010a) the global curvature is set in relation to the local curvature of a reference vector. If the local curvature is higher signals in such an area count more, which leads to more reference vectors in those areas. (Annuth and Bohn, 2012) presents a general concept of enhancing the GCS approach, the Smart Growing Cells (SGC). The concept introduces individual behavior for the reference vectors. That means that for different situations in the graph, the adaption process alters. The biggest challenge is the inability of GCS to model Surface of different genus (see Fig. 4). The presented process in (Ivrissimtzis et al., 2003b) had no mechanism to cut or coalesce the surface. In (Ivrissimtzis et al., 2003a) the cutting of surface was triggered by the size of triangles and then coalescing by a certain threshold for the Hausdorff distance of two bound-

aries. This method has two important disadvantages. First the triangle sizes that reliably indicates misplace surface takes quite a long time to appear and second in case of none uniform sample densities where triangle naturally have different sizes the system does not work. Excluding those sample distributions again would destroy a big advantage of the whole approach. In (Annuth and Bohn, 2010a) the cutting is triggered by a high number of connections $|\mathcal{N}_v|$ of a reference vector v. This mechanism detects misplaced surface a lot faster and also works for none uniform sample densities. The coalescing process happens dynamically when equally oriented boundaries come close to each other.

GNG. Both methods (Do Rêgo et al., 2010; Melato et al., 2008) introduced an operation which changes the amount of iterations λ before adding reference Vectors. In (Melato et al., 2008) with increasing size $|\mathcal{M}|$ they increase the size of λ. In (Do Rêgo et al., 2010) after the demanded size $|\mathcal{M}|$ was reached they introduced another learning phase, where only topology is learned, but no additional reference vectors were added. The reason for this operations is that the basic reference vector adding step creates holes, because it creates reference vectors that have two connections. A different approach to tackle this problem is presented in (do Rego et al., 2007), were they modified the basic adaption step and the adding step. They search for three instead of two closest points to a random sample p. By that they tried to insure that the process creates triangles rather than arbitrary dimensional structures. The adding of reference vectors was done according to the GCS method, for this $\mathcal{M}$ had to be an triangle based graph. The problem with this concept is that the connections that are made in the basic step are still arbitrary, therefore the approach still creates cross connections and overlapping triangles.

4 RESULT

In this paper we practically compared the performance of GCS and GNG and worked out differences and explained their meaning towards the approximation process. In the following we show results with different parameter modifications. To make them comparable we used the Stanford Dragon with a fixed resolution $|\mathcal{M}|$ of thirty-thousand reference vectors. The model has uneven point densities, some detail areas like the horns and a relatively challenging from. We measure point distance, valence, triangle quality and the run time. The following results have been calculated single threaded on a 2.53GHz CPU.

We measure the distance after root-mean-square (RMS). As the surface of GNG is not sealed, only points directly above or beneath triangle are included. We define the distance one as the diagonal of the bounding box of $\mathcal{P}$. The valence of a reference vector v is its number of it's connections $|\mathcal{N}_v|$, we measure the degree of reference vectors that have a valence from five to seven. We measure the triangle quality as the average closeness of a triangle to an equilateral triangle. We calculate the surface area A of an triangle t. Then we calculate the surface area A_2 of second triangle t_2. t_2 is an equilateral triangle with the side length of the longest side of t. Then we divide A by A_2. In our benchmark versions we used the parameters as presented in (Annuth and Bohn, 2010a) and for GNG as in (Fritzke, 1995).

Table 1: The table shows our results for the Stanford Dragon build of 30K reference vectors, under different circumstances.

	RMS $\cdot 10^4$	val	Tri Qual	time
Benchmark				
GCS	3.83	0.900	0.785	17.3sec
GNG	8.31	0.679	0.636	6.6sec
Neighbor Move vs Smooth				
GCS	5.48	0.861	0.610	13.3sec
Different # of smoothing rounds				
GCS 5	4.10	0.905	0.786	40.3sec
GCS 20	4.08	0.907	0.790	234.4sec
Different λ				
GNG 200	8.11	0.802	0.675	12.3sec
GCS 200	3.72	0.906	0.787	35.3sec
GNG 400	8.13	0.873	0.705	24.8sec
GCS 400	3.59	0.913	0.789	75.8sec
GNG 2000	7.84	0.944	0.750	113.4sec
GCS 2000	3.45	0.924	0.790	415.2sec
Different a_{max}				
GNG 44	7.97	0.797	0.685	6.5sec
GNG 176	12.38	0.318	0.459	6.4sec

In our test we investigate for GCS the neighbor smoothing process and different values for λ. For GNG we will use different values for λ and different maximum lifespans for the edges a_{max} (see table 1).

5 CONCLUSIONS

Both algorithms have great potential in geometric approximation, because of their resistance to noise and none uniform sample densities and because of their independence of sample set sizes. For structures of mixed or unknown dimension GNG is capable of an approximation, even so we could not think of a practical use case. Apart from that the algorithm needs in-

herently more points to reconstructed an approximation of the same resolution in comparison to the GCS approach. Thru low age_{max} values triangles quality can be gained in exchange for more holes in the surface. Thru λ time can be exchanged for an overall better approximation. We found the best trade of at $age_{max} = 88$ and $\lambda = 200$. GNG needs a cleaning phase, that can only be done in a post-process. However we think that an GNG approach based on radial base functions might has the potential to overcame some of these disadvantages.

The GCS approach is less time efficient, which is mostly due to its more sophisticated graph. The smoothing operation, is also time consuming, but has very positive effects on the valance, triangle quality and also the distance to the points. Note that this operation cannot be used for GNG, since the operation uses triangle normals.

In our analysis of these approaches we came to the conclusion that the progressively evolving surface in the GCS approach and the ability to augment the process with enhancing operations that smoothly integrate in the process due to the more sophisticated base graph makes the process is in context of geometric approximation overall to the superior technique.

REFERENCES

Annuth, H. and Bohn, C.-A. (2010a). Surface reconstruction with smart growing cells. *Studies in Computational Intelligence*, 321:47–66.

Annuth, H. and Bohn, C. A. (2010b). Tumble tree: reducing complexity of the growing cells approach. In *Proceedings of the 20th international conference on Artificial neural networks: Part III*, ICANN'10, pages 228–236, Berlin, Heidelberg. Springer-Verlag.

Annuth, H. and Bohn, C.-A. (2012). Smart growing cells: Supervising unsupervised learning. In Madani, K., Dourado Correia, A., Rosa, A., and Filipe, J., editors, *Computational Intelligence*, volume 399 of *Studies in Computational Intelligence*, pages 405–420. Springer Berlin / Heidelberg. 10.1007/978-3-642-27534-0_27.

Barhak, J. and Fischer, A. (2002). Adaptive reconstruction of freeform objects with 3d som neural network grids. *Computers & Graphics*, 26(5):745–751.

do Rego, R. L. M., Arajo, A. F. R., and de Lima Neto, F. B. (2007). Growing self-organizing maps for surface reconstruction from unstructured point clouds. In *IJCNN*, pages 1900–1905. IEEE.

Do Rêgo, R. L. M. E., Araújo, A. F. R., and De Lima Neto, F. B. (2010). Growing self-reconstruction maps. *Trans. Neur. Netw.*, 21(2):211–223.

Fritzke, B. (1993). Growing cell structures - a self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7:1441–1460.

Fritzke, B. (1995). A growing neural gas network learns topologies. In Tesauro, G., Touretzky, D. S., and Leen, T. K., editors, *Advances in Neural Information Processing Systems 7*, pages 625–632. MIT Press, Cambridge MA.

Fritzke, B. (1996). Automatic construction of radial basis function networks with the growing neural gas model and its relevance for fuzzy logic. In *Proceedings of the 1996 ACM symposium on Applied Computing*, SAC '96, pages 624–627, New York, NY, USA. ACM.

Hoffmann, M. and Vrady, L. (1998). Free-form surfaces for scattered data by neural networks. *Journal for Geometry and Graphics*, 2:1–6.

Holdstein, Y. and Fischer, A. (2008). Three-dimensional surface reconstruction using meshing growing neural gas (mgng). *Vis. Comput.*, 24:295–302.

Ivrissimtzis, I., Jeong, W.-K., and Seidel, H.-P. (2003a). Neural meshes: Statistical learning methods in surface reconstruction. Technical Report MPI-I-2003-4-007, Max-Planck-Institut f"ur Informatik, Saarbrücken.

Ivrissimtzis, I. P., Jeong, W.-K., and Seidel, H.-P. (2003b). Using growing cell structures for surface reconstruction. In *SMI '03: Proceedings of the Shape Modeling International 2003*, page 78, Washington, DC, USA. IEEE Computer Society.

Jeong, W.-K., Ivrissimtzis, I., and Seidel, H.-P. (2003). Neural meshes: Statistical learning based on normals. *Computer Graphics and Applications, Pacific Conference on*, 0:404.

Kohonen, T. (1982). Self-Organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics*, 43:59–69.

MacQueen, J. B. (1967). Some methods for classification and analysis of multivariate observations. pages 281 – 297.

Martinetz, T. and Schulten, K. (1991). A "Neural-Gas" Network Learns Topologies. *Artificial Neural Networks*, I:397–402.

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

Melato, M., Hammer, B., and Hormann, K. (2008). Neural gas for surface reconstruction. *IfI Technical Report Series*.

Taubin, G. (1995). A signal processing approach to fair surface design. In *SIGGRAPH '95: Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, pages 351–358, New York, NY, USA. ACM.

Vrady, L., Hoffmann, M., and Kovcs, E. (1999). Improved free-form modelling of scattered data by dynamic neural networks. *Journal for Geometry and Graphics*, 3:177–183.

Yu, Y. (1999). Surface reconstruction from unorganized points using self-organizing neural networks. In *In IEEE Visualization 99, Conference Proceedings*, pages 61–64.