

# Smart Growing Cells: Supervising Unsupervised Learning

Hendrik Annuth and Christian-A. Bohn

Wedel University of Applied Sciences  
{annuth,bohn}@fh-wedel.de  
<http://cg.fh-wedel.de>

**Abstract.** In many cases it is reasonable to augment general unsupervised learning by additional algorithmic structures. Kohonens self-organizing map is a typical example for such kinds of approaches. Here a 2D mesh is superimposed on pure unsupervised learning to extract topological relationships from the training data. In this work, we propose generalizing the idea of application-focused modification of ideal, unsupervised learning by the development of the *smart growing cells* (SGC) based on Fritzke’s *growing cells structures* (GCS). We substantiate this idea by presenting an algorithm which solves the well-known problem of surface reconstruction based on 3D point clouds and which outperforms the most classical approaches concerning quality and robustness.

**Keywords:** Neural networks, unsupervised learning, self-organization, growing cell structures, surface reconstruction.

## 1 Introduction

The idea of developing the smart growing cells approach is driven by the need of an algorithm for robust surface reconstruction from 3D point sample clouds.

The demand for efficient high quality reconstruction algorithms has grown significantly in the last decade, since the usage of 3D point scans has widely been spread into new application areas. These include geometric modeling to supplement interactive creation of virtual scenes, registering landscapes for navigation devices, tracking of persons or objects in virtual reality applications, medicine, or reverse engineering.

3D points, retrieved by laser scanners or stereo cameras, introduce two vital questions. First, how can one recognize a topology of the originating 2D surfaces just from independent 3D sample points and without any other information from the sampled objects? Second, for further processing, how is it possible to project this topological information on a data structure like a triangle mesh — meeting given constraints concerning mesh quality and size?

Although this issue has intensely been tackled since the early eighties [1] a general concept that addresses all the problems of surface reconstruction has not been determined up to now. Noise contained in the sample data, anisotropic

point densities, holes and discontinuities like edges, and finally, handling vast amounts of sampling data are still a big challenge.

**Previous work.** The issue of surface reconstruction is a major field in computer graphics. There are numerous approaches with different algorithmic concepts. In [6] and [12] an implicit surface is created from point clouds which then is triangulated by the marching cubes approach. [2] and [14] reduce a delaunay tetrahedralization of a point cloud until the model is carved out. Approaches like [16] or [7] utilize techniques based on the Bayes’ theorem.

In the area of artificial neural networks a famous work is [13]. They propose the *Self Organizing Map* (SOM) which iteratively adapts its internal structure — a 2D mesh — to the distribution of a set of samples and enables clustering or dimensionality reduction of the sample data. While a SOM has a fixed topology, the growing cells structures concept [3, 4] allows the network for dynamically fitting its size to the sample data complexity. SOM and GCS are suitable for processing and representing vector data like point samples on surfaces. [5] uses a SOM and [18] and [20] a GCS for the purpose of surface reconstruction. Further improvements are made by [8] where constant Laplacian smoothing [17] of surfaces is introduced, and in [9] the curvature described by the input sample distribution is taken to control mesh density. In [11] the GCS reconstruction process is further enhanced in order to account for more complex topologies. [10] use several meshes of the same model for a mesh optimization process, and [19] present a concept for combining common deterministic approaches and the advantages of the GCS approach.

In the following, we outline the basis of our approach — the growing cells structures — and then derive our idea of the smart growing cells, which matches the specific requirements of reconstruction. Afterwards, an analysis is compiled discussing mesh quality and performance of our approach, and finally, we close with a summary and a list of future options of this work.

## 2 Reconstruction with Smart Growing Cells

Classical growing cells approaches for reconstruction tasks are based on using the internal structure of the network as a triangulation of the object described by a set of surface sample points. A 2D GCS with 3D *cells* is trained by 3D sample points. Finally, the cells lie on the object surface which the 3D points represent. The network structure — a set of 2D simplices (triangles) — is directly taken as triangulation of the underlying 3D object. The reason for using a GCS for reconstruction are its obvious advantages compared to deterministic approaches.

- They can robustly handle arbitrary sample set sizes and distributions which is important in case of billions of unstructured points.
- They are capable of reducing noise and ply discontinuities in the input data.
- They are capable of adaption — it is not required to regard all points of the sample set on the whole. Incrementally retrieved or stored samples can be used for retrain without starting the triangulation process from scratch.



Place  $k$  reference vectors  $\mathbf{c}_i \in \mathbb{R}^n$ ,  $i \in \{0..k-1\}$  randomly in input space.

**repeat**

Chose sample  $\mathbf{s}_j \in \mathbb{R}^n$  randomly from the input set.

Find reference vector  $\mathbf{c}_b$  closest to  $\mathbf{s}_j$  (“*best matching*” or “*winning unit*”).

Move  $\mathbf{c}_b$  into the direction of  $\mathbf{s}_j$  according to a certain strength  $\epsilon_{bm}$ , like  $\mathbf{c}_b^{\text{new}} = \mathbf{c}_b^{\text{old}}(1 - \epsilon_{bm}) + \mathbf{s}_j\epsilon_{bm}$ .

Decrease  $\epsilon_{bm}$ .

**until**  $\epsilon_{bm} \leq$  certain threshold  $\epsilon_0$ .

**Fig. 1.** The general unsupervised learning rule.

- They guarantee to theoretically find the best solution possible. Thus, approximation accuracy and mesh quality are automatically maximized.

Nevertheless, these advantages partly clash with the application of reconstruction. On the one hand, discontinuities are often desired (for example, in case of edges or very small structures on object surfaces). On the other hand, smoothing often destroys important aspects of the model under consideration (for example, if holes are patched, if separate parts of the underlying objects melt into one object, or if the object has a very complex, detailed structure). In such cases, GCS tend to generalize which mostly lets vanish visually important features which the human is sensitized to.

The presented smart growing cells approach accounts for these application-focused issues and emphasizes that modification of the general learning task in the classical GCS is suitable for many novel application fields.

## 2.1 Unsupervised Learning and Growing Cells Structures

*General unsupervised learning* is very similar to *k-means clustering* [15] which is capable of placing  $k$   $n$ -dimensional reference vectors in a set of  $n$ -dimensional input samples such that they may be regarded as means of those samples which lie in the  $n$ -dimensional Voronoi volume of the reference vectors. Unsupervised learning is based on iteratively adapting reference vectors by comparing them to the  $n$ -dimensional input samples set, described with the algorithm in Fig. 1. Surface reconstruction with pure unsupervised learning would place a set of reference vectors on the object but does not determine information about the underlying surface topology, which leads to the Kohonen Self Organizing Map.

*The Kohonen self organizing map* is based on reference vectors which now are connected through a regular 2D mesh. The general unsupervised learning rule is extended to account for the direct neighborhood of a best matching unit with the loop from Fig. 2.

Insertion into the general unsupervised learning algorithm (after moving of  $\mathbf{c}_b$ ) in Fig. 1 leads to the phenomenon that the reference vertices now are moved by

```

for all  $\mathbf{c}_{nb} \in \text{neighborhood of } \mathbf{c}_b$  do
    Move  $\mathbf{c}_{nb}$  in the direction of  $\mathbf{s}_j$  according to a certain strength  $\epsilon_{nb}$ , like
     $\mathbf{c}_{nb}^{\text{new}} = \mathbf{c}_{nb}^{\text{old}}(1 - \epsilon_{nb}) + \mathbf{s}_j\epsilon_{nb}$ .
    Decrease  $\epsilon_{nb}$ .
end for

```

**Fig. 2.** Accounting for the cell topology by introducing the neighborhood of a winning unit in the general training from Fig. 1.

accounting for the regular 2D mesh topology of the SOM. Training a plane-like sample set leads to an adaption of the SOM grid to this implicit plane — the sample topology is recognized and finally represented by the SOM mesh.

Nevertheless, the mesh size of a SOM is fixed and cannot adjust to the sample structure complexity. The growing cells structures overcome this drawback.

*The Growing cells structures* — to a certain degree — may be seen as SOM which additionally are capable of growing and shrinking according to the problem under consideration which is defined by the sample distribution. This mechanism is based on a so called *resource term* contained in every reference vector and which — in the original approach — is a simple counter. It counts how often a certain reference vector has been detected being a best matching unit. A big counter value signalizes the requirement for insertion of new reference vectors.

With a GCS one could train a sample set lying on a certain object surface and the network structure would fit the object surface at a certain approximation error. The problem is that in reconstruction tasks sample distributions are often not uniform. The represented surfaces usually contain discontinuities like sharp edges and holes, and the objects to be reconstructed are not that simple like a plane or a tetrahedron. The latter usually are chosen as initial networks and can hardly adapt to complex topologies, since only objects which are homeomorphic the start object can be represented satisfactorily.

Thus, general unsupervised learning must evolve to a kind of constrained unsupervised learning which detects and adapts to certain structures which the sample set implicitly contains.

## 2.2 Smart Growing Cells

Let’s have a “biological view” on a network of neural cells. Here, growing cells would expose a typical unicellular organism — all cells are identical, they possess the same abilities.

Now, smart growing cells break this limitation, like it happens in “real world”. During training they change their capabilities according to the tasks they will have to fulfill and which is implicitly determined by the training input — in case of the SGC, the underlying sample distribution. This changes the abilities of the general unsupervised neural network significantly. In contrast to common

```

repeat
  for  $j = 1$  to  $k_{del}$  do
    for  $i = 1$  to  $k_{ins}$  do
      Select sample  $s$  from point cloud randomly, find closest neural vertex and
      move it together with neighbor vertices towards  $s$ .

      Increase signal counter at  $s$  (the resource term mentioned above) and
      decrease the signal counters of all other vertices.

    end for
    Find best performing neural vertex (with highest signal counter value) and
    add new vertex at this position (see Fig. 4).

  end for
  Find worst performing neural vertices, delete them and collapse regarding
  edges (see Fig. 4).
until certain limit like approximation error, or number of vertices is reached.

```

**Fig. 3.** Classical growing cells structures algorithm.

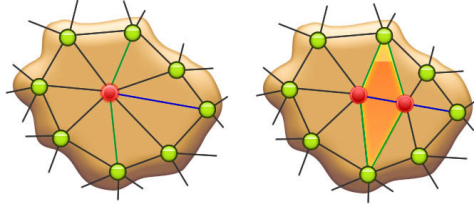
GCS networks where the cell behavior is limited to rules that concern Euclidian distance only, the behavior of an SGC network can precisely be modelled but without breaking the favorable, iterative, unsupervised characteristics of the GCS training rules.

The SGC basic structure is identical to general GCS, i.e., there are  $n$ -dimensional cells which — from now on — are termed *neural vertices* connected by *links* through an  $m$ -dimensional topology. Let  $n = 3$  since neural vertices are directly taken as vertices of the triangulation mesh and  $m = 2$  since we aim at 2D surfaces to be reconstructed.

The main training loop is outlined in Fig. 3. Here  $k_{del}$  and  $k_{ins}$  are simple counter parameters defined below (see section 2.3). Movements of vertices and their neighbors slightly differ from the classical SOM. Again, there are two parameters for the learning rates,  $\epsilon_{bm}$  for the winner and  $\epsilon_{nb}$  for its neighbors, but these are not decreased during learning since vertex connections automatically become smaller together with the learning rates. For drawing the neighboring vertices, a smoothing process like described in [8] and [17] is applied, which replaces the classical movement, and which makes the adaption of the topology more robust.

As initial network, usually a tetrahedron or a plane with random vertices is suitable. Two operations enable the network to grow and shrink — “vertex split” and “edge collapse”.

*The vertex split* operation adds three edges, two faces, and a new neural vertex. The longest edge at the neural vertex with the highest resource term is split and a new vertex is added in the middle. The signal counter value is equally spread between the two vertices (see Fig. 4).



**Fig. 4.** Neural vertex split operation (read from left to right) to increase mesh granularity locally, and edge collapse (read from right to left) to shrink mesh locally.

*Edge collapse* removes all neural vertices with resource terms below a certain threshold  $r_{min}$  together with three edges and two connected faces (see Fig. 4). The determination of the edge to be removed is driven by connectivity irregularities as proposed in [8].

It follows the adaption of the cell behavior driven by the application needs of surface reconstruction. It leads to our proposal of lifting cell capabilities above that of general unsupervised learning described in the following paragraphs.

**A) Cell Weeding.** Deleting neural vertices which are not part of a sound underlying mesh structure is the most important new training rule of the SGC approach. It is essential for giving the network the chance of adapting to any topology despite of its initial topology (overcoming the homomorphic restriction). Before the edge collapse operation is applied at a vertex, it will be tested if the vertex is contained in a *degenerated mesh region* (definition follows below). If so, an *aggressive cut out* of the vertex and its surrounding vertices is started.

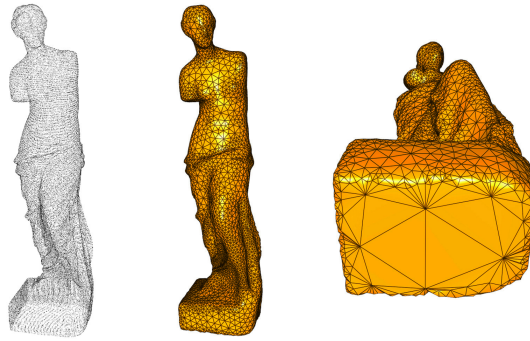
It has been shown that degeneration of a part of a mesh serves as perfect indicator for a mesh topology which does not fit the underlying sample structure correctly. For example, consider a region where sample densities equal zero. Although vertices are not directly drawn into it by training adjustment, their neighbors may be moved there through their mesh connections. Due to their resource terms, these vertices will be deleted by edge collapse operations, but their links remain and mistakenly represent the existence of some topology. In this case, the structure of the links is degenerated, i.e., it usually shows a surpassing number of edges with *acute-angled*<sup>1</sup> vertices (see Fig. 5).

The reason for terming this deletion "aggressive" are the triggering properties which are quite easy to match — suspicious neural vertices will be cut out early.

A *Criterion for degenerated mesh regions* is already proposed in [11] where a large area of a triangle is taken as sign for a degenerated mesh structure. But it has been shown that this criterion warns very late. Also, anisotropic sample densities are mistakenly interpreted as degenerated mesh regions. Our proposal is a combination of *vertex valence*<sup>2</sup>, triangle quality, and quality of neighboring

<sup>1</sup> A triangle is termed acute-angled if the ratio of its area and the area which is spanned by a second equilateral triangle built from the longest edge of the first lies below a certain threshold  $\epsilon_{acute}$ .

<sup>2</sup> Vertex valence is the number of connected vertices.



**Fig. 5.** Statue’s bottom is not represented by samples. On the right, the acute-angled triangles expose a degenerated mesh region.

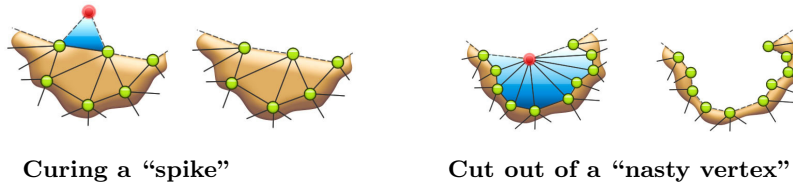
vertices. If all of the following conditions hold, deleting of the mesh structure at that vertex is triggered.

1. Vertex valence rises above a certain threshold  $n_{degvalence}$ .
2. Vertex is connected to at least  $n_{degacute}$  acute-angled triangles.
3. Vertex has more than  $n_{degnb}$  neighbors for which conditions (1) or (2) hold.

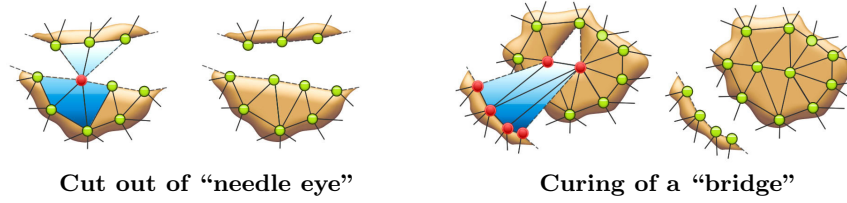
The latter condition says that deletion is only started if at least one or two neighbors have the same inconsistencies in their local mesh structure. This is reasonable since single degenerated vertices do not necessarily expose a problem but may arise by accident.

*Curing boundaries after weeding* is needed, since, after an aggressive extinction of a neural vertex and its surrounding faces has happened, usually a boundary will be left which may consist of unfavorable mesh structure elements. Curing finds these structures along the boundary and patches them discriminating between four cases, described in the following.

*i) The Spike.* A boundary vertex with a valence of 2 (see left part of Fig. 6) is termed a spike. This type of vertex is very unlikely to support a correct reconstruction process since it will be adjusted to an acute-angled triangle after few iteration steps. A spike must be deleted completely.



**Fig. 6.** Two types of unwanted vertices and their extinction.



**Fig. 7.** Two types of unwanted connections between separate topologies.

ii) *The Nasty Vertex.* A nasty vertex is a neural vertex with at least  $n_{nastyacute}$  acute-angled triangles and/or triangles with a valence greater than  $n_{nastyval}$  (see Fig. 6). It is suspected to be part of a degenerated mesh region and is deleted.

iii) *The Needle Eye.* A needle eye is a neural vertex that is connected to at least two boundaries (see Fig. 7, on the left). At these locations the mesh does not have a valid mesh structure. To delete a needle eye, all groups of connected faces are determined — the group with the most faces survives, all others are deleted.

iv) *The Bridge.* A bridge is very likely to be part of a degenerated mesh region. A mesh with a hole consisting of three vertices would soon be closed by a coalescing process (see section 2.2). This is not allowed if exactly one of the edges of this hole would additionally be connected to a face (which we term a “bridge”, see Fig. 7) since an invalid edge with three faces would arise. The entire bridge structure is deleted and the hole will be closed by generating a new face.

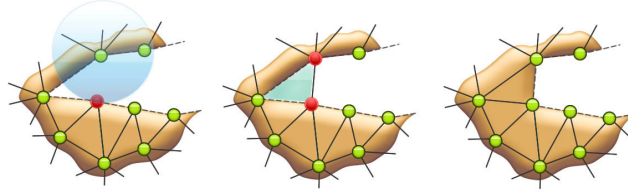
**B) Coalescing Cells.** Like a mesh can be split through deletion of vertices, it must also be possible to merge two mesh boundaries during training. For that, a coalescing test is accomplished each time a vertex at a mesh boundary is moved.

*The coalescing test* determines if two boundaries are likely to be connected to one coherent area. For that, a sphere is created with the following parameters. Given the neighboring boundary vertices  $\mathbf{v}_1$  and  $\mathbf{v}_2$  of  $\mathbf{c}_b$ , then let  $\mathbf{c} = \mathbf{1}/2(\mathbf{v}_1 + \mathbf{v}_2)$ . A *boundary normal*  $\mathbf{n}_c$  is calculated as the average of all vectors originating at  $\mathbf{c}$  and ending at neighbors of  $\mathbf{c}_b$ , where  $\mathbf{v}_1$  and  $\mathbf{v}_2$  are not taken into account. The boundary normal can be seen as a direction pointing to the opposite side of the boundary. We define a sphere with the center at  $\mathbf{c} + \mathbf{n}_c r$  with radius  $r$  as the average length of the edges at  $\mathbf{c}_b$ .

The coalescing condition at two boundaries hold, i.e., merging of the boundaries containing  $\mathbf{c}_b$  and  $\mathbf{q}$  on the opposite side happens, if

- $\mathbf{q}$  is contained in the defined sphere, and
- scalar product of the boundary normals at  $\mathbf{c}_b$  and  $\mathbf{q}$  is negative.

*The Coalescing process* is required since after detecting the neural vertex  $\mathbf{q}$  to be connected with  $\mathbf{c}_b$ , the according faces must be created starting with one edge from  $\mathbf{c}_b$  to  $\mathbf{q}$ . There are two cases which have to be considered.



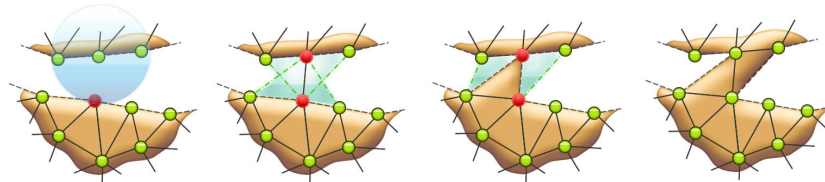
**Fig. 8.** Coalescing process at a mesh corner. On the left, the search process of a coalescing candidate. In the middle, one edge is created, on the right, the only face capable of being added is the corner face.

*i) Corner.* A corner of the same boundary arises when  $\mathbf{c}_b$  and  $\mathbf{q}$  have one neighboring vertex in common (see Fig. 8). A triangle of the three participating vertices is created.

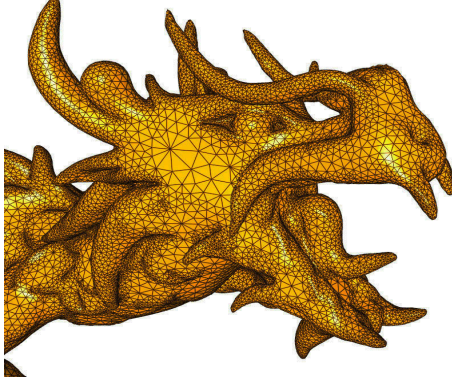
*ii) Long side.* Here, two boundaries appear to be separated. After determining the new edge, there are four possibilities for insertion of a new face containing the edge (see second picture in Fig. 9). The triangle with edge lengths which vary fewest is taken in our approach (see third picture in Fig. 9) since it is the triangle with the best features concerning triangle quality. Finally, to avoid a needle eye, a further triangle must be added — again, the face with the greatest edge similarity is taken (see fourth picture in Fig. 9).

**C) Roughness Adaption.** Up to now, the SGC are able to approximate an arbitrary sample set by a 2D mesh. What remains is an efficient local adaption of the mesh density in a way that areas with a strong curvature are modeled by a finer mesh resolution (see Fig. 10). This also relieves the influence of the sample density on the mesh granularity making the SGC less vulnerable to sampling artefacts like holes or regions which are not sampled with a uniform distribution.

Each time a vertex is adapted by a new sample the estimated normal  $\mathbf{n}_k$  at a neural vertex  $\mathbf{v}_k$  is calculated by the average of the normals at the surrounding



**Fig. 9.** Coalescing of two separate boundaries. In the second picture, the edge is determined, in the third, the triangle with smallest variance of edge lengths is added, in the fourth, another triangle must be added to avoid a needle eye.



**Fig. 10.** Roughness adaption correlates surface curvature with mesh density, details of the model are accentuated.

faces. The curvature  $c_k \in \mathbb{R}$  at a vertex is determined by

$$c_k = 1 - \frac{1}{|\mathcal{N}_k|} \sum_{\forall \mathbf{n} \in \mathcal{N}_k} \mathbf{n}_k \cdot \mathbf{n} \quad (1)$$

with the set  $\mathcal{N}_k$  containing the normals of the neighboring neural vertices of  $\mathbf{v}_k$ . Each time a neural vertex is selected as winner, its curvature value is calculated and a global curvature value  $\bar{c}$  is adjusted. Finally, the curvature dependent resource term  $r_k$  at  $\mathbf{v}_k$  is adapted through  $r_k^{new} = r_k^{old} + \Delta r_k$ , and

$$\Delta r_k = \begin{cases} 1, & \text{if } (c_k < \bar{c} + \sigma_{r_k}) \\ [c_k / (\bar{c} + \sigma_{r_k})] (1 - r_{min}) + r_{min} & \text{else,} \end{cases} \quad (2)$$

with the deviation  $\sigma_{r_k}$  of the resource term  $r_k$ , and a constant resource  $r_{min}$  that guarantees that the mesh does not completely vanish at plane regions with a very small curvature.

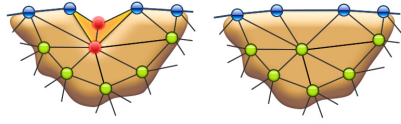
**D) Discontinuity Cells.** A sampled model that exposes discontinuities like edges is difficult to be approximated by the neural network mesh. Discontinuities are smoothed out since the network tries to create a surface over them. This might be acceptable in many application areas since the approximation error is fairly small, but this effect is unfavorable in computer graphics since it is clearly visible. And even worse: edges are quite common in real world scenarios.

Therefore, we propose discontinuity neural vertices which, first, are only capable of moving in the direction of an object edge to represent them more properly, and second, the smoothing process is not applied to them.

Recognizing those vertices is accomplished as follows. The curvature values of those neighbors which have a distance of two connections from the vertex (the “second ring” of neighbors) are determined. Then the average  $\delta_{ring}$  of the squared differences of consecutive curvature values on the ring is calculated.

If a curvature value clearly deviates from the average curvature value, it is assumed being a discontinuity vertex if the average of the neighbors’ (second





**Fig. 11.** A dent (left picture) on a sharp edge is solved (right picture) by an edge swap operation. Finally, connections of discontinuity vertices model object edges

ring) curvature gradient differs to a certain amount. Thus, a vertex  $\mathbf{v}_k$  is defined a discontinuity vertex if

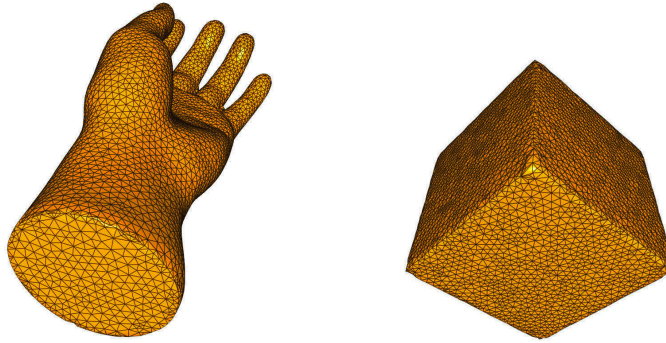
$$(c_k > 2\sigma_{c_k}) \wedge (\forall c \in \mathcal{C}_k : \delta_{ring} > 4\sigma_{c_k}^2) \quad (3)$$

with  $\mathcal{C}_k$  the set of curvature values of the second ring of neighbors.

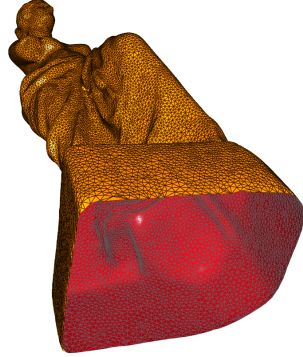
For approximating the edge normal the average of the normals of two of the neighboring vertices of  $\mathbf{v}_k$  are taken, either those with the highest curvature value, or those which are already marked as discontinuity vertex. Finally, the normal is mirrored if the edge angle lies above  $180^\circ$  indicated by the average of the surrounding vertex normals; in the first case it points in the direction of  $\mathbf{v}_k$ .

*An Edge swap operation* is applied if two connected discontinuity vertices grow into an edge, they nicely represent this edge by a triangle edge. But if the line is interrupted by a non-discontinuity vertex, a dent arises since this vertex is not placed on the edge. In this case an edge swap process is proposed which minimizes this effect.

Each time a discontinuity vertex is moved towards a sample, the need for an edge swap operation will be determined by collecting the three consecutive faces with the most differing face normals. In case of a dent, the face in the middle is assumed to be the one which is misplaced and an edge swap operation is applied (see Fig. 11). Then, if the difference of the normals is now lower than before, edge swap is accepted, if not, the former structure is held.



**Fig. 12.** Discontinuity vertices focus on edges. Edge swap operations let mesh edges map to object edges.



**Fig. 13.** Mesh boundary due to the missing bottom of the statue is represented exactly by boundary cells.

Edge swap results in models where finally edges are represented by mesh boundaries (see Fig. 12).

**E) Boundary Cells** Similar to discontinuity vertices which are capable of moving to object edges, boundary vertices are able to move to the outer border of a surface (see Fig. 13). They are recognized by being part of a triangle edge which is connected to one face only.

Then, these vertices are moved only into the direction of the boundary normal like described in section 2.2 for avoiding vertices just lying in the average of the surrounding samples but directly match the surface boundaries at their locations.

### 2.3 Results

For the full algorithm of this approach see the pseudocode in Fig. 14. To keep it comprehensive, the outermost loop of the algorithm is neglected, and vertex split and edge collapse operations are triggered by counters.

Parameters which have been proven to be reliable for almost all sample sets we took for reconstruction are  $\epsilon_{bm} = 0.1$ ,  $\epsilon_{nb} = 0.08$ ,  $r_{min} = 0.3$ ,  $\epsilon_{acute} = 0.5$ ,  $n_{degacute} = 4$ ,  $k_{ins} = 100$ ,  $k_{del} = 5$ ,  $n_{degnb} = 1$ ,  $n_{nastyacute} = 4$ ,  $n_{nastyval} = 3$ .

The following results have been produced on a *Dell*<sup>®</sup>Precision M6400 Notebook with *Intel*<sup>®</sup>Core 2 Extreme Quad Core QX9300 (2.53GHz, 1066MHz, 12MB) processor with 8GB 1066 MHz DDR3 Dual Channel RAM. The algorithm is not parallelized.

Visual results are exposed in Fig. 16. All pictures are drawn from an SGC mesh. Most models stem from the *Stanford 3D Scanning Repository*. Besides visual results, reconstruction with SGC comes up with impressive numbers compared to classical approaches, which are listed in the table in Fig. 15. It can be seen that mesh quality, i.e. the percentage of perfect triangles in the mesh lies at 96% at average. This is an outstanding but expected result, when using an approach from the field of unsupervised learning, since this guarantees an ideal representation of the underlying training sample distribution.

```

Adjust samples regarding roughness.
Calculate average curvature and deviations.
Recognize and sign discontinuity and curvature cells.
for all Boundary cells do
  if  $\exists$  coalescing candidate then
    Melt boundary.
    for all Weeding candidates do
      Weeding process.
    end for
  end if
end for
if Edge collapse operation triggered then
  Collapse edge.
  for all Weeding candidates do
    Weeding process
  end for
end if
if Vertex split operation triggered then
  Split vertex.
end if

```

**Fig. 14.** Outline of the complete SGC algorithm.









Further, the distance (RMS/object size) between samples and mesh surface is negligible low — far below 1% of the object size at average. This is even more pleasant, since usually the problem at edges generate big error terms. Also the computing times needed are very short, just few minutes in common cases.

All those measurements are far better than those from classical approaches, in so far as these are revealed in the specific papers. Our algorithm works very robustly. There are nearly no outliers visible in the mesh.

### 3 Conclusions

We presented smart growing cells as an expansion of general unsupervised learning. The novel core skill of a smart growing cell is the added intelligence — cells may not only adapt to the sample distribution but can also use application-focused aspects of the data they operate on. This extension is consistently injected into the standard unsupervised learning rule avoiding the extinction of its beneficial properties.

As a proof of concept a surface reconstruction algorithm was presented that overcomes the well-known problems of approaches that use the standard growing cells structures concept. Features like creases and corners are preserved, the triangle density relates to the curvature and arbitrary topologies are reconstructed. Our approach even outperforms classical surface reconstruction approaches. For

								
Samples	36K	438K	544K	14,028K	5,000K	511K	38K	346K
Vertices	30K	100K	260K	320K	500K	10K	5K	346K
Time [m:s]	0:39	2:47	9:15	12:17	21:5	0:11	0:6	0:6
Quality	95.6	95.5	93.1	98.5	95.9	99.8	99	98.3
RMS/Size	4.7e-5	3.3e-5	1.7e-5	1.3e-5	2.7e-5	6.6e-5	15e-5	0.7e-5

**Fig. 15.** Results with sample sets from the *Stanford 3D Scanning Repository*. “Quality” means percentage of triangles which hold the Delaunay criterion. RMS/Size is the root of the squared distances between original point samples and the triangle mesh, divided by the diameter of the sample set.

evaluation purposes, several sample sets were used which provide different reconstruction challenges (see table in Fig. 15). The average deviation of the mesh from the sample points is  $2 \cdot 10^{-3} \%$  compared to the diameter of the object under consideration, and about 96% of the triangles fulfil the Delaunay criterion for triangle quality.

A further important feature of SGC is their robustness. The network is able to handle arbitrary topologies and billions of samples easily. It recognizes and solves discontinuities in the sample data and it is capable of adapting to varying sample distributions without the need for training from the scratch.

The network is able to match arbitrary surface structures like single objects, landscapes, or even separate objects with very complex topologies. The final triangulation is directly taken from the network structure. No additional triangulation or cleaning processes are required.

**Future work.** We propose three directions for ongoing work on SGC. First, currently, the additional intelligence of a cell is purely defined on heuristics. A great improvement would be to use a separate neural network which detects and realizes cell behavior automatically. Second, in the area of surface reconstruction smart growing cells produced great results which is a proof of concept. To establish the flexibility of smart growing cells new applications cases will be realized. Third, the movement of a cell is independent from other cells. This offers great opportunities concerning the parallelisation of the presented approach. With a decent speed up real time application should be in reach.

## References

1. Jean-Daniel Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Trans. Graph.*, 3(4):266–286, 1984.
2. Herbert Edelsbrunner and Ernst P. Mcke. Three-dimensional alpha shapes, 1994.
3. Bernd Fritzke. Growing cell structures - a self-organizing network for unsupervised and supervised learning. *Neural Networks*, 7:1441–1460, 1993.

4. Bernd Fritzke. A growing neural gas network learns topologies. In G. Tesauro, D. S. Touretzky, and T. K. Leen, editors, *Advances in Neural Information Processing Systems 7*, pages 625–632. MIT Press, Cambridge MA, 1995.
5. Mikls Hoffmann and Lajos Vrády. Free-form surfaces for scattered data by neural networks. *Journal for Geometry and Graphics*, 2:1–6, 1998.
6. Hugues Hoppe, Tony DeRose, Tom Duchamp, John Alan McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In James J. Thomas, editor, *SIGGRAPH*, pages 71–78. ACM, 1992.
7. Qi-Xing Huang, Bart Adams, and Michael Wand. Bayesian surface reconstruction via iterative scan alignment to an optimized prototype. In *SGP '07: Proceedings of the fifth Eurographics symposium on Geometry processing*, pages 213–223, Aire-la-Ville, Switzerland, Switzerland, 2007. Eurographics Association.
8. I. P. Ivriissimtzis, W-K. Jeong, and H-P. Seidel. Using growing cell structures for surface reconstruction. In *SMI '03: Proceedings of the Shape Modeling International 2003*, page 78, Washington, DC, USA, 2003. IEEE Computer Society.
9. Ioannis Ivriissimtzis, Won-Ki Jeong, and Hans-Peter Seidel. Neural meshes: Statistical learning methods in surface reconstruction. Technical Report MPI-I-2003-4-007, Max-Planck-Institut für Informatik, Saarbrücken, April 2003.
10. Ioannis Ivriissimtzis, Yunjin Lee, Seungyong Lee, Won-Ki Jeong, and Hans-Peter Seidel. Neural mesh ensembles. In *3DPVT '04: Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium*, pages 308–315, Washington, DC, USA, 2004. IEEE Computer Society.
11. I.P. Ivriissimtzis, W.-K. Jeong, S. Lee, Y. Lee, and H.-P. Seidel. Neural meshes: surface reconstruction with a learning algorithm. Research Report MPI-I-2004-4-005, Max-Planck-Institut für Informatik, Stuhlsatzenhausweg 85, 66123 Saarbrücken, Germany, October 2004.
12. Michael Kazhdan, Matthew Bolitho, and Hugues Hoppe. Poisson surface reconstruction. In *Proceedings of the fourth Eurographics symposium on Geometry processing*, SGP '06, pages 61–70, Aire-la-Ville, Switzerland, Switzerland, 2006. Eurographics Association.
13. Teuvo Kohonen. Self-Organized Formation of Topologically Correct Feature Maps. *Biological Cybernetics*, 43:59–69, 1982.
14. Ravikrishna Kolluri, Jonathan Richard Shewchuk, and James F. O'Brien. Spectral surface reconstruction from noisy point clouds. In *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 11–21, New York, NY, USA, 2004. ACM.
15. J. B. MacQueen. Some methods for classification and analysis of multivariate observations. pages 281 – 297, 1967.
16. Geir Storvik. Bayesian surface reconstruction from noisy images. In *Interface 96*, 1996.
17. Gabriel Taubin. A signal processing approach to fair surface design. In *SIGGRAPH*, pages 351–358, 1995.
18. Lajos Vrády, Mikls Hoffmann, and Emd Kovcs. Improved free-form modelling of scattered data by dynamic neural networks. *Journal for Geometry and Graphics*, 3:177–183, 1999.
19. Mincheol Yoon, Yunjin Lee, Seungyong Lee, Ioannis Ivriissimtzis, and Hans-Peter Seidel. Surface and normal ensembles for surface reconstruction. *Comput. Aided Des.*, 39(5):408–420, 2007.
20. Yizhou Yu. Surface reconstruction from unorganized points using self-organizing neural networks. In *IEEE Visualization 99, Conference Proceedings*, pages 61–64, 1999.



**Fig. 16.** Upper lines: mesh training stages with number of vertices, lower lines, assorted pictures of reconstructed models.