Surface Reconstruction with Smart Growing Cells

Hendrik Annuth and Christian-A. Bohn

Abstract. We propose *Growing Cells Meshing* (GCM) — a reconstruction algorithm which creates triangle meshes from clouds of arbitrary point samples registered on object surfaces. GCM is different to classical approaches in the way that it uses an artificial neural network together with an iterative learning technique to represent the triangle mesh. Based on the *Growing Cell Structures* (GCS) approach [3] we introduce the *Smart Growing Cells* (SGC) network as extension to fulfill the requirements of surface reconstruction. Our method profits from the well-know benefits entailed by neural networks, like autonomy, robustness, scalability, the ability of retrieving information from very complex data, and adaptability. On the downside, typical drawbacks like undesirable smoothing of information, inability to exactly model detailed, discontinuous data, or a vast amount of computing resources at big network sizes are overcome for the application of surface reconstruction. The GCM approach creates high-quality triangulations of billions of points in few minutes. It perfectly covers any amount and distribution of samples, holes, and inconsistent data. It discovers and represents edges, manages clusters of input sample points, and it is capable of dynamically adapting to incremental sample data.

1 Introduction

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 (see Fig. 1 to get an idea of the problem). First, how can one recognize a

Hendrik Annuth · Christian-A. Bohn
Wedel University of Applied Sciences, Wedel, FRG
e-mail: annuth@fh-wedel.de, bohn@fh-wedel.de

D. Plemenos & G. Miaoulis (Eds.): Intelligent Computer Graphics 2010, SCI 321, pp. 47–66.
springerlink.com

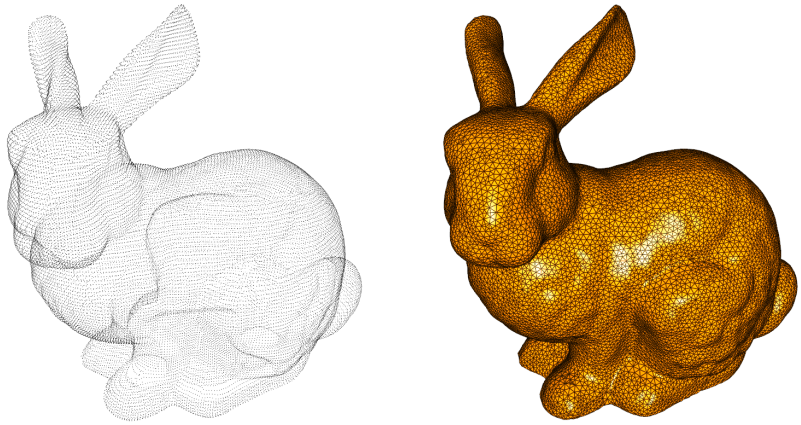

Fig. 1 Reconstruction algorithms cope with the task of recognizing a surface topology (see image on the right) from a set of arbitrary points (see image on the left) which are registered by a laser scanner or stereo photographs from real object surfaces.

topology of the originating 2D surfaces just from 0-dimensional 3D samples and without any other information from the sampled object? Second, for further processing, how is it possible to project this topological information on a data structure like a triangle mesh while meeting given constraints concerning mesh quality and size?

Although these issues have intensely been tackled since the early eighties a general concept that addresses all 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 with adequate computing resources are still a big challenge.

Previous Work

The problem of surface reconstruction is a major field in computer graphics. There have been numerous approaches with different algorithmic concepts.

An important method is [8]. They construct an implicit surface using local information from an unorganized point cloud and then compile a mesh with the marching cubes approach. In contrast [6, 17] use radial basis functions as bases for a global implicit surface. A different method for surface reconstruction is proposed in [2, 15] where a delaunay tetrahedralization of a point cloud is successively reduced until the model is carved out. Further approaches, like [7], do not create a mesh but a piecewise smooth surface which then is converted into a mesh as described for example in [20]. Another class of algorithms suggest techniques based on the Bayes' theorem, like [18, 9]. These approaches are comparable with the GCS concept (see below) and they share some advantages like noise resistance.

In the area of artificial neural networks a famous work is [14] where the *Self Organizing Map* (SOM) is proposed which iteratively adapts its internal structure — a

2D mesh — to the distribution of a set of samples. It enables clustering or dimensionality reduction of the sample data. While a SOM has a fixed number of elements, the growing cell structures concept [3, 4] allows the network for dynamically fitting its size to the sample data complexity. SOM and GCS are likely to process and represent vector data like point samples on surfaces. [5] uses a SOM and [21, 23] use a GCS for the purpose of surface reconstruction. Further improvements in this field are proposed by [13] where constant Laplacian smoothing [19] of surfaces is introduced, and in [11] the curvature described by the input sample distribution is taken to control the mesh density. In [10] the GCS reconstruction process is further enhanced in order to account for more complex topologies. [12] use several meshes of the same model for a mesh optimization process, and [22] presents a concept for combining common deterministic approaches and the advantages of the GCS approach. In [1] the GCS approach is used for focusing on mesh optimization, and in [16] a surface optimization process based on an edge swap operation for the GCS approach is presented.

In the following, we describe the basis of our approach — the growing cell structures — and then derive our idea of the smart growing cells which matches the specific requirements of reconstruction.

2 Mesh Generation with Smart Growing Cells

GCM is based on using the internal structure of a smart growing cells network as triangulation of an object surface which is described by a set of surface sample points. The SGC is a neural network approach which is derived from the growing cells structures network.

The reason for using a neural network scheme for reconstruction tasks are its obvious advantages compared to deterministic approaches.

- They robustly handle arbitrary sample set sizes and distributions which is important in case of billions of unstructured points from scanned objects.
- 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. Further, incrementally retrieved samples can be used to retrain the network without starting the triangulation process from scratch.
- They guarantee to — theoretically — find the best solution possible. Approximation accuracy and mesh quality are automatically driven to the maximum.
- Training can be designed by simple learning rules which nevertheless solve complex problems. This is important for our application case, since the converged neural network structure and the general learning algorithm only match few requirements of a reconstruction task.

Nevertheless, these advantages partly clash with our needs. 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, neural networks tend to generalize which may be advantageous from the physical point of view, but which mostly lets vanish visually important features which the human is quite sensitized to.

Growing cells meshing, from our point of view, is a breakthrough in this area. Through the use of a neural network, it delivers a general, robust, high-quality reconstruction algorithm which entails several advantages compared to classical approaches. Problems which often arise when using neural networks are solved for the application of surface reconstruction.

In the following, we outline the SGC algorithm. Then, our modifications to meet the requirements of surface triangulation are described.

2.1 Unsupervised Learning and Growing Cells

Unsupervised learning is accomplished by certain types of artificial neural networks which are able to organize its internal structure automatically depending on an arbitrary input sample distribution. After training, a set of *reference vectors* match the input sample distribution — classification is a typical task for this type of networks. In case of this work, reference vectors are interpreted as a set of vertices located on object surfaces.

Adaption of reference vectors in iterative unsupervised learning approaches is called *learning* or *training* and is generally accomplished by randomly presenting single n-dimensional samples from the input sample set to n-dimensional reference vectors and moving them in n-dimensional space (see Fig. 2).
Surface reconstruction with pure unsupervised learning would place a set of reference vectors on object surfaces, but does not determine information about the underlying surface topology. This leads to the Kohonen self organizing map described in the following.

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

repeat

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

Determine reference vector $\mathbf{c}_b$ (the "best matching vector" or the "winning unit") which lies closest to $\mathbf{s}_j$

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

Decrease ε_{bm}

until $\varepsilon_{bm} \leq$ a certain threshold ε_0

Fig. 2 Training loop of general unsupervised training

Kohonen Self Organizing Map

The SOM is based on reference vectors which are randomly connected in a regular 2D mesh. The learning rule is extended to account for the direct neighborhood of a best matching unit (see Fig. 3)

for all $\mathbf{c}_{nb} \in$ neigborhood of $\mathbf{c}_b$ **do**

Move $\mathbf{c}_{nb}$ in the direction of $\mathbf{s}_j$ according to a certain strength ε_{nb}, like $\mathbf{c}_{nb}^{\text{new}} = \mathbf{c}_{nb}^{\text{old}}(1-\varepsilon_{nb}) + \mathbf{s}_j \cdot \varepsilon_{nb}$

Decrease ε_{nb}

end for

Fig. 3 The "neighborhood loop" of the Kohonen SOM to be added at the general unsupervised learning scheme

Insertion of this "neighborhood loop" into the general unsupervised learning algorithm (after moving of $\mathbf{c}_b$ in Fig. 2) leads to the phenomenon that the reference vertices are moved by accounting for the regular 2D mesh topology of the SOM. For example, training a plane-like arranged 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, mesh size of a SOM is fixed and cannot adjust to the sample structure complexity. The growing cell structures approach overcomes this drawback.

Growing Cell Structures

To a certain degree, GCS may be seen as SOM which additionally is capable of growing and shrinking according to the problem under consideration defined by the sample distribution. This mechanism is based on a so called *resource term* contained in every reference vector, which is a simple counter. It counts the reference vector being a best matching unit and a high counter value signalizes the requirement of insertion of new reference vectors in that region. Generally, "resource" may defined differently, for example by accumulating an approximation error.

But using a GCS for surface reconstruction still exposes vital problems. There is only one given type of topology available. In other words, starting training with a regular mesh can best approximate the topology of plane-like structures, and a tetrahedron would be adequate for sphere-like objects — only objects which are homeomorphic to the start object can be represented satisfactorily. Further problems arise at discontinuities like sharp edges and holes. These are commonly difficult to model through neural network type algorithms like mentioned above. This brings us to our approach, the smart growing cells.

2.2 Smart Growing Cells for Reconstruction

Smart Growing Cells overcome the most problems mentioned above by introducing the three basic mechanisms of (a) *aggressive cut out* (ACO), (b) *discontinuity vertices*, and (c) a specific curvature criterion, explained below.

The SGC basic structure is identical to general GCS. There are n-dimensional cells which we now term *neural vertices* connected by *links* through an m-dimensional topology. We 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.

2.2.1 General Training

The simplified main training loop is very similar to to the classical GCS approach (see section 2.1) and outlined in Fig. 4. For adaption of the neighboring vertices, a smoothing process like described in [13] and [19] 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. A *vertex split* and an *edge collapse* operation adjust the network size dynamically. See Fig. 5 to get an idea of what happens in surface reconstruction with the SGC approach.

repeat

for $j = 1$ to k_{del} **do**
for $i = 1$ to k_{ins} **do**

Choose a sample randomly, find closest neural vertex $\mathbf{c}_b$ and move it with its neighbors towards the sample

Increase signal counter of $\mathbf{c}_b$ (the resource term mentioned above) and decrease the signal counters of all other vertices

end for

Find the best performing neural vertex — which is the one with the highest signal counter value — and add a new vertex at this position

end for

Find the worst performing neural vertices, delete them and affected edges

until a certain limit like approximation error, or number of vertices is reached

Fig. 4 The main loop of the SGC approach is similar to that of the GCS

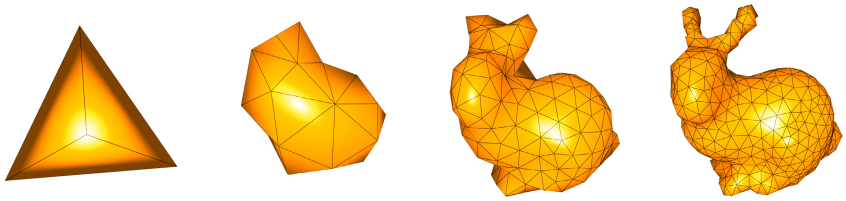

Fig. 5 Given a distribution of samples the SGC develops an initial simple polyeder into a final mesh which represents the underlying object topology

Vertex Split

A 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. 6).

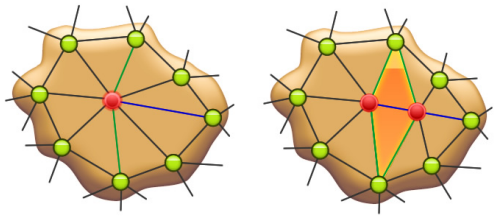

Fig. 6 Vertex split operation (from left to right) to increase mesh granularity locally. Edge collapse runs inversely.

Edge Collapse

Neural vertices with resource terms below a threshold r_{min} are removed together with three edges and two connected faces (see Fig. 6). Determination of the edge to be removed is driven by connectivity irregularities as proposed in [13].

2.2.2 Aggressive Cut Out

Aggressive Cut Out may be the most important new scheme we added to the standard GCS algorithm for achieving sufficient flexibility to match any topology of the training samples (the "homomorphism problem"). Before the edge collapse operation is applied to a vertex, it is tested if the vertex is contained in a *degenerated mesh region* (definition follows below). If so, the ACO process deletes it and additionally all connected faces.

It has been shown that degeneration of a part of the mesh serves as perfect indicator for a mesh topology which does not fit the underlying sample distribution correctly. For example, consider a region where sample density equals zero. Although vertices are not directly drawn into it by training adjustment, their neighbors may be moved there through their mesh connections. After a certain time, the resource term triggers the deletion of these vertices by edge collapse operations but

their links remain alive. These links mistakenly represent the existence of a topology and their structure is degenerated, i.e., it usually shows a surpassing number of edges with *acute-angled*[1] vertices (see Fig. 7).

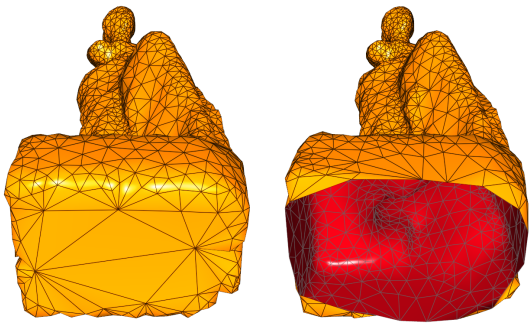

Fig. 7 The statue's bottom on the left is not represented by samples. Its acute-angled triangles refer to a degenerated mesh region. On the right, ACO has detected this region and deleted the related mesh elements.

The term "aggressive" is chosen since triggering properties are matched during training more often — suspicious neural vertices are cut out early.

Criterion for Degenerated Mesh Regions

In [10] 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, triangles generated from anisotropic sample densities are mistakenly interpreted as degenerated mesh regions. Our proposal is a combination of *vertex valence*[2], triangle quality, and quality of neighboring vertices. If all of the following conditions hold, an ACO is started at the vertex to be deleted.

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 then n_{degnb} neighbors for which conditions (1) or (2) hold.

The latter condition says that ACO is only started if at least one or two neighbors show 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 an ACO

Obviously after an aggressive extinction of a neural vertex and its surrounding faces, a boundary will be left behind which may consist of unfavorable mesh structure elements. Curing detects structures like a "spike", a "nasty vertex", a "needle eye", or a "bridge" along a boundary and patches them.

[1] 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 ε_{acute}.

[2] The term vertex valence represents the number of connected vertices.

Spike. A boundary vertex with a valence of 2 (see Fig. 8) is termed spike. Such a vertex is very unlikely supporting a correct reconstruction process since it will be adjusted to an acute-angled triangle after few more iteration steps. A spike must be deleted in any case.

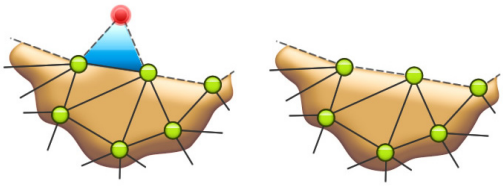

Fig. 8 Curing a boundary which contains a "spike" (see image on the left) results in the deletion of the involved vertex (see image on the right).

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. 9). These vertices are suspected to be part of a degenerated mesh region and are deleted.

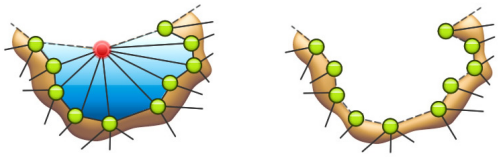

Fig. 9 A "nasty vertex" is a vertex with too many neighbors (see image on the left) and has to be deleted (image on the right).

Needle Eye. A needle eye is a neural vertex that is connected to at least two boundaries (see Fig. 10). At these locations the mesh does not have a valid mesh structure. To delete a needle eye, all groups of connected faces are determined. From these, the group with the most faces is kept and all others are deleted.

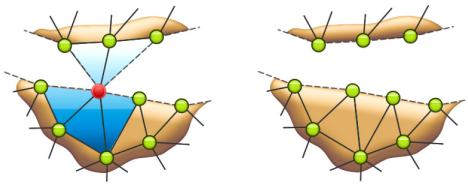

Fig. 10 A single vertex connected to two separate boundaries is termed a "needle eye" (see image on the left) and has to be deleted (see image on the right).

Bridge. A bridge is very likely to be part of a degenerated mesh region. If a mesh has a hole that consist of three vertices, then it would soon be closed by a coalescing process (see section 2.2.3). This is not allowed if exactly one of the edges of this hole is additionally connected to a face (which we term "bridge", see Fig. 11) since an invalid edge with three faces would arise. The entire bridge structure is deleted and the hole will be closed with a new face.

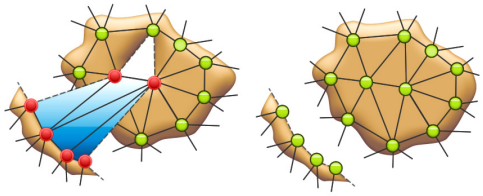

Fig. 11 If an edge of a hole is connected to a second boundary and the hole is to be closed (termed a "bridge", see image on the left), then the connection to the second boundary should be destroyed (see image on the right)

Multiple Boundary Search Through

After deletion of a neural vertex by the ACO process the curing mechanism will look for unfavorable structures along the boundary. There is more than one boundary to be considered, if the deletion destroys a coherent set of faces and multiple separate groups of faces arise due to other deletion processes.

Four cases may appear. First, the usual case with no additional boundaries. Second, when a needle eye is destroyed, the boundaries of all groups of connected faces need to be tested. Third, when surrounding faces of a vertex are interrupted by boundaries. And fourth, when a needle eye is connected to the surrounding faces of a vertex (see Fig. 12).

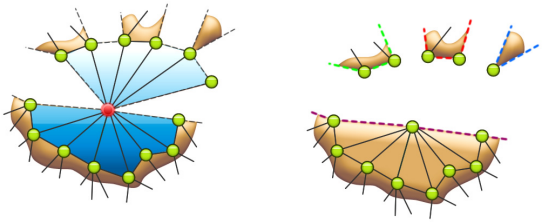

Fig. 12 Example for the cut out of a needle eye at a row of faces. Faces are not necessarily connected to other faces.

2.2.3 Coalescing

Like the mesh can be split by deletion of vertices or by the ACO process, it must also be possible to merge two mesh boundaries during the training process. For that, a coalescing test is accomplished each time a boundary vertex is moved.

Coalescing Test

This test recognizes if two boundaries are likely to be connected to one coherent area. For that, a sphere is created with the following parameters. Given the neigboring boundary vertices $\mathbf{v}_1$ and $\mathbf{v}_2$ of $\mathbf{c}_b$, then we define $\mathbf{c} = \mathbf{1/2}(\mathbf{v_1} + \mathbf{v_2})$. 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 a 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 the dot product of the boundary normals at $\mathbf{c}_b$ and $\mathbf{q}$ is negative.

Coalescing Process

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 — "corner" and "long side" — which have to be considered.

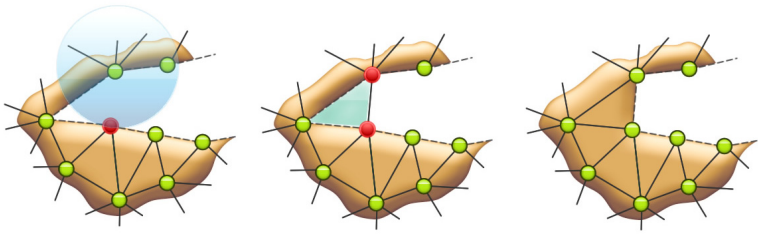

Fig. 13 Coalescing process at a mesh corner. On the left, the search process of a coalescing candidate. In the middle, one edge is created, on the right, the only face capable of being added is the corner face.

Corner. A corner of the same boundary arises when $\mathbf{c}_b$ an $\mathbf{q}$ have one neighboring vertex in common. Here, it is only possible to generate the face which closes the corner (see Fig. 13). Otherwise a needle eye would be created which is not a satisfactory mesh structure.

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 Fig. 14b). The triangle with edge lengths which vary fewest is taken in our approach (see Fig. 14c) 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, we take, the face with the highest edge similarity (see Fig. 14d). Coalescing

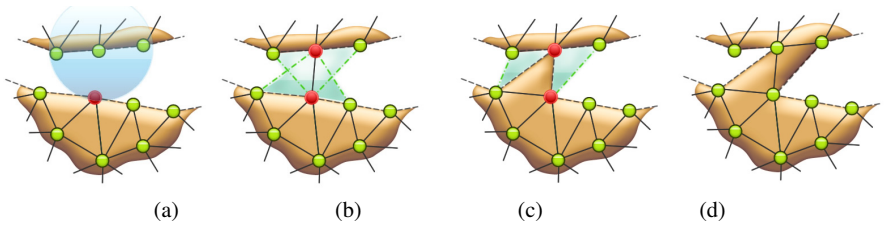

Fig. 14 Coalescing of two separate boundaries. In (a) and (b) the edge is determined, in (c) the triangle with the smallest variance of edge lengths is added, in (d) another triangle is created to avoid a needle eye.

sometimes creates unfavorable bridges that need to be deleted through a further ACO process.

ACO is a vital scheme which allows for changing the topology of the initial SGC mesh to match the topology of the underlying object. If ACO sees inconsistencies in the mesh it clears them by destroying parts of the mesh at the involved locations. Coalescing is the counterpart of this deletion. It forms new connections to create new topological structures (see Fig. 15).

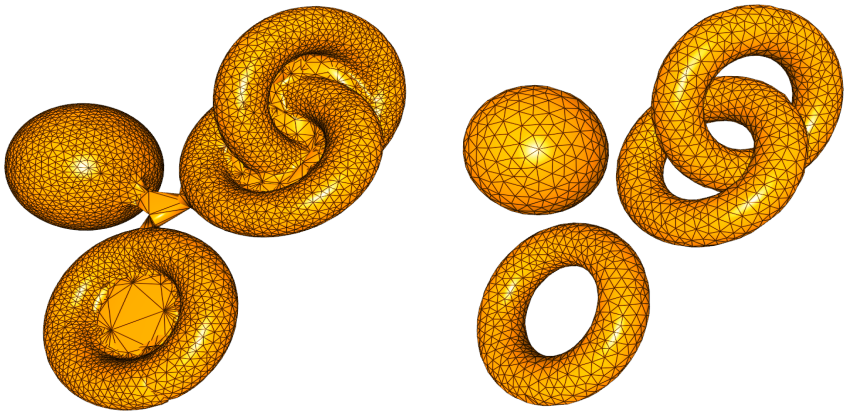

Fig. 15 Aggressive cut out detects inconsistent regions in the SGC mesh (visible in the image on the left side at the centers of the tori), deletes them and melts the remaining boundaries (see image on the right side). Thus, the mesh topology is adapted during training.

2.2.4 Granularity Adaption

Up to now, the SGC is able to approximate an arbitrary sample set by a 2D mesh. Now we present an efficient local adaption scheme of the mesh density in a way that areas with a strong curvature are modeled by a finer mesh resolution (see Fig. 16). 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 represented by a uniform sample distribution.

Each time a vertex is adapted by a new sample we calculate the estimated normal $\mathbf{n}_k$ at a neural vertex $\mathbf{v}_k$ by the average of the normals at the surrounding faces. The curvature $c_k \in \mathbb{R}$ at a vertex is determined like

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

with the set $\mathcal{N}_k$ containing the normals of the neighboring neural vertices of $\mathbf{v}_k$. Each time a neural vertex is chosen as winner, its curvature value is calculated and a

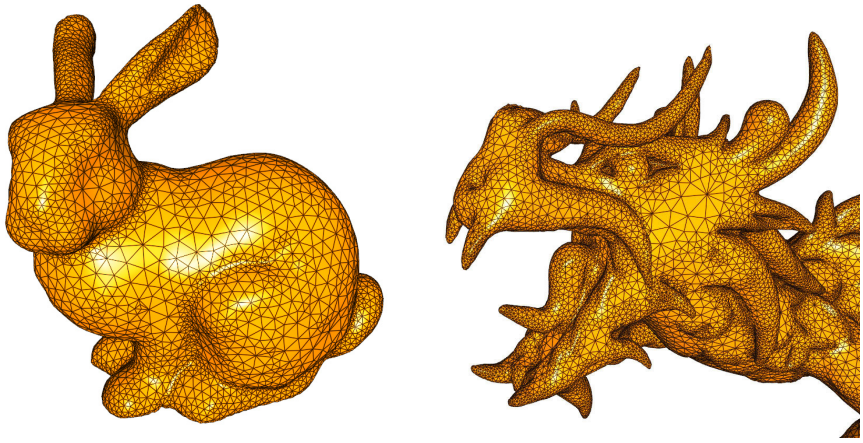

Fig. 16 Granularity adaption correlates surface curvature to mesh density. See for example, the accentuated mesh density at the bunny's leg compared to its torso (left side), or the dragon's horns compared to its cheek (right side)

global curvature value $\overline{c}$ is adjusted. Finally, the curvature dependent resource term r_k at $\mathbf{v}_k$ is adjusted through $r_k^{new} = r_k^{old} + \Delta r_k$, with

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

and the deviation σ_{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 low curvature.

2.2.5 Discontinuity Modeling

A sampled model that exposes discontinuities like edges is difficult to be approximated by a 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 the 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 handling vertices at regions classified as discontinuities of the underlying scanned object differently — first, they are only allowed to move in the direction of an object edge to represent it more properly, and second, the smoothing process from section 2.2.1 is not applied at them (see Fig. 17, the same model with and without discontinuity modeling).

Recognizing these *discontinuity vertices* is accomplished as follows. We determine the curvature values of those neighbors which have a distance of two

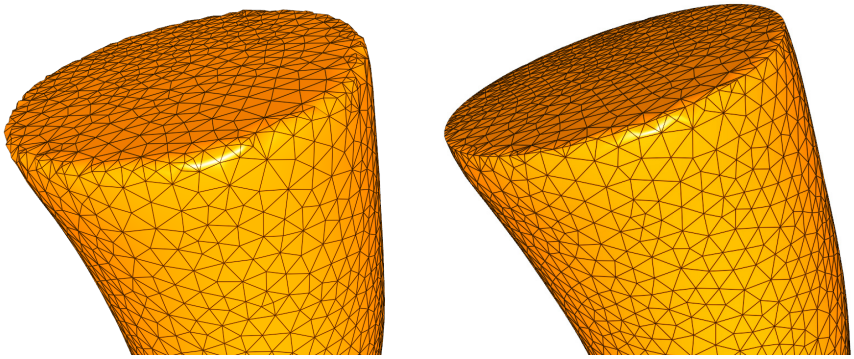

Fig. 17 Discontinuity modeling lets vertices move mainly in the direction of discontinuities. On the left, a result from the SGC approach without discontinuity modeling is exposed, on the right, with discontinuity modeling cells have grown into the corner of the plate.

connections from the vertex **v** under consideration — the "second ring" of neighbors. Then the average δ_{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, then a discontinuity at this location is assumed if the average of the neighbors' (second ring) curvature gradient differs to a certain amount, i.e., if

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

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

For approximating the edge normal at a discontinuity vertex **v**, we take the average of the normals of two of the neighboring vertices, either those with the highest curvature value, or those which are already detected as being a discontinuity vertex. Finally, the normal is mirrored if the edge angle lies above 180°, which is indicated by the average of the surrounding vertex normals; in the first case it points in the direction of **v**.

Fig. 18 shows two further examples for the discontinuity modeling facility of GCM.

3 Results

In Fig. 19 the whole GCM algorithm is listed as pseudocode. To keep it comprehensive, the outermost loop of the algorithm is neglected, vertex split and edge collapse operations are triggered by using counters (compared with the basic algorithm from section 2.2.1).

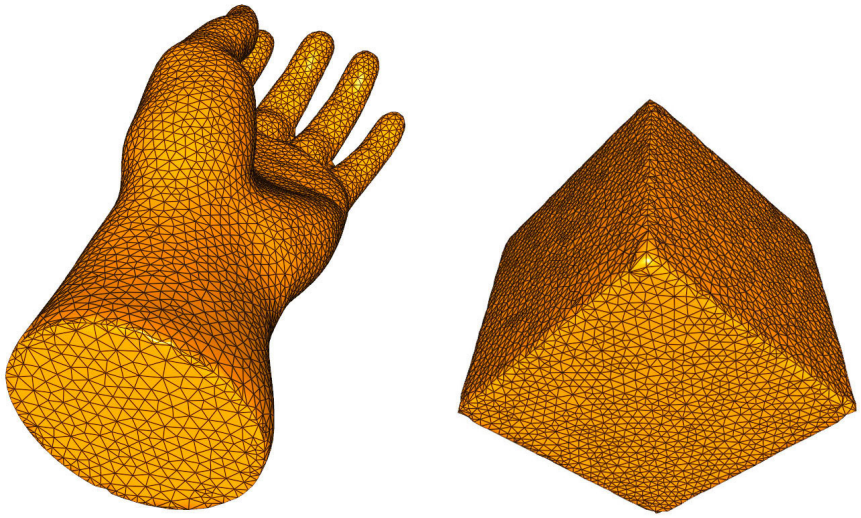

Fig. 18 Two example objects which expose the discontinuity modeling facility of the SGC

Parameters

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

Test Hardware

We used a *Dell*® Precision M6400 Notebook with *Intel*® Core 2 Extreme Quad Core QX9300 (2.53GHz, 1066MHz, 12MB) processor with 8MB 1066 MHz DDR3 Dual Channel RAM. The algorithm is not parallelized.

Some visual results are exposed in Fig. 21. All pictures are drawn from one SGC mesh and they are rendered using Phong Shading. Most models stem from the *Stanford 3D Scanning Repository*.

Besides visual results, GCM comes up with impressive numbers, listed in Fig. 20. 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 result, nevertheless this is typically expected when using an approach from the field of unsupervised learning, since it guarantees an ideal representation of the underlying training sample distribution.

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 large error terms. It proves that discontinuity vertices are worth it. Also the computing times needed are very short, i.e., few minutes in almost all cases.

```
Adjust samples regarding curvature
Calculate average curvature and deviations
Determine and sign discontinuity vertices
for all Boundary vertices do
  if ∃ coalescing candidate then
    Melt boundary
    for all ACO candidates do
      Clean by ACO
    end for
  end if
end for
if Edge collapse operation triggered then
  Collapse edge
  for all ACO candidates do
    ACO-cleaning
  end for
end if
if Vertex split operation triggered then
  Split vertex
end if
```

Fig. 19 The complete GCM algorithm. The outermost loop of the algorithm is neglected, vertex split and edge collapse operations are triggered by counters

All those measurements seem to be far better than those from classical approaches, as long as we could extract them from the regarding papers. The presented algorithm works very robustly. There are nearly no outliers visible in the mesh.

4 Summary and Future Work

We presented a new neural network approach for surface reconstruction purposes — the smart growing cells. The algorithm, on the one hand, profits from the common outstanding facilities of neural networks, on the other hand, well-known problems when using such iterative techniques are avoided successfully.

Growing cells meshing is capable of recognizing and representing arbitrary surface topologies. The network training generates valid meshes at any size and it is able to handle 3D point samples at an arbitrary amount. The mesh accounts for discontinuities like sharp edges and holes, smoothes over missing sample data, handles difficult distributions of sample data, and its granularity efficiently adapts to the underlying curvature of the object shape. The iterative approach enables dynamically changing the training sample set to adjust the network to small changes in a shorter time.

To achieve this, we introduced several mechanisms like, first, aggressive cut out, which can be seen as a brute cleaning mechanism for ill-formed structures. We

	Samples	Vertices	Time [min:sec]	Quality [Delaunay]	RMS/ Size
	36K	30K	1:19	95.6%	4.7e-5
	438K	100K	5:33	95.5%	3.3e-5
	544K	260K	18:33	93.1%	1.7e-5
	14,028K	320K	24:27	98.5%	1.3e-5
	5,000K	500K	42:10	95.9%	2.7e-5
	511K	10K	0:21	99.8%	6.6e-5
	38K	5K	0:12	99.0%	15e-5
	346K	5K	0:12	98.3%	0.7e-5

Fig. 20 Results with objects from the Stanford 3D Scanning Repository. "Quality" means the percentage of triangles which hold the Delaunay criterion. RMS/Size is the root of the squared distances between the original point samples and the triangle mesh, divided by the longest edge of the maximum expansion of the sample set.

found out that these structures are also a great hint for changing connectivities to match the underlying surface topology. Second, face normals are regarded and included in the neural network training loop to assimilate mesh granularity and to make the reconstruction process independent from the sample distribution. Third, we proposed a specific learning strategy for vertices which lie in a sample region which represents discontinuities of the scanned model.

The proof of concept of our approach is enriched by the achieved quality and performance measures. For the tested geometries which each hold specic challenges of reconstruction, we get approximation errors for comparable mesh resolutions that lie far below 1% at average. Mesh quality, measured by the percentage of triangles which comply the Delaunay criterion, lies at 96% at average, and the time needed to compute meshes of several hundreds of thousands of polygons is just few minutes. Thus, with these results, we are confident that we could show that the growing cells meshing approach is a considerable alternative to common reconstruction methods.

Future Work

There are some object structures which can hardly be modeled adequately by the presented method and which will be challenged in future work. These are, for example, planes which are thinner than the average sample distance. From the SGC point of view, these surfaces are not clearly distinguishable.

Computation times are that small that one could think of a real-time reconstruction approach. Scanning users in VR applications for sending their shape to other

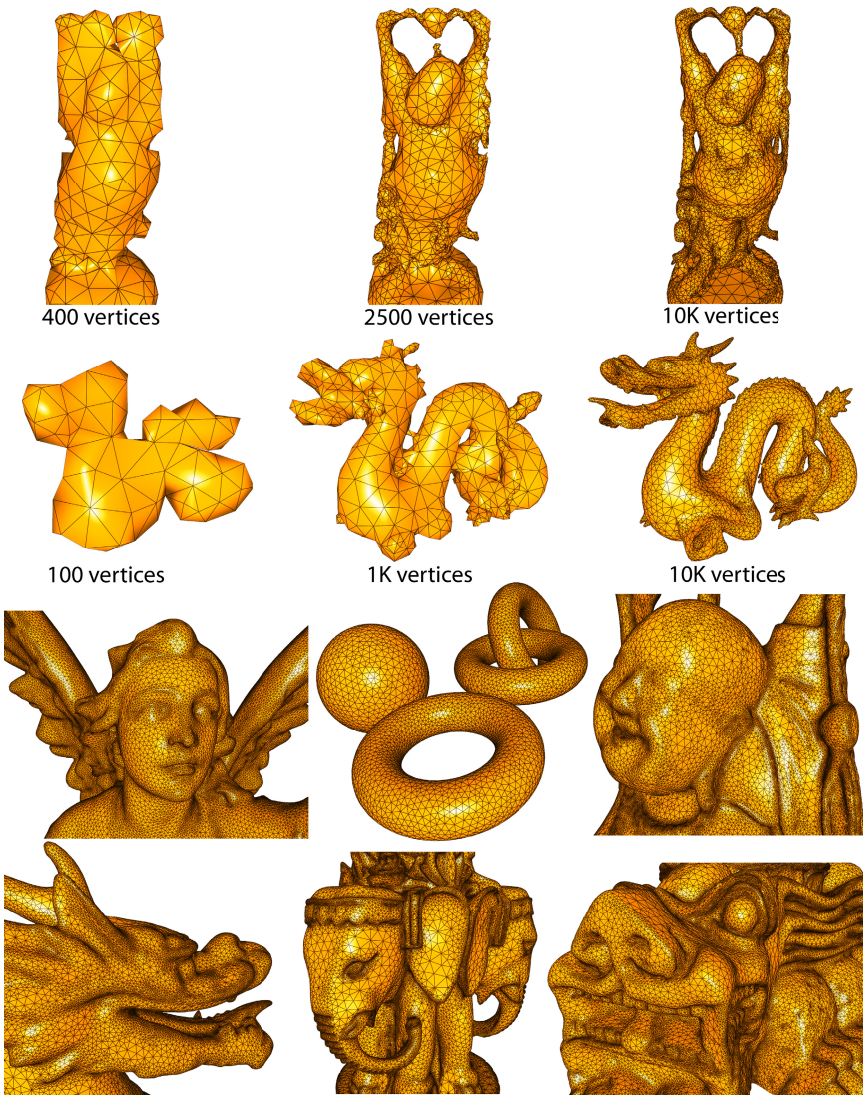

Fig. 21 The top two lines show each three states during a complete training process together with the number of vertices which have been created at that time. The lower lines expose some pictures of reconstructed models.

places could be a impressive feature. Here, an effective parallelization of the algorithm would be required and could be accomplished by executing sample adjustment on several processing units concurrently.

Generally, the SGC approach should be able to bring a lot of new ideas because of its flexibility and since it is capable of learning arbitrary sample sets in a very short time.

Acknowledgements. The authors would like to thank Kai Burjack for his support in rendering the presented images.

References

1. Álvarez, R., Noguera, J.V., Tortosa, L., Zamora, A.: A mesh optimization algorithm based on neural networks. Inf. Sci. 177(23), 5347–5364 (2007)
2. Edelsbrunner, H., Mcke, E.P.: Three-dimensional alpha shapes (1994)
3. Fritzke, B.: Growing cell structures - a self-organizing network for unsupervised and supervised learning. Neural Networks 7, 1441–1460 (1993)
4. Fritzke, B.: A growing neural gas network learns topologies. In: Tesauro, G., Touretzky, D.S., Leen, T.K. (eds.) Advances in Neural Information Processing Systems 7, pp. 625–632. MIT Press, Cambridge (1995)
5. Hoffmann, M., Vrady, L.: Free-form surfaces for scattered data by neural networks. Journal for Geometry and Graphics 2, 1–6 (1998)
6. Hoppe, H.: Poisson surface reconstruction and its applications. In: SPM 2008: Proceedings of the 2008 ACM symposium on Solid and physical modeling, p. 10. ACM, New York (2008)
7. Hoppe, H., Derose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schweitzer, J., Stuetzle, W.: Piecewise smooth surface reconstruction, pp. 295–302 (1994)
8. Hoppe, H., DeRose, T., Duchamp, T., McDonald, J.A., Stuetzle, W.: Surface reconstruction from unorganized points. In: Thomas, J.J. (ed.) SIGGRAPH 1992, pp. 71–78. ACM, New York (1992)
9. Huang, Q.-X., Adams, B., Wand, M.: Bayesian surface reconstruction via iterative scan alignment to an optimized prototype. In: SGP 2007: Proceedings of the fifth Eurographics symposium on Geometry processing, pp. 213–223. Eurographics Association, Aire-la-Ville, Switzerland (2007)
10. Ivrissimtzis, I., Jeong, W.K., Lee, S., Lee, Y., Seidel, H.P.: 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 (2004)
11. Ivrissimtzis, I., Jeong, W.K., Seidel, H.P.: Neural meshes: Statistical learning methods in surface reconstruction. Tech. Rep. MPI-I-2003-4-007, Max-Planck-Institut fr Informatik, Saarbrücken (2003)
12. Ivrissimtzis, I., Lee, Y., Lee, S., Jeong, W.-K., Seidel, H.-P.: Neural mesh ensembles. In: 3DPVT 2004: Proceedings of the 3D Data Processing, Visualization, and Transmission, 2nd International Symposium, pp. 308–315. IEEE Computer Society, Washington (2004)
13. Ivrissimtzis, I.P., Jeong, W.K., Seidel, H.P.: Using growing cell structures for surface reconstruction. In: SMI 2003: Proceedings of the Shape Modeling International 2003, p. 78. IEEE Computer Society, Washington (2003)
14. Kohonen, T.: Self-Organized Formation of Topologically Correct Feature Maps. Biological Cybernetics 43, 59–69 (1982)

15. Kolluri, R., Shewchuk, J.R., O'Brien, J.F.: Spectral surface reconstruction from noisy point clouds. In: SGP 2004: Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, pp. 11–21. ACM, New York (2004)
16. Mari, J.F., Saito, J.H., Poli, G., Zorzan, M.R., Levada, A.L.M.: Improving the neural meshes algorithm for 3d surface reconstruction with edge swap operations. In: SAC 2008, pp. 1236–1240. ACM, New York (2008)
17. Muraki, S.: Volumetric shape description of range data using "blobby model". In: SIGGRAPH 1991: Proceedings of the 18th annual conference on Computer graphics and interactive techniques, pp. 227–235. ACM, New York (1991)
18. Storvik, G.: Bayesian surface reconstruction from noisy images. In: Interface 1996 (1996)
19. Taubin, G.: A signal processing approach to fair surface design. In: SIGGRAPH, pp. 351–358 (1995)
20. Tournois, J., Wormser, C., Alliez, P., Desbrun, M.: Interleaving delaunay refinement and optimization for practical isotropic tetrahedron mesh generation. In: SIGGRAPH 2009: ACM SIGGRAPH 2009 papers, pp. 1–9. ACM, New York (2009)
21. Vrady, L., Hoffmann, M., Kovcs, E.: Improved free-form modelling of scattered data by dynamic neural networks. Journal for Geometry and Graphics 3, 177–183 (1999)
22. Yoon, M., Lee, Y., Lee, S., Ivrissimtzis, I., Seidel, H.P.: Surface and normal ensembles for surface reconstruction. Comput. Aided Des. 39(5), 408–420 (2007)
23. Yu, Y.: Surface reconstruction from unorganized points using self-organizing neural networks. In: IEEE Visualization 1999, Conference Proceedings, pp. 61–64 (1999)