Photogrammetrie • Fernerkundung • Geoinformation 3/2008, S. 207–215, 2 figs.

From Detailed Digital Surface Models to City Models Using Constrained Simplification

ROLAND WAHL, RUWEN SCHNABEL & REINHARD KLEIN, Bonn

Keywords: DSM, City Model, Geometry Simplification, Abstraction, Visualization.

Summary: We present a method to simplify high-detail full-featured digital surface models (DSM) of cities (i. e., containing the heights of trees, cars, buildings, etc.) geometrically in such a way that all relevant features are preserved, whereas noise and superfluous details collapse. The relevance of features is automatically evaluated using a semantically motivated shape detection and serves as constraint during the simplification. Our results show that we are able to preserve fine details of complex roof structures while all irrelevant features are effectively removed. Thus, we achieve an excellent abstraction of the city data without any interaction of the user, which is not only beneficial for visualization, but could also be used for GIS related applications.

Zusammenfassung: *Von digitalen Oberflächenmodellen zu Stadtmodellen mittels eingeschränkter Simplifizierung*. Wir stellen ein geometrisches Simplifizierungsverfahren für hoch detaillierte ungefilterte digitale Oberflächenmodelle (DOM) von Städten (inkl. Abtastwerten von Bäumen, Autos, Gebäuden, etc.) vor, welches alle relevanten Merkmale erhält, aber Rauschen und überflüssige Details verwirft. Die Relevanz der Merkmale wird automatisch mittels semantisch motivierter Formerkennung bewertet und dient als Einschränkung der Simplifizierung. Unsere Resultate zeigen, dass wir in der Lage sind, feine Details komplexer Dachstrukturen zu erhalten, während irrelevante Merkmale effektiv eliminiert werden. Auf diese Weise erreichen wir ohne jegliche Benutzerinteraktion eine ausgezeichnete Abstraktion der Stadtdaten, die sich nicht nur für Visualisierungszwecke eignet, sondern auch in GIS-Applikationen benutzt werden könnte.

1 Introduction

Conventionally, the sole aim of geometry simplification in the context of real-time visualization of landscape or urban environments is to enable smooth, real-time navigation through the scene without disturbing interruptions for data loading or decoding. To this end, the simplification generates a suitable set of geometric levels of detail (LOD) of the terrain data. To sustain a satisfactory user experience, the blending in of additional detail without notable flickering or jumps while the user zooms in on objects should be supported by the underlying LOD structure. Hence, the LODs are usually generated with respect to a geometric error measure, e. g., Hausdorff error that guarantees pixel correct images at given viewing distances. However, often additional requirements arise when dealing with city-data:

- In the context of city visualization, a photorealistic visualization is not always desired, e. g., on a small PDA or cell phone display, the overwhelming amount of detail is difficult to grasp for the user and an abstracted view is usually preferred. The abstraction however should be semantically motivated and cannot be based on geometric error alone.
- A lot of existing GIS related software, e. g., for city-planning, operate on abstracted data in the form of CityGML or similar formats. To this day no automatic

1432-8364/08/2008/207 $ 2.25

conversion of height-field data into this representation is available.
- For city visualization in a client-server setting over the internet, as it is available in a primitive form in Google Earth today, it is usually not possible to transmit all the necessary detail of geometry and texture in the short time available while the user navigates through the scene because of limited bandwidth. Therefore it is unavoidable that the user will frequently see coarser LODs from a distance where the simplifications therein become clearly visible (i. e., larger than a couple of pixels). Current simplification methods for high resolution height-field data however are only based on geometric error considerations, so that often façades of houses are askew or roofs have unnatural looking shapes, which results in views that are irritating to the user.

All of these requirements are not vital as long as we deal with city models derived from cadastral data or semi-automatic reconstruction, given that these models are generally reasonably abstract. However, considering the ongoing advances in camera and reconstruction techniques and the consequently increasing detail and extent of full-featured digital surface models (DSM), automatic abstraction methods capable of handling out-of-core data are necessary.

In order to address this situation, in this work, we propose a novel form of constrained simplification that incorporates additional shape information together with geometric error considerations to generate LODs from highly detailed DSMs that respect both geometric as well as semantically motivated criteria. The incorporated shape information is low-level and very general. It is used to find edges and corners in the geometry that make up the important features of building geometry without resorting to more involved and specialized building models. The simplification is constrained to preserve these features even in coarse LOD. Due to the continuous nature of the LOD and the consideration of geometric error, this representation is still suitable for real-time pixel correct photorealistic terrain and city renderers. Moreover, due to the preservation of important edges and corners, it is applicable in client-server settings on the internet or visualization on mobile devices as well – all from the same data representation and generated fully automatically.

2 Previous Work

In Computer Graphics, LOD representations of objects and scenes have been extensively researched during the last 15 years. In combination with methods for efficient occlusion calculations, image based rendering as well as prediction and caching mechanisms they are employed for efficient visualization of large scenes.

2.1 *Topology-Preserving Simplification*

Even for triangulated height fields it is challenging to find an optimal approximating mesh with a given small number of faces in the sense of the L1-norm. Indeed Agarwal & Suri (1994) have proven this problem to be NP-complete. Therefore, iterative greedy algorithms have prevailed which in each simplification step either eliminate a vertex (vertex contraction) or an edge (edge collapse) from the triangulation (Schroeder et al. 1992 and Hoppe et al. 1993). Several different error measures have been proposed and evaluated in the literature. Compared to other distance measures, the Hausdorff metric has the advantage that the projection of the 3D approximation tolerance onto screen space can be used to select a corresponding LOD automatically for pixel correct rendering (Klein et al. 1996). The quadric error metric introduced later by Garland & Heckbert (1997) has the advantage of a simpler and more efficient computation. Therefore, it has become very widespread, although it does not guarantee any bounds on the screen space error. Since then, there were also improvements in computing fast Hausdorff distance approximations (Cignoni et al. 1998 and Guthe et al. 2005).

2.2 *Topology-Changing Simplification*

The family of *vertex clustering* methods has been introduced by (Rossignac & Borrel 1993) and has been refined in numerous more recent works (Kok-Lim & Tiow Seng 1997). The algorithms of this family essentially apply a 3D grid to the object and for each cell contract all the vertices inside the cell. This way holes in objects are closed or objects in close proximity are merged. Although the degenerate faces are subsequently removed, it is difficult to influence the fidelity of the result due to lack of control over the induced topological changes. The already mentioned *vertex contraction* operator (Garland & Heckbert 1997 and Popovic & Hoppe 1997) offers more control over the topological modifications. However, without further processing it possibly generates non-manifold meshes.

2.3 *Out-of-Core Simplification*

To simplify models of ever increasing size, a number of out-of-core simplification algorithms have been developed. El-Sana & Yi-Jen (2000) sort all edges according to their lengths and use this ordering as decimation sequence. Lindstrom (2000) uses vertex clustering to reduce the number of vertices. As the representing position of each vertex cluster is computed from an accumulated quadric error metric, the memory requirement of the algorithm is proportional to the size of the output model. For cases where neither input nor output model fit into main memory, an out-of-core vertex clustering (Lindstrom & Silva 2001) was developed. The multiphase algorithm (Garland & Shaffer 2003) uses vertex clustering to reduce the complexity of the input model followed by a greedy simplification approach and achieves high quality results. Another way for out-of-core simplification is to split the model into smaller blocks, simplify these blocks and stitch them together for further simplification. In (Hoppe 1998) this approach is applied to terrain and in (Cignoni et al. 2003) to arbitrary meshes. The approach has the problem that special care has to be taken at patch boundaries. Recently, stream decimation algorithms (Wu & Kobbelt 2003; Isenburg et al. 2003) for out-of-core simplification have been developed, but the resulting model is not optimal with respect to mesh size and Hausdorff distance of the simplified model to the original.

2.4 *Remeshing*

Another area related to our approach is remeshing of triangulated geometry. Remeshing algorithms take a triangle mesh and resample it such that some quality requirements are satisfied but the original geometric shape is retained. In this sense, mesh simplification can be seen as a special case of remeshing. Other remeshing techniques include surface fairing (Taubin 1995 and Desbrun et al. 1999), where connectivity is preserved but vertex positions are optimized in order to remove noise or to evenly distribute vertex positions. Hildebrandt & Polthier (2004) presented a bilateral mesh smoothing algorithm that is able to preserve edges and corners in the geometry. A similar approach is given by Vorsatz et al. (2001) who describe a remeshing algorithm that is feature sensitive. Both approaches however are not combined with simplification and are not robust to outliers.

3 Overview

Given a high-resolution height-field model, it is converted into a 3D point-cloud and decomposed by our recently proposed efficient RANSAC shape detection (Schnabel et al. 2007) into areas that correspond to primitive shapes such as planes, spheres, cylinders etc. and a set of remaining points. The points of the original DSM are then tagged with the indices of shapes detected in their proximity. These index sets then implicitly define the shape, edge or corner property of the points, which is subsequently used to constrain the geometric simplification. Only those simplification operations are allowed that respect the detected primitives on the corresponding LOD. This way it is asserted that coarse building models are

generated which obey the abstract structure defined by the segmentation into primitives. Depending on the chosen size and approximation fidelity of the detected primitives, the resulting coarse polygonal models adhere to different semantically motivated levels of detail. In areas where no primitives could be detected (e. g., areas of natural cover such as in parks), the simplification is guided by geometric error alone, which has been proven to give good results for terrain in general.

4 Geometric Simplification

We build our simplification framework around the edge-collapse operation with tight upper bounds on the Hausdorff distance against the original mesh. Each edge of the original mesh generates three collapse candidates, which are either of the two corresponding halfedge-collapses or an edge-collapse with vertex placement. As optimizing the new vertex position with regard to the Hausdorff distance, which includes evaluating the maximum, does not make sense, we use the quadric error metric (Garland & Heckbert 1997) for candidate generation. This metric is fast and easy to compute, and directly yields the optimal vertex for the edge collapse operation in general cases. For degenerate cases the distance to the original edge is used as an additional criterion. Each collapse candidate is then checked for validity, that is whether it introduces flipping of orientations or degeneration of neighboring triangles, and scheduled in a priority queue keyed to its approximation error. For the sake of speed we again use the quadric error metric for computing priorities.

After these preparational steps, iteratively the best collapse candidate is evaluated, this time using the actual distance metric and if it does not surpass the current error threshold, it will be applied to the mesh. As it changes the appearance of its 1-ring, all conflicting candidates are rescheduled or deleted from the priority queue. This process comes to an end when each remaining valid collapse operation surpasses the threshold and therefore the bottom-up simplification scheme is in a local optimum. Although this approach is greedy, it is able to collapse a mesh completely, if the distance threshold allows it (i. e., it does not get stuck in a local minimum).

4.1 *Distance Metric*

For pixel-true rendering, the Hausdorff distance is almost the perfect choice, since it guarantees two crucial properties, directly linked to its definition:

Firstly, for every feature of the original mesh, there exists a part of the proxy mesh which represents that feature within a distance of at most the predefined threshold. And secondly, as also the inverse Hausdorff hemimetric is accounted for, the resulting approximation does not introduce artifacts which have no justification from the original mesh.

The arguments against using Hausdorff distance are that it is very difficult to compute and that in many cases, simpler approximations well serve their purpose.

Especially, in the domain of terrain rendering, measuring only along the z-axis is a popular alternative. Its main advantage is that opposed to strict Hausdorff distance, the counterpart on the other mesh is implicitly given and therefore, we get two piecewise linear distance functions parameterized over the plane. It has been observed that evaluating this metric only at the vertices of the two corresponding meshes does not yield tight bounds on the Hausdorff distance, but also the edges need to be considered, as otherwise the error can become arbitrary large. It can be shown that for small maximum steepness angles α the overestimation of the distance is bounded by $\cos^{-1}\alpha$. So this does not lead significant overheads for coarsely sampled terrain datasets.

However, in the presence of high-frequency signal, which is very common in high-resolution digital surface models, this approximation is no longer effective. Therefore, we only use the implicit correspondence between the meshes as given by the z-projection and evaluate the Hausdorff distance locally between the corresponding parts. That

way, the distance computations remain local (i. e., in the 1-ring of the edge in question) and still we get tighter bounds and effective simplification of steep geometry.

5 Semantic Constraints

As mentioned in the introduction, LOD generation based on purely geometric simplification often leads to unwanted results, since it does not consider the overall shape, but only local geometric features. For terrain datasets, the resulting approximation is generally good enough, but especially for man-made objects as buildings, where the shape is often dominated by recurring patterns, geometric simplification fails to maintain symmetries and structures and is therefore not well suited as an abstraction method. Nevertheless, it has the big advantage that it always yields a complete representation of the underlying scene automatically, irrespective of whether it can interpret the scene or not. Therefore, it is desirable to combine its strengths with global semantic analysis which is able to identify important feature edges and corners in order to get the best of both worlds.

One way to have simplification respect the overall shape is via accordingly designed constraints. For this approach care must be taken that the constraints achieve the desired feature preservation and that they do not limit the effectiveness of the simplification.

In the following we will first discuss a very general method to automatically add semantically motivated metainformation to the input data. Then, we deal with how these data are used as constraints during simplification.

5.1 *Edges & Corners*

In this work, we propose to use primitive shapes to detect important edges in the height data. The reason a shape-based detection is preferred over more traditional methods such as Laplace edge detection is that the shape detection can handle outliers and noise in a robust fashion and has a more global notion of structure (i. e., based on connected components of parts with equal curvature), which enables it to detect edges reliably comprising a wide angle between two primitives, e. g., on top of a shallow roof.

As a first step, we employ the shape detection described in (Schnabel et al. 2007). As it operates on 3D point-clouds, the input height-field is first converted to 3D by insertion of additional points at discontinuities in the 2.5D data (e. g., for facades). We use the same sampling density for this vertical upsampling as in the planar domain, in order to maintain a close relation between the number of samples and surface area. The resulting point-cloud $P = \{p_1, \ldots, p_N\}$ is partitioned into subsets S_i associated to shape primitives Φ_i (i. e., planes, spheres, cylinders, cones and tori) as well as a single subset R containing any remaining points that could not be assigned to a shape for the given parameters. In order to ensure heuristically that only parameterizable patches are created, a point is considered compatible if its Euclidean distance to the shape is within a given distance threshold and its normal does not deviate from the respective shape normal by more than a given angle threshold. After removing the compatible points, the algorithm is restarted on the remaining points until no more shapes can be found for the given set of parameters.

For details on the efficient probabilistic RANSAC-based algorithm we refer to the original work, we only want to emphasize here that there are parameters which allow us to select what kind of shapes are considered valid and therefore define a low-level interface to the interpretation of the data (e. g., surface area). If wanted also more complex parameters (e. g., neighboring shapes, shape orientation) can be used to decide whether the shape is important or not (cf. Schnabel et al. 2008) or the results can be cross-validated against cadastral data. But as we aim at a high level of automatism and generality we will work with the inherent data and few parameters if possible.

In our setting, we define vertices of the DSM to be edge points if they are close to

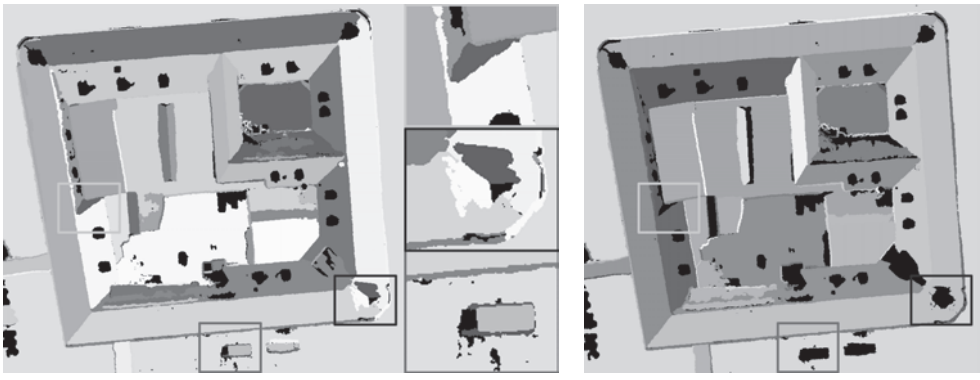

Fig. 1: Shape detection results with 4 m^2 (left) or 16 m^2 (right) size thresholds. Intensities are random, black means no shape detected. The middle column shows close-ups of a small part of the roof, a large dormer and a truck, which are no longer present in the 16 m^2 detection result.

two different shape primitives. Points that are close to even more primitives are classified as corner points. For closeness again we use a distance threshold ε, but this time we do not measure to the ideal shape but its points. That is, a point is close to Shape j if it is within ε distance of any of the points from S_j. In order to identify all edge and corner points efficiently, the point-cloud P is sorted into an axis aligned 3D grid. Then for all grid cells that contain points belonging to different shapes, the contained points' distances are compared to ε and a counter is increased for each potentially different assignment. In order to avoid discretization dependencies due to the location of the grid cells, we use eight translated versions of the grid, corresponding to the eight corners of a cube. Given the distance threshold ε, the width of the cells is set to ε and shifted versions of the grid are created with an offset of ε/2 along the respective axes. Cells are stored in a hash table, so that memory is only allocated for occupied cells. However, in order to get most out of the semantic constraints it is valuable not only to classify edges and corners, but to keep the whole information to which shape each point corresponds. This additional information will be used to not restrict simplification in the presence of features blindly, but to guide which of the possible combinations of features are allowed. This information is stored in an additional raster of shape-IDs, which is read along with the height field during simplification.

5.2 *Constrained Simplification*

In order to respect and maintain the shape information of the vertices, we pose an additional constraint to each collapse candidate during validity check (see sec. 4): The vertex which is about to collapse must ensure that its set of shape-IDs is a subset of the shape-IDs of its collapse partner.

That this simple rule maintains the vertex' shape-IDs is obvious, but how does it help in maintaining features? The principle is that a vertex, once it collapsed to a corner or edge, cannot move away from there, as it can only move along the feature. So, as the IDs are globally unique and each two planes share only one line (cf. Section 6 for discussion of non-planar shape primitives), this approach guarantees that every corner and every edge as defined by the shape-map is maintained.

But whenever a feature edge is not detected along its whole extent, or is not enclosed by two corner features, it might collapse to a single point, which is of course not the desired representation. This case occurs very often due to the presence of noise and occlusions and because of the incomplete shape segmentation. In digital surface models, this

is probably the default case. In order to cope with that situation, we suggest the use of additional topological constraints. We define border vertices of a shape as those vertices which have one incident edge pointing to a vertex that is not in the same shape. Such vertices are not allowed to move inside the shape, but may only collapse to neighboring border vertices. This constraint can be checked by looking at the shape-IDs of the two tip vertices of the incident triangles. One of these must be outside of the shape if the collapse takes place at the border. As opposed to labeled edge vertices, this criterion does not allow finding a low-error approximation within a defined small range in the proximity of the hypothetical intersection, but the purpose of maintaining the border is served and still effective complexity reduction along the border is possible.

Now there remains one situation in which detected features still might degenerate, namely if two edges of the same shape do not meet in a common corner but are connected via a series of border vertices. As the collapses along each of the two edges are legitimate, they may again collapse to their next corners respectively introducing an unwanted shortcut edge. We deal with this problem by detecting the implicit corners defined as those edge vertices which only have one neighboring border vertex with respect to one of their shapes. Implicit corners are then treated as corners and may not be collapsed to other vertices unless they are of the same corner type.

6 Results

We tested the proposed methods on a 256 m × 256 m part of a highly detailed digital surface model of downtown Berlin, featuring complex buildings at an input resolution of 12.5 cm (resampled from the 7 cm resolution dataset courtesy of DLR). The original heightmap therefore contained 4.2 million vertices. After adding the façade points, the point-cloud had more than 11 million points. The shape detection took 197 sec. and resulted in 1,658 planes larger than 4 m^2 and 695 planes larger than 16 m^2.

The resulting segmentations are depicted in Fig. 1. The small borders around the buildings are no artifacts but belong to the façades which are not represented in 2.5D. Now we performed geometric simplification

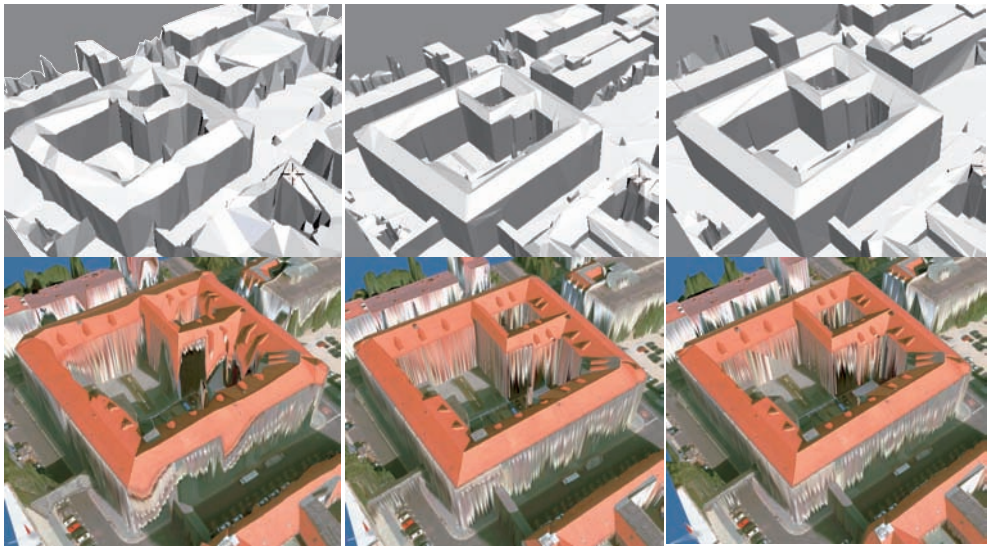

Fig. 2: Simplification results. Left column: unconstrained, 4 m threshold. Middle column: constrained, 4 m threshold. Right column: constrained, 32 m threshold. The upper row shows the shaded TIN whereas the lower row shows a rendering textured with full 12.5 cm resolution.

with exactly the same algorithm but either using a map of shape-IDs or not. Without shape-IDs the simplification of the 4.2 million vertices took 266 sec. and resulted in 2172 vertices, with the constraints it took 291 sec. and resulted in 7493 vertices. Fig. 2 shows the resulting models. The leftmost column of Fig. 2 shows the results at an error threshold of 4 m without additional constraints. Even the huge gable roofs look already scrambled, the small chimney in front turned into a strange looking peak and also on the flat roofs in the background we see some disadvantageous collapse artifacts. Texturing this model (lower row) reveals the spatial inaccuracy of the feature edges. That is definitely not what we would expect from an abstracted model, even though from a distance where a pixel projects to about 4 m it will be almost indistinguishable from the original.

In the middle column we see which difference the constraints make here. Geometric error is the same, but all collapses trying to demolish feature edges were inhibited. A lot of features, which are significantly smaller than 4 m and hence missing in the leftmost mesh are still present in the data, e. g., the glass roof in the courtyard or the chimney are reasonably represented. The textured rendering reveals the high positional accuracy of the feature edges which is due to the small ε threshold used during point classification. This effect becomes even clearer when we look at the rightmost column of Fig. 2. As the whole patch is only little more than 30 m high, distance threshold 32 m means collapse everything you can. So, every feature which is still there is there due to the shape-IDs it has.

7 Conclusions

We proposed a robust way to derive feature edges and corners from highly detailed digital surface models. Such constraints can be easily integrated into a Hausdorff distance simplification framework. Adding topological shape constraints and inhibiting collapse-vertex placement makes the resulting mesh strictly following the prescribed edge features, while still simplifying along these edges. Since the features are defined using a low-level shape detection, we are able to preserve the shape of very complex roof structures and buildings without having a specialized model of them. If a semantic annotation was added, the resulting geometry could be directly exported into a high-level format as CityGML.

Directions of future work will include a better support for curved features, such that there also the positional accuracy is independent of the global error threshold and also refining the definition of shapes, such that they approximately match existing concepts of semantic LODs. As the results in Figure 2 revealed that, the resulting triangulation is in some places even less complex than the pure geometric simplification, we will also try to improve the concept of shape such that vegetation and point-cloud artifacts do not lead to additional constraints.

Acknowledgements

We thank DLR – Institute of Robotics and Mechatronics for providing us the high-detailed Berlin dataset. This work was funded by the German Research Foundation DFG as a part of the bundle project ''Abstraction of Geographic Information within the Multi-Scale Acquisition, Administration, Analysis and Visualization''.

References

Agarwal, P.K. & Suri, S., 1994: Surface approximation and geometric partitions. – Proceedings of the SCCG (5th ACM-SIAM Sympos. On Discrete Algorithms): 24–33.

Cignoni, P., Rocchini, C., Montani, C. & Scopigno, R., 2003: External memory management and simplification of huge meshes. – IEEE Trans. on Vis. and Comput. Graph. **9** (4): 525–537.

Cignoni, P., Rocchini, C. & Scopigno, R., 1998: Metro: Measuring error on simplified surfaces. – Computer Graphics Forum **17** (2): 167–174.

Desbrun, M., Meyer, M., Schröder, P. & Barr, A. H., 1999: Implicit fairing of irregular meshes using diffusion and curvature flow. – In Computer Graphics (SIGGRAPH) **33:** 317–324.

El-Sana, J. & Yi-Jen, C., 2000: External memory view-dependent simplification and rendering. – Computer Graphics Forum **19** (3): 139–150.

Garland, M. & Heckbert, P., 1997: Surface simplification using quadric error metrics. – Computer Graphics (SIGGRAPH) **31:** 209–216.

Garland, M. & Shaffer, E., 2003: A multiphase approach to efficient surface simplification. – In IEEE Visualization: 117–124.

Guthe, M., Borodin, P. & Klein, R., 2005: Fast and accurate Hausdorff distance calculation between meshes. – Journal of WSCG **13** (2): 41–48.

Hildebrandt, K. & Polthier, K., 2004: Anisotropic Filtering of Non-Linear Surface Features. – Computer Graphics Forum **23** (3): 391–400.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J. & Stuetzle, W., 1993: Mesh Optimization. – SIGGRAPH: 19–26.

Hoppe, H., 1996: Progressive meshes. – Computer Graphics (SIGGRAPH) **30:** 99–108.

Hoppe, H., 1998: Smooth view-dependent level-of-detail control and its application to terrain rendering. – IEEE Visualization: 35–52.

Isenburg, M., Lindstrom, P., Gumhold, S. & Snoeyink, J., 2003: Large mesh simplification using processing sequences. – IEEE Visualization: 465–472.

Klein, R., Liebich, G. & Straßer, W., 1996: Mesh Reduction with Error Control. – IEEE Visualization: 311–318.

Kok-Lim, L. & Tiow Seng, T., 1997: Model simplification using vertex-clustering. – Symposium on Interactive 3D Graphics: 75–82.

Lindstrom, P., 2000: Out-of-core simplification of large polygonal models. – SIGGRAPH: 259–262.

Lindstrom, P. & Silva, C., 2001: A memory insensitive technique for large model simplification. – IEEE Visualization '01: 121–126.

Popovic, J. & Hoppe, H., 1997: Progressive simplicial complexes. – SIGGRAPH '97: 217–224.

Rossignac, J. & Borrel, P., 1993: Multi-resolution 3D approximations for rendering. – Modeling in Computer Graphics: Methods and Applications. Springer: 455–465.

Schnabel, R., Wahl, R. & Klein, R., 2007: Shape Detection in Point Clouds. – Computer Graphics Forum **26** (2): 214–226.

Schnabel, R., Wessel, R., Wahl, R. & Klein, R., 2008: Shape Recognition in 3D Point Clouds. – WSCG '2008 Full Papers: 65–72.

Schroeder, W., Zarge, J. & Lorensen, W., 1992: Decimation of triangle meshes. – Computer Graphics (SIGGRAPH '92) **26:** 65–70.

Taubin, G., 1995: A signal processing approach to fair surface design. – SIGGRAPH '95: 351–358.

Vorsatz, J., Rössl, C. & Kobbelt, L. & Seidel, H.-P., 2001: Feature Sensitive Remeshing. – Computer Graphics Forum **20** (3): 393–401.

Wu, J. & Kobbelt, L., 2003: A stream algorithm for the decimation of massive meshes. – In Graphics Interface: 185–192.

Address of the Authors:

Dipl.-Inform. Roland Wahl, Dipl.-Inform. Ruwen Schnabel, Prof. Dr. Reinhard Klein, Universität Bonn, Institut für Informatik II, AG Computergrafik, D-53117 Bonn, Tel.: +49-228-73-4146 (Wahl), 4122 (Schnabel), 4201 (Klein), Fax.: +49-228-73-4212, e-mail: wahl | schnabel | rk@cs.uni-bonn.de

Manuskript eingereicht: Dezember 2007
Angenommen: März 2008