6

Surface Reconstruction from Unorganized 3D Point Clouds

Patric Keller[1], Martin Hering-Bertram[2] and Hans Hagen[3]
[1]*University of Kaiserslautern*
[2]*University of Applied Sciences Bremen*
[3]*University of Kaiserslautern*
Germany

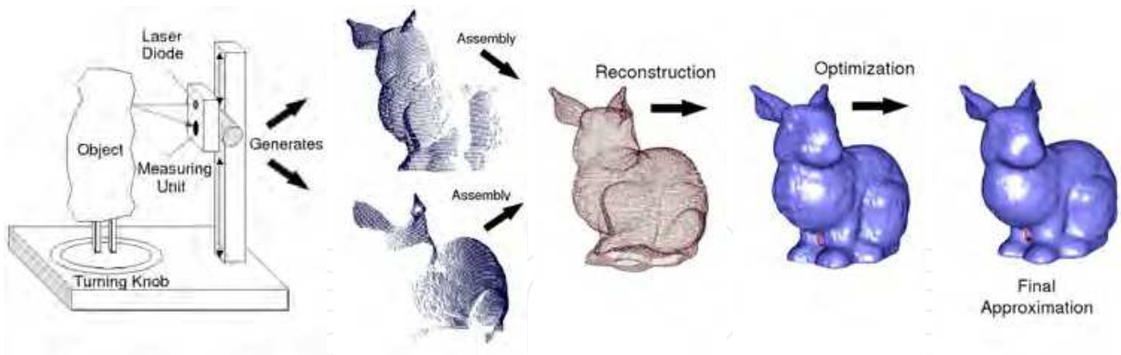
1. Introduction

Computer-based surface models are indispensable in several fields of science and engineering. For example, the design and manufacturing of vehicles, such as cars and aircrafts, would not be possible without sophisticated CAD and simulation tools predicting the behavior of the product. On the other hand, designers often do not like working on virtual models, though sophisticated tools, like immersive VR-environments are available. Hence, a designer may produce a physical prototype made from materials of his choice that can be easily assembled and shaped like clay models. Reverse engineering is the process of reconstructing digital representations from physical models. The overall reverse-engineering framework mainly is composed of four steps (see Figure 1): data acquisition, pre-processing, surface reconstruction, and post-processing;

The point cloud acquisition generally is performed by stationary scanning devices, like laser-range or computer-tomography scanners. In the case of a 3D laser scanner, the surface is sampled by one or more laser beams. The distance to the surface is typically measured by the time delay or by the reflection angle of the beam. After taking multiple scans from various sides or by rotating the object, the sampled points are combined into a single point cloud, from which the surface needs to be reconstructed.

Pre-processing of the data may be necessary, due to sampling errors, varying sampling density, and registration errors. Regions covered by multiple scans, for example, may result in noisy surfaces since tangential distances between nearest samples may be much smaller than the sampling error orthogonal to the surface. In this case, it is necessary to remove redundant points introduced by combining different points from multiple scans. In other regions, the density may be lower due to cavities and highly non-orthogonal scanning. If additional information, like a parametrization originating from each scan is available, interpolation can be used to fill these gaps.

In the present chapter, a powerful algorithm for multi-resolution surface extraction and -fairing, based on hybrid-meshes Guskov et al. (2002), from unorganized 3D point clouds is proposed (cf. Keller et al. (2005) and Keller et al. (2007)). The method uses an octree-based voxel hierarchy computed from the original points in an initial *hierarchical space partitioning* (HSP) process. At each octree level, the *hybrid mesh wrapping* (HMW) extracts the outer



(a) (b) (c) (d) (e)

Fig. 1. Principle steps of reverse engineering: (a) point cloud acquisition by 3D object scanner, mostly laser range scan devices; (b) slices scanned from different views, (c) combined point cloud, (d) reconstructed mesh, (e) result after post-processing.

boundary of the voxel complex, taking into account the shape on the next coarser level. The resulting meshes both are linked into a structure with subdivision connectivity, where local topological modifications guarantee the resulting meshes are two-manifold. Subsequently, a *vertex mapping* (VM) procedure is proposed to project the mesh vertices onto locally fitted tangent planes. A final post-processing step aims on improving the quality of the generated mesh. This is achieved by applying a mesh relaxation step based on the constricted repositioning of mesh vertices tangential to the approximated surface.

The remainder of the chapter is structured as followed. In Section 2 a short overview about related reconstruction techniques is provided. Section 3 discusses the individual steps of our approach in detail. Section 4 presents some results and discusses the advantages and disadvantages of the proposed method in terms of performance, quality and robustness. In addition section 5 presents some experimental results in the context of surface reconstruction from environmental point clouds. The conclusion is part of section 6.

2. Related work

A possible approach to obtain surfaces from unorganized point clouds is to fit surfaces to the input points Goshtasby & O'Neill (1993), such as fitting polynomial Lei et al. (1996) or algebraic surfaces Pratt (1987). To be able to fit surfaces to a set of unorganized points it is necessary to have information about the topology of the point cloud inherent surfaces or to have some form of parametrization in advance. For example, Eck and Hoppe Eck & Hoppe (1996) generate a first parametrization using their approach presented in Hoppe et al. (1992) and fit a network of B-Spline patches to the initial surface. This allows to reconstruct surfaces of arbitrary topology. A competing spline-based method is provided by Guo Guo (1997). Another form of surface reconstruction algorithm applying high-level model recognition is presented in Ramamoorthi & Arvo (1999).

Alexa et al. Alexa et al. (2001) introduced an approach for reconstructing point set surfaces from point clouds based on Levin's MLS projection operator. Further approaches following the idea of locally fitting polynomial surface patches to confined point neighborhoods are proposed in Alexa et al. (2003) Nealen (2004.) Fleishman et al. (2005) Dey & Sun (2005). In

Mederos et al. (2003) the authors introduce a MLS reconstruction scheme which takes into account local curvature approximations to enhance the quality of the generated surfaces. One of the main problems associated with the MLS-based techniques is that, in general, they have to adapt to meet the underlying topological conditions. Depending on the type of input data this can be rather challenging. Another drawback is, that the MLS-technique in general is not capable of constructing surfaces having sharp features. One attempt for solving this problem was proposed by Fleishman et al. Fleishman et al. (2005).

Whenever accuracy matters, adaptive methods are sought, capable of providing multiple levels of resolution, subdivision surfaces, for example, can be used together with wavelets Stollnitz et al. (1996) to represent highly detailed objects of arbitrary topology. In addition, such level-of-detail representations are well suited for further applications, like view-dependent rendering, multi-resolution editing, compression, and progressive transmission. In addition to adaptive polyhedral representations, subdivision surfaces provide smooth or piecewise smooth limit surfaces similar to spline surfaces. In Hoppe et al. (1994) Hoppe et al. introduce a method for fitting subdivision surfaces to a set of unorganized 3D points. Another class of reconstruction algorithms are computational geometry approaches. These algorithms usually extract the surface from previously computed Delaunay- or dual complexes. The reconstruction is based on mathematical guarantees but relies on clean data e.g., noisy, and non-regularly sampled points perturb the reconstruction process and may cause the algorithms to fail.

An early work concerning a Delaunay-based surface reconstruction scheme was provided by Boissonnat (1984). Following this idea, methods like the crust algorithm introduced by Amenta, Bern & Kamvysselis (1998) have been developed exploiting the structure of the Voronoi diagrams of the input data. Other works Funke & Ramos (2002) Amenta et al. (2001) Mederos et al. (2005) aimed at improving the original crust algorithm regarding efficiency and accuracy. The cocone algorithm Amenta et al. (2000) evolved from the crust algorithm provides further enhancements. Based on this work Dey et. al. Dey & Goswami. (2003) introduced the tight cocone. Other Delaunay/Voronoi-based reconstruction algorithms are presented by Kolluri et al. (2004), Dey & Goswami (2004).

One challenge concerns the separation of proximal sheets of a surface Amenta, Bernd & Kolluri (1998). When considering local surface components, it may be helpful to construct a surface parametrization, i.e. a one-to-one mapping from a proper domain onto the surface. Having a surface of arbitrary topology split into a set of graph surfaces, for example by recursive clustering Heckel et al. (1997), one can reduce the reconstruction problem to scattered-data approximation in the plane Bertram et al. (2003). A very powerful meshless parametrization method for reverse engineering is described by Floater and Reimers Floater & Reimers (2001).

A completely different approach is the construction of α-shapes described by Edelsbrunner and Mücke Edelsbrunner & Mücke (1994). Depending on a single radius α, their method collects all simplices (e.g. points, lines, triangles, tetrahedra, etc.) fitting into an α-sphere. The method efficiently provides a data structure valid for all choices of α, such that a user may interactively adapt α to obtain a proper outer boundary of a point cloud. Out-of-core methods like ball-pivoting Bernardini et al. (1999) employ the same principle, rolling a ball of sufficiently large radius around the point cloud filling in all visited triangles. Other methods

e.g., Azernikov et al. (2003) and Wang et al. (2005), exploit octree-based grid structures to guide surface reconstruction.

3. Reconstruction approach

3.1 Overview

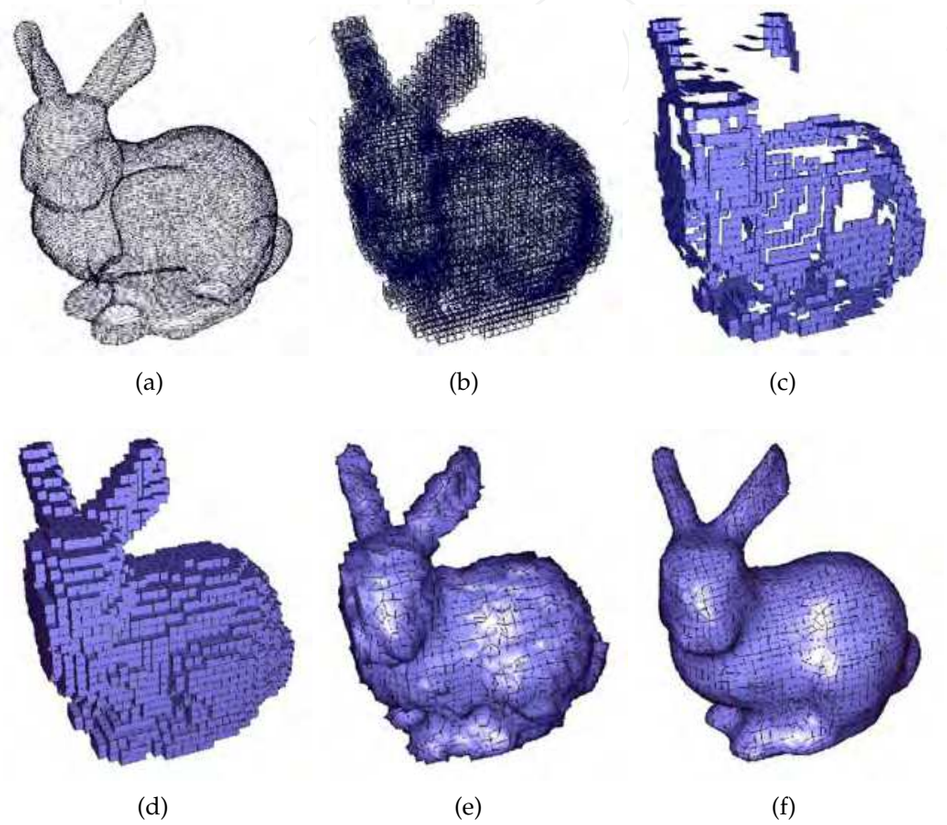


Fig. 2. (a)-(f) principle steps of the proposed multi-resolution surface reconstruction approach. (a) input point cloud of the Stanford bunny data set, consisting of 35,947 points, (b) voxel grid after 4 subdivision steps, (c) and (d) extracted voxel hull before and after application of the HMW operation, (e) surface mesh after vertex projection, (f) final surface after mesh relaxation.

This work aims at finding an efficient way to extract a connected quadrilateral two-manifold mesh out of a given 3D point cloud in a way that the underlying ”unknown“ surface is approximated as accurately as possible. The resulting adaptive reconstruction method is based upon the repetitive application of the following steps:

- Starting from an initial bounding voxel enclosing the original point cloud (see Figure 2(a)), the hierarchical space partitioning creates a voxel set by recursively subdividing each individual voxel into eight subvoxels. Empty subvoxel are not subject to subdivision and are deleted. Figure 2(b) presents an example of a generated voxel grid.
- The outer boundary of the generated voxel complex is extracted by the HMW operation. This exploits the voxel-subvoxel connectivity between the current and the next coarser

voxel grid. The resulting mesh is obtained by subdividing the coarser mesh (cf. Figure 2(c)) and adapting its topology at locations where voxels have been removed (see Figure 2(d)).

- The final vertex mapping locally constrains the mesh toward the point cloud (cf. Figure 2(e)). All vertices are projected onto local tangent planes defined by the points of the individual voxels. The resulting mesh is relaxed toward its final position by applying additional post-processing (see Figure 2(f)).

3.2 Hierarchical spatial partitioning

The HSP presumes an already existing voxel set V^j defining the voxel complex at level j. At the coarsest level this set V^0 consists of the bounding voxel enclosing the entire point cloud. To obtain V^{j+1} the following steps are performed:

1 Subdivide every voxel $v \in V^j$ into 8 subvoxels.

2 Assign the points of v to the corresponding subvoxel and delete empty subvoxels.

The task of managing and maintaining the originated non-empty set V^{j+1} is accomplished by the usage of an octree data structure. As required by the succeeding HM wrapping operation we need certain connectivity information facilitating the localization of proximate voxel neighborhoods. The algorithm applied uses an efficient octree-based navigation scheme related to the approach of Bhattacharya Bhattacharya (2001).

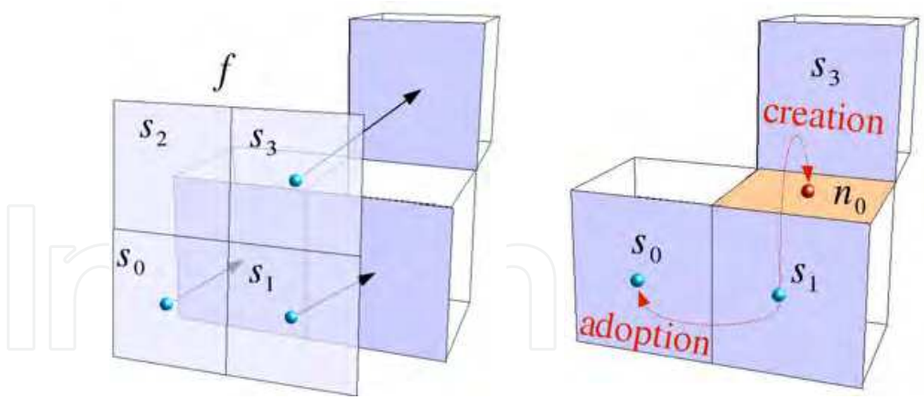
3.3 Hybrid mesh wrapping

The most difficult part of this work concerns the extraction of a two-manifold mesh from the generated voxel complex V^{j+1}. For the following let M^j denote the set of faces representing the existing mesh corresponding to the voxel complex V^j, where the term face abstracts the quadrilateral sidepart of a voxel. M^0 defines the face patches of V^0 forming the hull of the bounding voxel. Starting from the existing mesh M^j, we obtain the next finer mesh representation M^{j+1} by performing regular and irregular refinement operations. This includes the subdivision of M^j and the introduction of new faces at M^{j+1} inducing local changes in the mesh topology. These operations require M^j to meet the following conditions:

- Each face $f \in M^j$ is associated with exactly one voxel $v \in V^j$ (no two voxel can be associated with the same face). Thus the maximum number of faces associated with a voxel is restricted to six.
- The mesh represented by M^j is a two-manifold mesh.
- A face is linked to each of its proximate neighbor face. n is called a neighbor of (or adjacent to) the face f, if both share a common voxel edge. With limitation of one neighbor per edge the number of possible neighbor faces of f is four.

3.4 Regular mesh refinement

To accomplish the acquisition of the boundary hull associated with V^{j+1} the first step concerns the regular refinement of M^j. The refinement is achieved by subdividing each face $f \in M^j$ into four subfaces. To guarantee the resulting mesh fulfills the conditions outlined above, we assign the subfaces of f to voxel of V^{j+1}. For the following let $v \in V^j$ be the voxel



(a) (b)

Fig. 3. (a) regular refinement: Principles of the subface projection operation; (b) irregular mesh refinement: Principles of face creation and adoption;

assigned to $f \in M^j$ and s be a subface of f. The assignment procedure projects s onto the corresponding subvoxel of v as illustrated in Figure 3(a). f is restricted only to be projected onto the immediate subvoxel of v. Since we only address the extraction of the outer hull of V^j, additional rules have to be defined preventing faces of M^j to capture parts of the interior hull of the voxel complex e.g., in cases the surface is not closed. Thus, the subface s can not be assigned to a subvoxel of v if this subvoxel already is associated with an face on the opposite. Subfaces that can not be assigned because no corresponding subvoxel exist they could be projected onto, are removed. The resulting refinement operator $\mathcal{R}$ defines the set $N^{j+1} := \mathcal{R}(M^j)$, with $N^0 = M^0$ representing the collection of the created and projected subfaces (cf. Figure 2(c)). So far N^{j+1}, consisting of unconnected faces, represents the base for the subsequent *irregular mesh refinement*, in which the final mesh connectivity is recovered.

3.5 Irregular mesh refinement

The *irregular mesh refinement* recovers the mesh connectivity i.g., it reconnects the faces of N^{j+1} and closes resulting breaks in the mesh structure induced by the regular mesh refinement. This procedure is based on the propagation of faces or the adoption of existing proximate neighbor faces (see Figure 3(b)). More precisely: Assume f to be a face of N^{j+1}, the irregular refinement detects existing neighbor faces in N^{j+1} sharing a common voxel edge or creates new faces in case no neighbor is found. Considering the configuration of the principle voxel neighborhoods there are three possible cases a neighbor face of f can be adapted/created. Figure 4 depicts these cases. For the following considerations let $n \in N^{j+1}$ specifying the "missing" neighbor face of f, $v \in V^{j+1}$ the voxel associated with f, and $w \in V^{j+1}$ the voxel associated with n. Exploiting the voxel neighborhood relations, the propagation/adoption of n is performed according the summarized rules below:

1. In case that v and w are identical the creation/adoption of n is admitted if f does not already share an edge with a face on the opposite of v (see Figure 4(a)).
2. The faces n and f share a common edge, the corresponding voxel w and v adjoin a common face (see Figure 4(b)). The creation/adoption of n is allowed if no other face is associated with w, vis-a-vis from n (additional *cavity rule*).

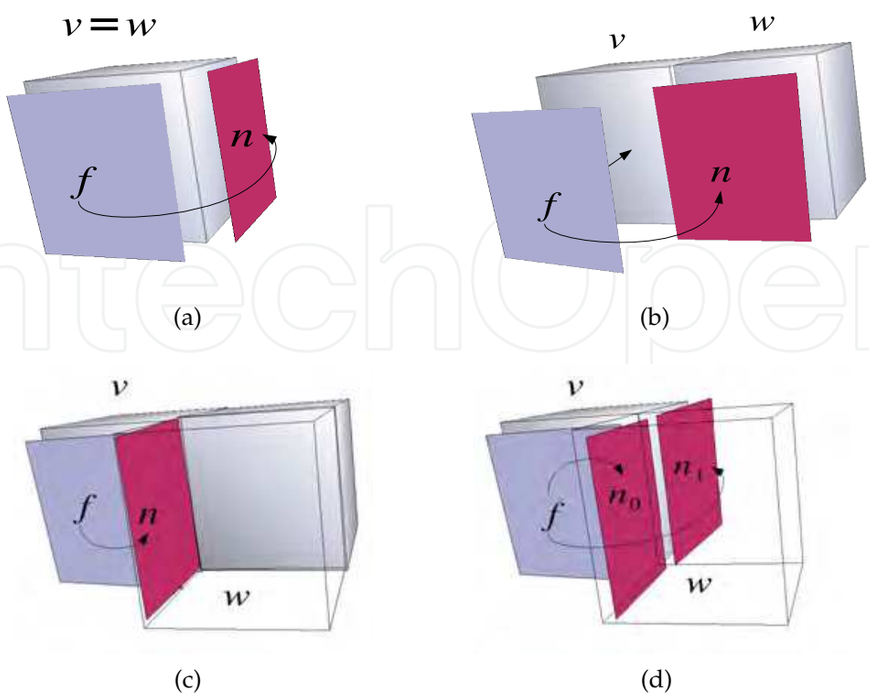


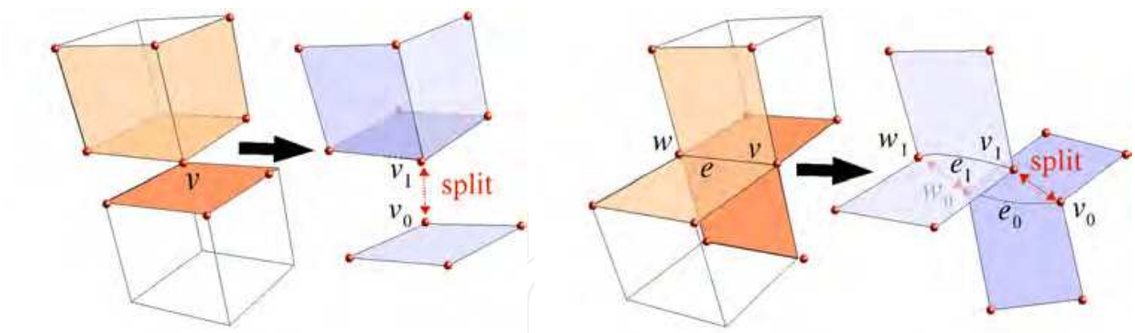
Fig. 4. Graphical interpretation of the face-propagation rules concerning the possible cases.

3. The voxel v and w adjoin a common edge. In this case we have to differentiate between two cases: a) an additional voxel adjoins v (see Figure 4(c)). In this case the creation/adoption of n is permitted. b) only v and w adjoin a common edge (see Figure 4(d)). This case requires to detect the underlying topological conditions. If the underlying surface passes v and w, we adopt/create n_0, otherwise, if the points within v and w represent a break of the underlying surface we adopt/create n_1. The right case is filtered out by comparing the principal orientation of the points within v and w.

The irregular refinement procedure is performed as followed: Given two initial sets $A^0 = N^{j+1}$ and $B^0 = \emptyset$, once a new neighbor face n is created/adopted, it is added to $A^{i+1} = A^i \cup \{n\}$. Simultaneously, every $n \in A^i$ which is fully connected to all of its existing neighbor faces is removed $A^{i+1} = A^i \setminus \{n\}$ and attached to $B^{j+1} = B^j \cup \{n\}$. This procedure is repeated until $A^i = \emptyset$ or $A^{j+1} = A^j$.

Applying the irregular mesh refinement operator $\mathcal{I}$ on N^{j+1} results in $M^{j+1} = \mathcal{I}(N^{j+1})$, where $M^{j+1} = A^i \cup B^j$ represents the final mesh at level $j+1$. Figure 3(b) illustrates the propagation procedure, where one neighbor is created and another adopted.

To force M^{j+1} to maintain the two-manifold condition each case at which the propagation of a face leads to a non-manifold mesh structure e.g., more than two faces share an edge, is identified and the mesh connectivity is resolved by applying vertex- and edge-splits. Figure 5(a) and Figure 5(b) illustrate the principles according to these the non-manifold mesh structures are resolved. In the depicted cases the connectivity of the vertices v and v_1, v_2 cause the mesh to be non-manifold. We avoid this by simply splitting the corresponding vertices and edges (see right part of Figure 5(a) and Figure 5(b)).



(a) (b)

Fig. 5. (a) example in which a non-manifold mesh structure is resolved by a vertex-split, (b) resolving non-manifold mesh structure by first applying an edge-split followed by two vertex-splits.

3.6 Vertex-mapping

The next step concerns moving the mesh toward the "unknown" surface by projecting the mesh vertices onto the local tangent planes defined by the set of proximate sample points. This is accomplished for each vertex v of the mesh M by first identifying the voxel set W directly adjoining v and collecting the enclosed points P_v. Next, we fit a plane to the points P_v by computing the centroid $\vec{c}$ and the plane normal $\vec{n}$ obtained from the covariance matrix C of P_v. In order to improve the accuracy of the fitting the points of P_v can be filtered according their distance to v yielding $P'_v = \{p \in P_v | \, 2\|p - v\| < l\}$, with l representing the edge length of the voxel complex W. The normal $\vec{n}$ is defined by the eigenvector associated with the smallest eigenvalue of C. Together with the former position of the vertex $\vec{v}$ we are able to compute the new coordinates of $\vec{v}_n$ by

$$\vec{v_n} = \vec{v} - ((\vec{v} - \vec{c}) \cdot \vec{n})\vec{n} \, . \tag{1}$$

To be able to perform this projection the number of points of P_v has to be $|P_v| \geq 3$. Otherwise, points from adjacent voxels need to be added from surrounding voxels. By extending W to

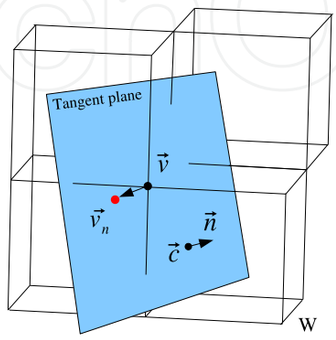
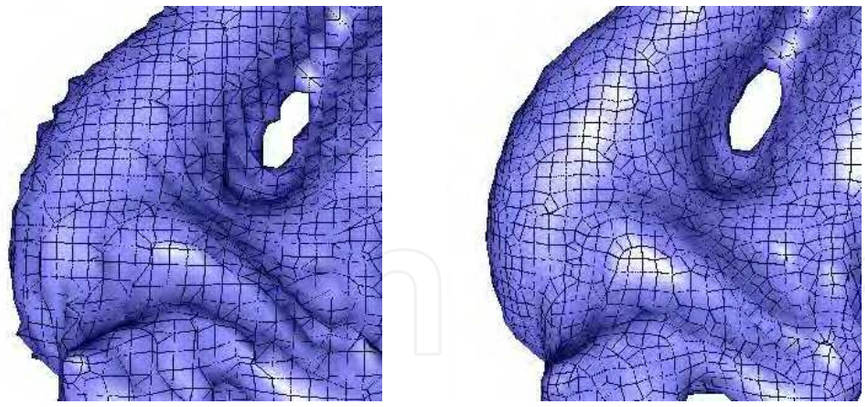


Fig. 6. Vertex projected onto the tangent plane defined by the points P_v of the adjacent voxel set W.



(a) (b)

Fig. 7. (a) part of the reconstructed surface mesh from the dragon data set before smoothing was applied, (b) corresponding mesh after few relaxation steps.

$W' = W \cup \{v \in V \setminus W | v$ is directly adjacent [1] to at least one $w \in W\}$ the number of points in P_v can be increased.

3.6.1 Post-processing

To improve the quality of the generated mesh we perform an additional mesh optimization step. Based on the principles of Laplacian smoothing, the vertices of the mesh are repositioned by first computing the centroid of the directly connected neighbor vertices. In a subsequent step these centroids are again projected onto the tangent planes of the corresponding point sets according to equation (1). Generally, mesh-optimization is a repetitive process, applied several times to obtain the most possible gain in surface quality, see Figure 7(a) and Figure 7(b).

4. Results

4.1 Performance

To find the overall time complexity we have to look at every step of the algorithm separately. We begin discussing the spatial decomposition analysis: In order to distribute the points contained by a voxel set V^{k-1} to their respective subvoxel we need to determine the subvoxel affiliation of every point. This leads to an computational complexity of $O(|P|)$ in every refinement step. Considering the HM wrapping we have to differentiate between the regular $\mathcal{R}$ and the irregular $\mathcal{I}$ operation but keep in mind that both are interdependent. Due to the fact that $\mathcal{R}$ basically depends linearly on the number of faces of $|M^{k-1}|$ and hence on $|V^{k-1}|$ we obtain a complexity of $O(|V^{k-1}|)$. Since it is difficult to estimate the number of attempts needed to find M^k we cannot reveal accurate statements concerning the computation time for $\mathcal{I}$. Based on empirical observations, an average time complexity of $O(k \cdot |V^k|)$ holds.

Assuming that $|V^k| \ll |P|$ with constant k (say $0 < k \leq 10$) the combined results of the particular sub-processes leads to an overall complexity of $O(|P|)$, concerning one single

[1] Two distinct voxels are directly adjacent if they share a common vertex, edge or face.

refinement step. The complexity to completely generate a mesh representing the unknown surface at refinement level k averages $O(k \cdot |P|)$.

The following time tables confirm the above discussed propositions. Table 1 shows the measured computation times for several data sets (*Stanford Scan Repository* (2009)) performed on an Intel Pentium 4 system with 1.6 GHz and 256 MB main memory. Table 2 presents more detailed results for the individual reconstruction steps for the Stanford dragon point data set.

Object	points	faces	ref. level	time[sec] reconstr.	time[sec] opt.
Rabbit	8171	25234	6	1.058	0.244
Dragon	437645	72955	7	5.345	0.714
Buddha	543652	56292	7	5.170	0.573

Table 1. Computation time table regarding several input models whereas the VM and the mesh-optimization (4 passes) are performed after the last reconstruction step.

Ref.-Level	Spatial-Partitioning [sec]	HM-Extraction $\mathcal{R}$ [sec]	HM-Extraction $\mathcal{I}$ [sec]	Vertex-Mapping [sec]	Opt. [sec]	Complete [sec]
1	0.240	< 0.001	< 0.001	0.019	< 0.001	0.262
2	0.330	< 0.001	< 0.001	0.068	< 0.001	0.401
3	0.388	< 0.001	0.002	0.241	0.002	0.634
4	0.401	< 0.001	0.014	0.676	0.008	1.100
5	0.278	0.002	0.056	0.787	0.050	1.173
6	0.362	0.008	0.170	0.928	0.180	1.648
7	0.662	0.039	0.717	1.683	0.725	3.826

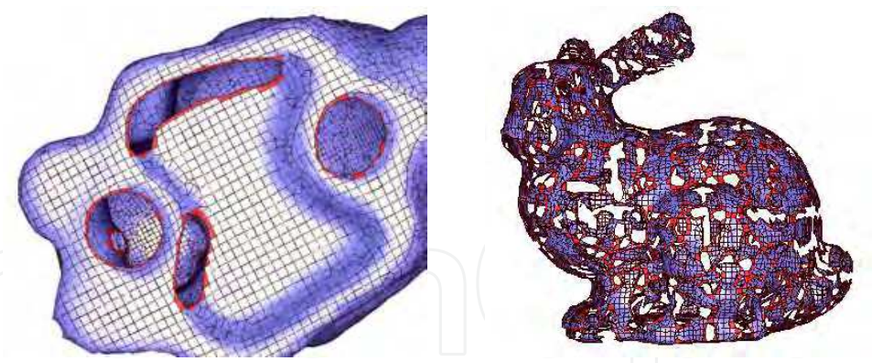
Table 2. Time values for each reconstruction step of the Stanford dragon.

4.2 Robustness and quality

As shown by the examples our reconstruction method delivers meshes of good quality, as long as the resolution of the voxel complex does not exceed a point density induced threshold. Topological features, such as holes and cavities were always detected correctly after a proper number of refinements, see figure 8(a). The HM wrapping rules prevent the underlying object from caving by simultaneously covering the real surface completely. Table 3 shows the measured L_2-errors obtained by processing point clouds of different models for several refinement levels.

ref. level	1	2	3	4	5	6	7
L_2-error Bunny	25,86	9.47	3.60	1.13	0.41	0.14	-
L_2-error Buddha	-	51.92	18.80	7.61	3.47	1.58	0.66

Table 3. L_2-error values of the Bunny and Buddha reconstruction for each ref. level (measured in lengths of diagonal of the Bounding Box).



(a) (b)

Fig. 8. (a) correctly detected holes in the bunny model, (b) fragmentation of the mesh due to lacking density of point cloud.

In case the voxel refinement level exceeds a specific bound imposed by sampling density, the resulting mesh may be incomplete in some areas, due to empty voxels, see figure 8(b). If a very fine mesh is desired, one can fix the topology at a coarser level and apply subdivision and vertex mapping in consecutive steps.

5. Experimental reconstruction of environmental point data

The exploration of point data sets, like environmental LiDaR data, is a challenging problem of current interest. In contrast to the point clouds obtained from stationary scanning devices, data complexity is increased by e.g., noise, occlusion, alternating sample density and overlapping samples. Despite of the high scanning resolution, additional undersampling may occur at small and fractured artifacts like fences and leaves of trees. These are only some problems immanent to the reconstruction of surfaces from unorganized environmental point clouds.

We have applied our method to environmental point clouds in order to analyse the effects of the aforementioned influence factors on our proposed reconstruction approach. In this context we have performed some experiments on environmental input data generated by tripod mounted LiDaR scanners. Figure 10(a) shows a point cloud of a water tower located at the UC Davis, CA, USA. It consists of about 4.3 million points and features complex environmental structures like buildings and trees. Figure 10(b) shows the corresponding reconstruction after 10 steps. The generated mesh exhibiting about 300 thousand faces took 23 seconds on an Intel Core2 Duo system with 4GB RAM to finish. Another example is presented in figure 11. It shows the final surface representation (215 thousand faces) of a slope with parts of a building and some vegetation. The underlying point cloud has 3.8 million points. The reconstruction finished at level 9 after 24 second. The reconstructed mesh features holes and cracks at those areas at which the resolution lacks in density.

Environmental point clouds by nature are large (consisting of up to several million points) and feature high complexity. The experiments showed that our reconstruction approach is capable of producing reliable representations even in cases in which the point clouds are not optimally conditioned. However, the reconstruction lacks at some regions in which the point density does not exhibit the required/satisfied resolution. Despite of this fact the introduced method

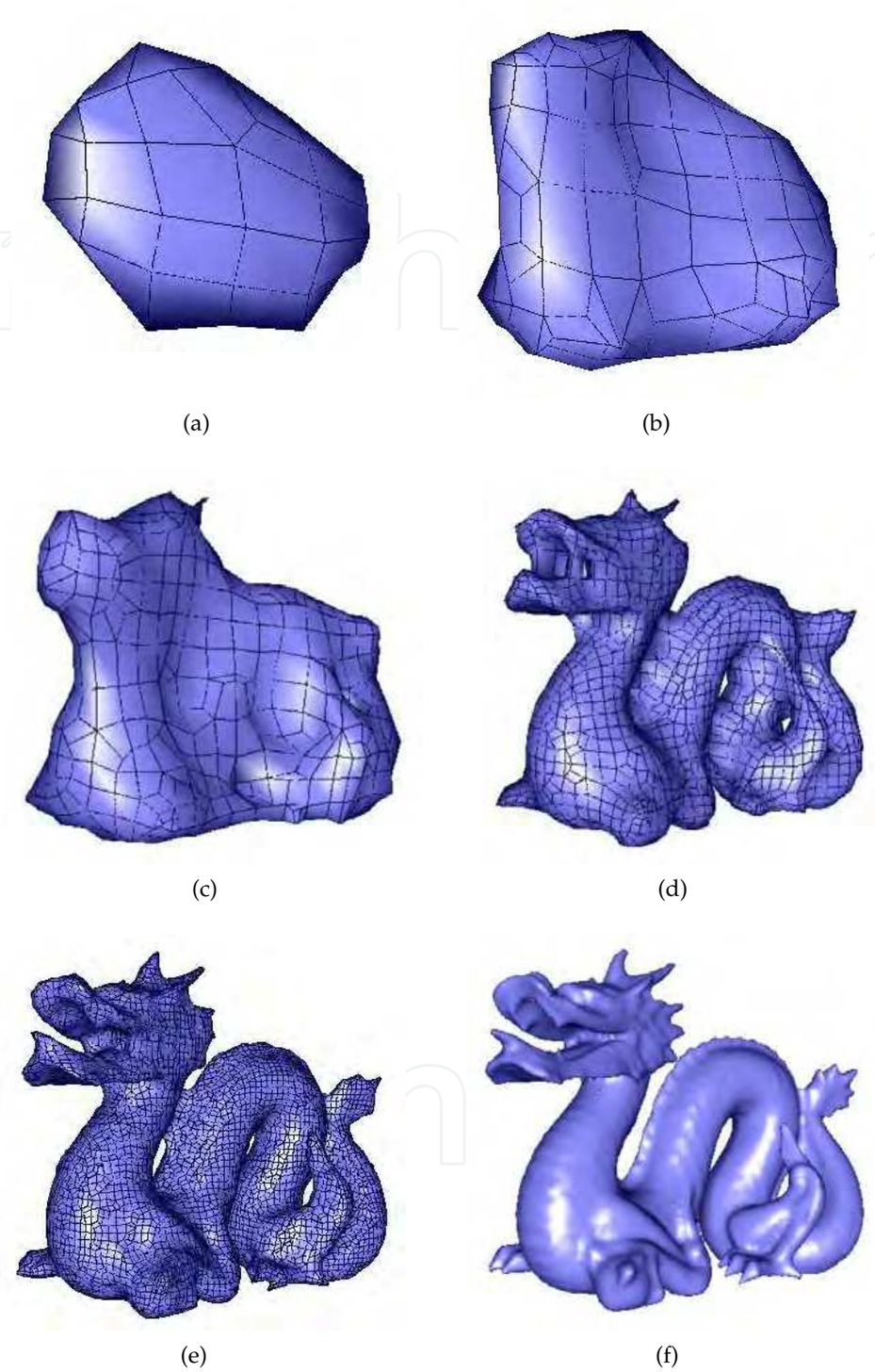


Fig. 9. (a)-(f) different reconstruction-levels, from level one to six, of the Stanford dragon point data set (*Stanford Scan Repository* (2009)) consisting of 437,645 points.

(a) (b)

Fig. 10. Water tower scan at the campus of UC Davis. (a) Raw point cloud having 4.3 mio. points, (b) reconstruction having 300k faces (10 subdivision steps) (total reconstruction time of about 23 seconds performed with an Intel Core2 Duo).

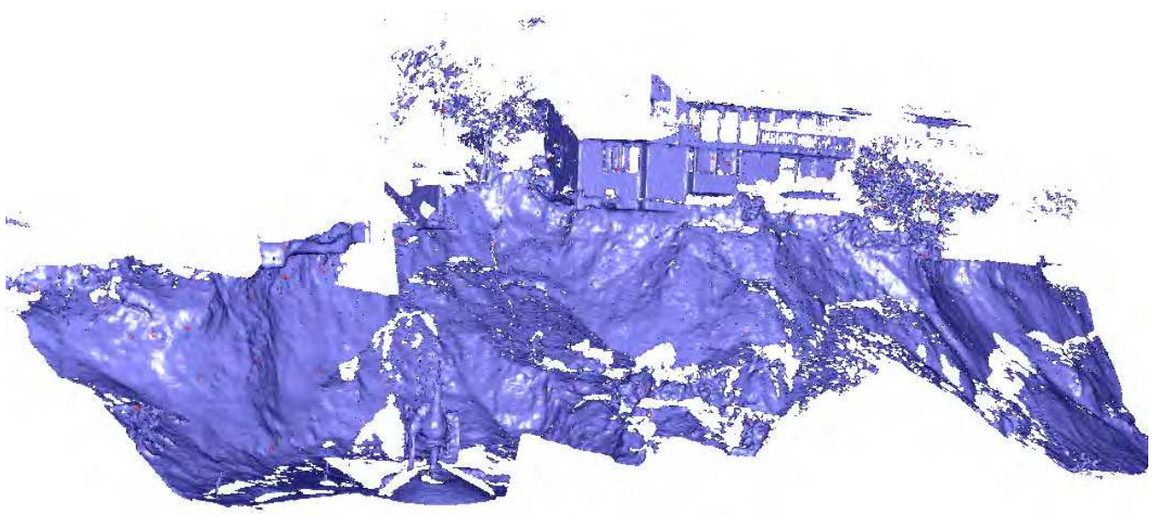


Fig. 11. Reconstruction of environmental point cloud (3.8 mio. points, 215k faces, 9 steps, 24 seconds).

is well suited for providing a fast preview of complex environmental scenes and serves as basis for providing initial reconstructions with respect to further mesh processing.

6. Conclusion

We provided a novel multi-resolution approach to surface reconstruction from point clouds. Our method automatically adapts to the underlying surface topology and provides a fully-connected hybrid-mesh representation. In the context of reverse engineering it is able to provide accurate reconstructions assumed that the input data shows a sufficient point density. However, in case the point distribution is not continuous the generated reconstruction may

exhibit cracks and holes. One possible direction for future work would be the improvement of the stability of the approach regarding such unwanted effects. This could be achieved e.g., by adapting the reconstruction in order to become more sensitive to irregular point distributions.

7. Acknowledgements

This work was supported by the German Research Foundation (DFG) through the International Research Training Group (IRTG) 1131, and the Computer Graphics Research Group at the University of Kaiserslautern. It was also supported in parts by the W.M. Keck Foundation that provided support for the UC Davis Center for Active Visualization in the Earth Sciences (KeckCAVES). We also thank the members of the Computer Graphics Research Group at the Institute for Data Analysis and Visualization (IDAV) at UC Davis, the UC Davis Center for Active Visualization in the Earth Sciences (KeckCAVES) as well as the Department of Geology at UC Davis for their support and for providing us the data sets.

8. References

Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D. & Silva, C. (2003). Computing and rendering point set surfaces, *IEEE Transactions on Visualization and Computer Graphics* 9(1): 3–âĂŞ15.

Alexa, M., Behr, J., Cohen-Or, D., Fleishman, S., Levin, D. & Silva, C. T. (2001). Point set surfaces, *Conference on Visualization*, IEEE Computer Society, pp. 21–28.

Amenta, N., Bern, M. & Kamvysselis, M. (1998). A new vornonoi-based surface reconstruction algorithm, *ACM Siggraph '98*, pp. 415–421.

Amenta, N., Bernd, M. & Kolluri, D. E. (1998). The crust and the beta-skeleton: combinatorial curve reconstruction, *Graphical Models and Image Processing, 60*, pp. 125–135.

Amenta, N., Choi, S., Dey, T. K. & Leekha, N. (2000). A simple algorithm for homeomorphic surface reconstruction, *SCG '00: Proceedings of the sixteenth annual symposium on Computational geometry*, pp. 213–222.

Amenta, N., Choi, S. & Kolluri, R. K. (2001). The power crust, *6th ACM symposium on Solid Modeling and Applications*, ACM Press, pp. 249–266.

Azernikov, S., Miropolsky, A. & Fischer, A. (2003). Surface reconstruction of freeform objects based on multiresolution volumetric method, *ACM Solid Modeling and Applications*, pp. 115–126.

Bernardini, F., Mittleman, J., Rushmeier, H., Silva, C. & Taubin, G. (1999). The ball-pivoting algorith for surface reconstruction, *IEEE Transactions on Visualization and Computer Graphics (TVCG)* 5(4): 349–359.

Bertram, M., Tricoche, X. & Hagen, H. (2003). Adaptive smooth scattered-data approximation for large-scale terrain visualization, *Joint EUROGRAPHICS - IEEE TCVG Symposium on Visualization*, pp. 177–184.

Bhattacharya, P. (2001). Efficient neighbor finding algorithms in quadtree and octree.

Boissonnat, J.-D. (1984). Geometric structures for three-dimensional shape representation, *ACM Trans. Graph.* 3(4): 266–286.

Dey, T. K. & Goswami., S. (2003). Tight cocone: A water tight surface reconstructor, *Proc. 8th ACM Sympos. Solid Modeling Appl.*, pp. 127–134.

Dey, T. K. & Goswami, S. (2004). Provable surface reconstruction from noisy samples, *20th Annual Symposium on Computational Geometry*, ACM Press, pp. 330–339.

Dey, T. K. & Sun, J. (2005). An adaptive mls surface for reconstruction with guarantees, *3rd Eurographics Symposium on Geometry Processing*, pp. 43–52.

Eck, M. & Hoppe, H. (1996). Automatic reconstruction of b-spline surfaces of arbitrary topological type, *ACM Siggraph*, pp. 325–334.

Edelsbrunner, H. & Mücke, E. P. (1994). Three-dimensional alpha shapes, *ACM Transactions on Graphics* 13(1): 43–72.

Fleishman, S., Cohen-Or, D. & Silva, C. (2005). Robust moving least-squares fitting with sharp features, *ACM Siggraph*, pp. 544–552.

Floater, M. & Reimers, M. (2001). Meshless parameterization and surface reconstruction, 18(2): 77–92.

Funke, S. & Ramos, E. A. (2002). Smooth-surface reconstruction in near-linear time, *SODA '02: Proceedings of the thirteenth annual ACM-SIAM symposium on Discrete algorithms*, pp. 781–790.

Goshtasby, A. & O'Neill, W. D. (1993). Surface fitting to scattered data by a sum of gaussians, *Comput. Aided Geom. Des.* 10(2): 143–156.

Guo, B. (1997). Surface reconstruction: from points to splines, 29(4): 269–277.

Guskov, I., Khodakovsky, A., Schroeder, P. & Sweldens, W. (2002). Hybrid meshes: Multiresolution using regular and irregular refinement, *ACM Symposium on Geometric Modeling (SoCG)*, pp. 264–272.

Heckel, B., Uva, A. & Hamann, B. (1997). Cluster-based generation of hierarchical surface models, *Scientific Visualization, Dagstuhl*, pp. 113–222.

Hoppe, H., DeRose, T., Duchamp, T., Halstead, M., Jin, H., McDonald, J., Schweitzer, J. & Stuetzle, W. (1994). Piecewise smooth surface reconstruction, *ACM SIGGRAPH 1994 Conference Proceedings*, pp. 295–302.

Hoppe, H., DeRose, T., Duchamp, T., McDonald, J. & Stuetzle, W. (1992). Surface reconstruction from unorganized points, *ACM SIGGRAPH '92 Conference Proceedings*, pp. 71–78.

Keller, P., Bertram, M. & Hagen, H. (2005). Multiresolution surface reconstruction from scattered data based on hybrid meshes, *IASTED VIIP*, pp. 616–621.

Keller, P., Bertram, M. & Hagen, H. (2007). Reverse engineering with subdivision surfaces, *Computing 2007*, pp. 127–134.

Kolluri, R., Shewchuk, J. R. & O'Brien, J. F. (2004). Spectral surface reconstruction from noisy point clouds, *SGP '04: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pp. 11–21.

Lei, Z., Blane, M. M. & Cooper, D. B. (1996). 3l fitting of higher degree implicit polynomials, 2(4): 148–153.

Mederos, B., Velho, L. & de Figueiredo, L. H. (2003). Moving least squares multiresolution surface approximation, *Sibgraphi*.

Mederos, L. V. B., Amenta, N., Velho, L. & de Figueiredo, L. (2005). Surface reconstruction from noisy point clouds, *Eurographics Symposium on Geometry Processing*.

Nealen, A. (2004.). An as-short-as-possible introduction to the least squares, weighted least squares and moving least squares methods for scattered data approximation and interpolation, *Technical Report, TU Darmstadt*.

Pratt, V. (1987). Direct least-squares fitting of algebraic surfaces, *SIGGRAPH Comput. Graph.* 21(4): 145–152.

Ramamoorthi, R. & Arvo, J. (1999). Creating generative models from range images, *SIGGRAPH '99: Proceedings of the 26th annual conference on Computer graphics and interactive techniques*, pp. 195–204.

Stanford Scan Repository (2009).
URL: *http://graphics.stanford.edu/data/3Dscanrep/*

Stollnitz, E., DeRose, T. & Salesin, D. (1996). *Wavelets for Computer Graphics–Theory and Applications*, Morgan Kaufmann.

Wang, J., Oliveira, M. & Kaufman, A. (2005). Reconstructing manifold and non-manifold surfaces from point clouds, *IEEE Visualization*, pp. 415–422.

Reverse Engineering - Recent Advances and Applications
Edited by Dr. A.C. Telea

ISBN 978-953-51-0158-1
Hard cover, 276 pages
Publisher InTech
Published online 07, March, 2012
Published in print edition March, 2012

Reverse engineering encompasses a wide spectrum of activities aimed at extracting information on the function, structure, and behavior of man-made or natural artifacts. Increases in data sources, processing power, and improved data mining and processing algorithms have opened new fields of application for reverse engineering. In this book, we present twelve applications of reverse engineering in the software engineering, shape engineering, and medical and life sciences application domains. The book can serve as a guideline to practitioners in the above fields to the state-of-the-art in reverse engineering techniques, tools, and use-cases, as well as an overview of open challenges for reverse engineering researchers.

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen