Chapter 8

Feature Adaptive Representation

While point set surfaces are used widely in point processing pipelines, their definition is inherently based on the assumption of uniform sampling. We explain, why simple approaches to account for feature adaptive sampling (e.g., [83, 6, 62]) are not sufficient to solve the problem and present a fundamentally new approach: Instead of locally estimating the feature size and then using this feature size for weighting the samples, each sample defines its influence on the space.

This weighting scheme allows representing featureless areas of the surface with fewer samples, resulting in a compact representation. We focus on using ellipsoidal weight functions for the points. Such representations can be derived directly from clean sample sets (Section 8.3.1), or, using variational shape approximations techniques, directly from dense, noisy samples (Section 8.3.2). It is also possible to derive the ellipsoids from the curvature of the surface. We present an approach where samples are generated by using repelling particles (or floaters) that are restricted to the surface (Section 8.3.3).

Note, however, that our goal is not shape compression, i.e. storing the surface with as few points/bits as possible given a certain error threshold [28, 82, 39, 100, 104]. We rather aim for robustness and flexibility with regard to variations in the input.

Section 8.2 details the algorithm to compute the surface from a set of points with ellipsoidal weight functions. The construction of spatial data structures for ellipsoidal weighting functions is more complex than for spherical supporters with equal radii. Details are discussed in Section 5.1. Sometimes the ellipsoidal weights are protruding into free space and could result in artifacts. This can be easily corrected by an adaptive closeness criteria (Section 8.2.1). Several examples show that ellipsoidal weighting allows shape representation that adapts to principal curvature directions and variations in the surface.

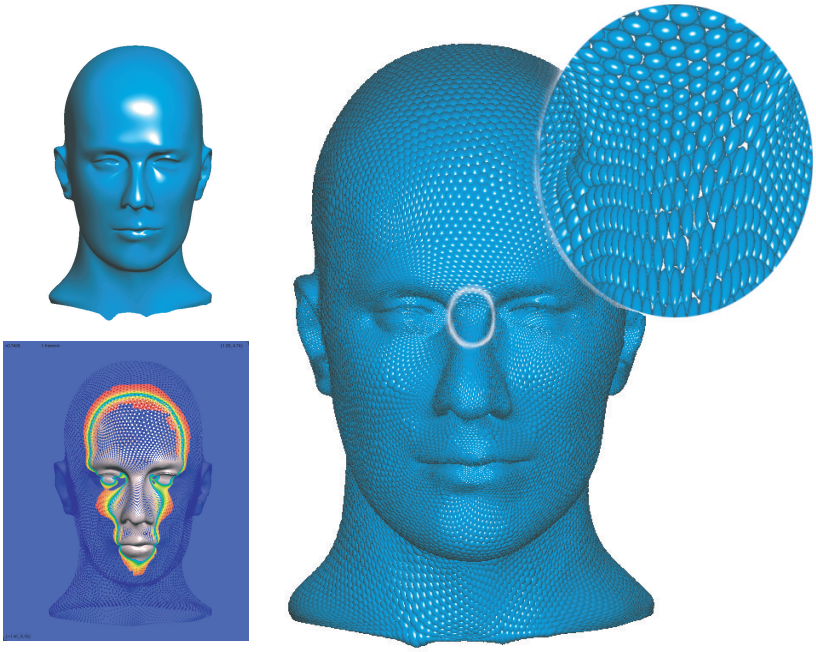

Figure 8.1: The smooth surface of the Mannequin model represented by an anisotropic sampling with ellipsoidal weight functions. The ellipsoids have been derived from a given set of point samples, which are used as the centers. The support of the weight functions overlaps and the ellipsoids have been scaled by half for this illustration.

8.1 Weighting

In many reconstruction approaches, all weight functions $w_i(\mathbf{x})$ behave in the same way, i.e. one could drop the index i and simply define

$$w_i(\mathbf{x}) = w(\mathbf{x}, \mathbf{p}_i) = \theta\left(\frac{\|\mathbf{x} - \mathbf{p}_i\|}{h}\right) \tag{8.1}$$

where $\theta(\cdot)$ is a smooth, monotonically decreasing function. In most cases we use piecewise polynomial approximations to a Gaussian-shaped function with limited support (see Section 2.4). The parameter h is a model-specific constant describing the average distance among point samples. The obvious drawback of this approach is that the point density has to be roughly constant all over the surface, or, at least, cannot vary arbitrarily. On the other hand, one would like to have more point samples in regions of high surface variation and less samples where the surface is flat. Pauly et al. [82, 83] suggest to adapt h to the local density of the points, i.e. to define a function $h : \mathcal{B} \to \mathbb{R}$ and

then

$$w_i(\mathbf{x}) = w(\mathbf{x}, \mathbf{p}_i) = \theta\left(\frac{\|\mathbf{x} - \mathbf{p}_i\|}{h(\mathbf{x})}\right). \tag{8.2}$$

However, it is known in statistics that these *baloon estimators*, especially those based on k-nearest neighbors, fail to improve the order of mean squared error (MSE) relative to h [88]. This means the reconstructed surface will suffer from irregular samplings, and varying h over space cannot fundamentally compensate for the problem.

For this reason, we rather suggest to use weight functions defined per point

$$w_i(\mathbf{x}) = \theta\left(d_i(x, \mathbf{p}_i)\right). \tag{8.3}$$

Not only does this type of definition have the potential to significantly improve the MSE, it also adds a lot of flexibility to the modeling framework: the shape of iso-contours of a weight function can be controlled per point.

In the following we discuss potential shapes for the weight functions. We begin with the case of planar curves and extend the ideas to surfaces in space.

8.1.1 Plane curves

Consider a plane curve with constant curvature. This curve should be sampled at regular intervals proportional to the curvature, and it is sufficient to weight each sample spherically:

$$w_i(\mathbf{x}) = \theta\left(\frac{\|\mathbf{x} - \mathbf{p}_i\|}{h}\right). \tag{8.4}$$

There is no reason to weight the influence of points differently along tangent and normal direction of the curve, as long as the curve is far away from other curve components. Of course, the supports of the weight functions have to be large enough to overlap. If they are very large, the representation is inefficient. Due to numerical problems very small or very large weights can also result in severe artifacts (see [4] for details).

For varying curvature, the radius of spheres and the spacing of samples should adapt to the curvature

$$w_i(\mathbf{x}) = \theta\left(\frac{\|\mathbf{x} - \mathbf{p}_i\|}{h_i}\right), \tag{8.5}$$

which can be modeled by adjusting h_i according to the curvature in $\mathbf{p}_i$. But since curvature varies only along one direction we have found no reason to use more complicated weights than radial ones. Quite on the contrary, the potential complications resulting from anisotropic weights (see Section 5.2) seem to outweigh their potential benefits.

8.1.2 Surfaces

The surface case differs significantly from plane curves in that the curvature can be different along the two dimensions of a surface. Let $(k_{\min}, k_{\max})$ be the minimum and

maximum principal curvatures. As mentioned for the curve case, the weighting should be relative to the curvature, and neither too small nor too large.

A conservative spherical support is typically bounded from above by $k_{\max}$. If the principal curvatures differed that would result in an inefficient representation along the direction of smaller curvature. A cylinder is an extreme case, where $k_{min} = 0$, $k_{max} = \frac{1}{r}$. Along the curved direction a sufficient number of samples is required, in order to approximate the circular shape. The small spacing of these samples yields a relatively small support, resulting in only a narrow ring of the cylinder being represented. Although there is no curvature in the other dimension, several samples are needed to represent a long cylinder.

To allow weighting the two tangental directions differently we propose using ellipsoidal weights H_i instead of radii h_i in each sample. Weighting is then performed by transforming the vector $\mathbf{x} - \mathbf{p}_i$ with H_i, i.e.

$$w_i(\mathbf{x}) = \theta(\|H_i^{-1}(\mathbf{x} - \mathbf{p}_i)\|) \tag{8.6}$$

The *weighting ellipsoid* H_i should be oriented so that one of its axes points into normal direction, and the other two align with the principal curvature directions. The length of the tangental half-axes should reflect $k_{\min}$ and $k_{\max}$. In Figure 8.8, top row, we show how to represent a cylinder using a ring of samples with ellipsoidal weight functions. The length of the cylinder could be arbitrary [1]

Other support functions would be also possible. However, we have found that except for extreme cases, ellipsoidal weight functions seem to be a flexible and natural choice. They allow the use of

- flat ellipsoids in flat regions, in order to avoid interference with other surface patches (if needed),
- cylindrical ellipsoids in curved regions with small Gauss curvature,
- and spherical supports in all other regions.

In order to reduce the memory needed and to improve the computational performance, we replace the ellipsoids by spheres if the two tangental axes are roughly equal, i.e. if they ratio is close to one.

[1] Note, that towards the ends the overlap of ellipsoids of the cylinder a transition of the curved surface to piecewise linear surface areas can be observed (upper rendering). Nevertheless, the surface quickly converges towards a cylinder when several supports overlap. In order to show this, we have cut off the cylinder at 0.5 of the half-axis length (lower rendering).

8.2 Defining and computing the feature adaptive surface

The generalized weighting scheme leads to slight changes in the basic approach: The points $\{\mathbf{p}_i\}$ now define a set of ellipsoidal support regions $E_i = \{\mathbf{x} | \|H_i^{-1}(\mathbf{x} - \mathbf{p}_i)\| < 1\}$ for each point, describing the region of space that is influenced by $\mathbf{p}_i$. In the union of these regions $\Omega = \cup_i E_i$ the two averaging functions are defined as before, using individual ellipsoidal weight functions for each points sample, i.e.

$$a : \Omega \to \Omega, \quad \mathbf{c}(\mathbf{x}) = \frac{\sum_i \theta(\|H_i^{-1}(\mathbf{x} - \mathbf{p}_i)\|)\mathbf{p}_i}{\sum_i \theta(\|H_i^{-1}(\mathbf{x} - \mathbf{p}_i)\|)} \tag{8.7}$$

and

$$\mathbf{n} : \mathcal{S} \to \mathbb{S}^2, \quad \max_{\|n(\mathbf{x})\|=1} \sum_i \theta(\|H_i^{-1}(\mathbf{x} - \mathbf{p}_i)\|)(\mathbf{n}(\mathbf{x})\mathbf{n}_i)^2. \tag{8.8}$$

The surface is then defined implicitly as before as $f(\mathbf{x}) = \mathbf{n}(\mathbf{x})^\mathsf{T}(\mathbf{c}(\mathbf{x}) - \mathbf{x}) = 0$. Note that the weight functions are now different for each point that contributes to the sum, however, the computation is not fundamentally more involved.

8.2.1 Ill-shaped ellipsoidal coverings

Choosing the locations and shapes of the ellipsoids has to be done with care. The methods described in Sections 8.3.1 and 8.3.2 usually produce good ellipsoidal coverings. However, in general it is possible that highly anisotropic ellipsoids lead to large regions in space that are only supported by a single sample (see Figure 8.2a)). For surfaces defined according to Section 8.2 this is a problem if the plane normal to $\mathbf{n}_i$ through $\mathbf{p}_i$ gets isolated. In these locations, the implicit function becomes zero, as $\|\mathbf{x} - \mathbf{c}(\mathbf{x})\|$ has no component in normal direction. Figure 8.3 (left) demonstrates the effect in a two-dimensional setting. Note, that also two or more supporting ellipsoids can produce the undesired additional zero transitions of the implicit function.

When only spherical support functions are used, this is unlikely to happen, as usually the only component which remains from cutting away other spheres contains the center (See Figure 8.2b)). Locations far away from a sample are also far away from the plane defined by the sample point and its normal because the neighboring samples are located (almost) coplanar. Therefore, their implicit values differ from zero. The result is a fair approximation of the surface's distance field.

If an ellipsoidal covering is ill-shaped, we use the following technique to avoid reconstructing surfaces from few samples protruding into empty space: A point $\mathbf{x}$ is considered only if it is close to its local centroid $\mathbf{c}(\mathbf{x})$ because $\mathbf{c}(\mathbf{x})$ is located inside the convex hull of the contributing samples, so it is necessarily close to the reconstructed surface (refer

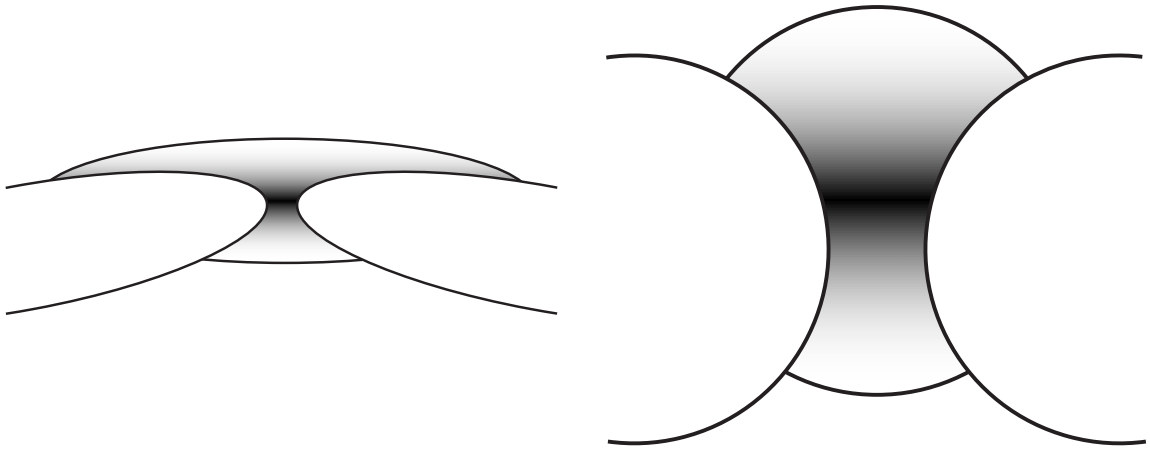

Figure 8.2: In these situations, the excentric locations of the planes defined by (p_i, n_i) are covered by other supports. No extra surface parts result in these cases. Note, that this is always the case when using spherical supports if we assume the neighboring samples to be close to coplanar.

to Figure 8.4). If excentric parts of an ellipsoid are isolated from the other ellipsoids, they are far away from the local centroid $\mathbf{c}(x)$ of these locations (i.e. the center of the ellipsoids). Setting a threshold to this distance eliminates these regions.

As the sizes of the ellipsoids vary, the threshold distance has to be adjusted to the ellipsoids. Note that the transformed difference vector $H_i^{-1}(\mathbf{x} - \mathbf{p}_i)$ already has length in the unit interval, i.e. a vector with almost unit length is far away from the center relative to the local scale. Averaging over these difference vectors

$$\mathbf{c}'(\mathbf{x}) = \frac{\sum_i w_i H_i^{-1}(\mathbf{x} - \mathbf{p}_i)}{\sum_i w_i} \tag{8.9}$$

also yields a vector with length between 0 and 1. We require

$$\epsilon < 1 - \|\mathbf{c}'(\mathbf{x})\| \tag{8.10}$$

for $\mathbf{x}$ to be a valid point on the surface. Figure 8.3 (right) shows the result for applying a cut-off with $\epsilon = 0.15$.

8.3 Computing weight functions

In the following two sections, we discuss how to choose weighting transforms H_i in different scenarios. A third method also allows to compute ellipsoidal weights, although we have, so far, only experimented with spherical weighting functions.

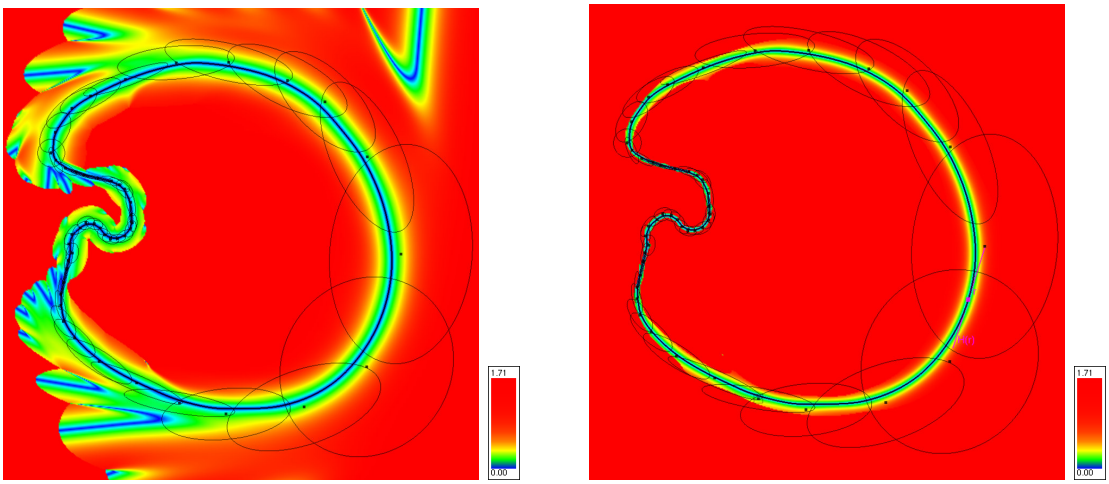

Figure 8.3: An example of undesired zero transitions in the implicit function (color-coded in blue) of an ill-conditioned two-dimensional setting of support ellipses (left) (For illustration purposes, the supporting ellipses are scaled down) The threshold for Equation 8.10 is applied ($\epsilon = 0.15$), erasing the undesired iso-surface parts (right).

8.3.1 Computing weights per sample point from clean samples

For dense clean samples of a surface our assumption is that the k-nearest neighbors of a point $\mathbf{p}_i$ in space are also close on the sampled surface. Some irregularities close to boundaries can be fixed by symmetrizing the k-nearest neighbor graph. Then, we can simply determine an ellipsoid covering the points by performing a covariance analysis [58] and an appropriate scale. Let $p_{i(1)}, \ldots, \mathbf{p}_{i(k)}$ be the k-nearest neighbors and set the covariance matrix

$$\Sigma_i = k^{-1} \sum_{j=1}^{k} (\mathbf{p}_{i(j)} - \mathbf{p}_i)(\mathbf{p}_{i(j)} - \mathbf{p}_i)^{\mathsf{T}}. \tag{8.11}$$

Then Σ_i contains squared lengths $\lambda_{i_k}^2$ of the main axes of an ellipsoid, which we want to use to cover the k points. Note that we cannot ensure a covering. Assuming that the k points are uniformly distributed over a planar surface, we can derive expected squared radii $r_{i_k}^2$ of an (ideal) enclosing ellipsoid. The relation, interestingly, depends on k and we first look at it in one dimension: For $k = 1$, we have $r^2 = \lambda^2$. For large k, many points contribute, and assuming a uniform distribution with zero mean asymptotically leads to

$$\lambda^2 = r^{-1} \int_0^r x^2 \, dx = \frac{1}{3} r^2. \tag{8.12}$$

For k points distributed uniformly on the surface we expect the radii to grow as $\sqrt{k}$. Together with the expected behavior for small and large k, we find that squared axes of

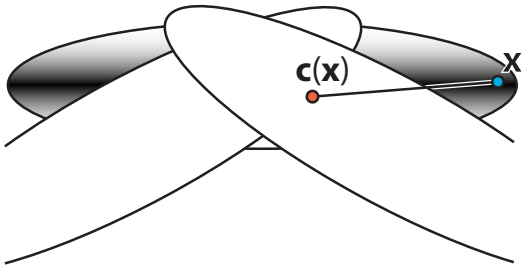

Figure 8.4: At locations inside excentrically isolated parts of the support ellipsoids $\|\mathbf{x}-\mathbf{c}(\mathbf{x})\|$ is large relative to the radii H_i of the ellipsoid. An upper bound on the local centroid of $H_i^{-1}(\mathbf{x}-\mathbf{p}_i)$set in order to eliminate these parts

the covariance matrix and squared radii of enclosing ellipsoids relate as

$$\gamma_k = \frac{1+\frac{2}{\sqrt{k}}}{3} = \frac{2+\sqrt{k}}{3\sqrt{k}}. \tag{8.13}$$

We use this relation to compensate for the effect of k and set the squared weighting transform to be $H_i^2 = \gamma_k^{-1}\Sigma_i$. In many cases we only need the square of H_i, however, if the root is needed, it can be obtained from the diagonalization of Σ_i as

$$H_i = \gamma_k^{-1/2} R_i \Lambda_i^{\frac{1}{2}}, \quad \Sigma_i = R\Lambda_i R^{\mathsf{T}}. \tag{8.14}$$

Figure 8.1 illustrates ellipsoids derived with this approach.

8.3.2 Computing weights for reduced sets from dense samples

In most cases, the original point set describing the surface contains more samples then necessary and the samples might contain noise. Furthermore, a small set of samples might be beneficial. A popular approach for decimating the number of primitives in mesh modeling is to collapse pairs of vertices [56, 41] and this has also been tried for point sets [82]. Our experience, however, shows that decimation is inferior to generating a reduced representation directly on the dense model.

This has recently been done very successfully for meshes [32], radial basis functions [81], and splats [104]. We believe that our representation is flexible enough to work well with either of these approaches. In particular, we have developed an approach reminiscent of Cohen-Steiner et al's [32], however, generating ellipsoids rather than planes and working directly on point sets.

The main idea is to maintain a set of proxy ellipsoids that approximate subsets of the point samples. Alternating, point samples are assigned to proxies and then proxies are recomputed based on the assigned samples.

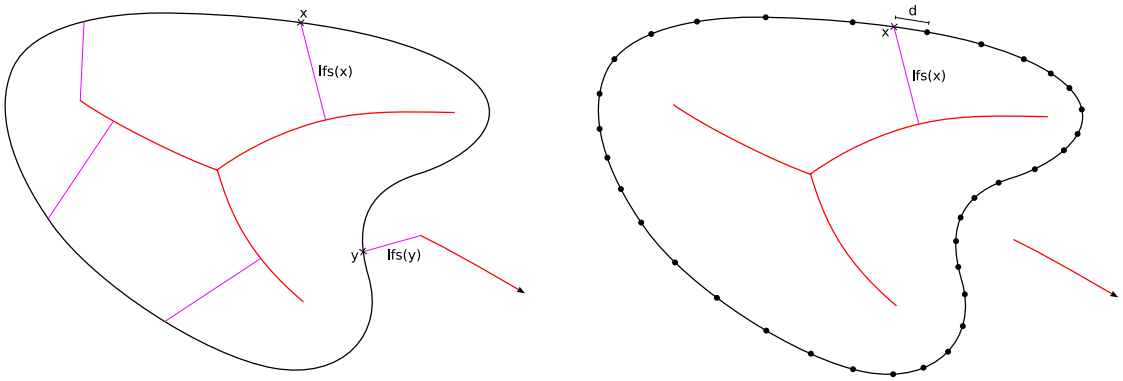

Figure 8.5: Left: a 2D-shape and its corresponding medial axis. The local feature size is the distance of a point **x** on the shape to the medial axis. Right: a sampling that is proportional to the local feature size.

Points are assigned to proxies based on the deviation of the point normal to the proxy normal and, as a secondary criterion, based on elliptically weighted distance to the center of the proxy. However, a point can be assigned to a different proxy only if one of its neighbors belongs to that proxy. This assures that points assigned to one proxy form connected components.

Proxies are computed from the assigned points and their neighbors to make sure that the ellipsoids sufficiently intersect. Proxy center and normal direction are computed by averaging. The ellipsoid in tangental directions is computed from a local covariance analysis and the axes are adjusted to cover all points.

8.3.3 Floater Approach

Here, our objective was to develop a resampling algorithm for point set surfaces, in the spirit of Witkin and Heckbert [103], where particles, restricted to the surface, repel each other. We want to adopt the sampling density to the surface characteristics. The approach we use is motivated by the sampling criterion proposed by Amenta et al. for curves [12] and for smooth surfaces [13]. It is based on the *local feature size*, which for each point of the surface is defined to be the distance from the point to the medial axis of the surface (see Figure 8.5, left for an illustration). The criterion basically says that the local sampling density should be inverse proportional to the local feature size (cf. Figure 8.5, right), i.e. the sampling density should be higher in regions of the surface that are near the medial axis and can be lower in regions that are farther away from the medial axis.

To use this sampling criterion we have to be able to compute the local feature size for each point of the surface. Typically this requires the computation of the medial axis

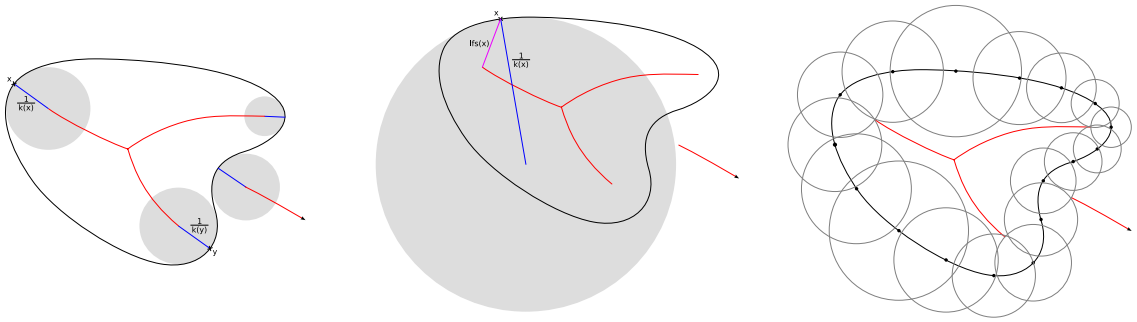

Figure 8.6: Left: The inverse of the curvature is an upper bound on the lfs. At Points, where the lfs is determined by a point on the boundary of the medial axis, the inverse of the curvature equals the lfs. Middle: On the other locations $\mathbf{x}$, $\text{lfs}(x) < 1/k(\mathbf{x})$. Right: Sampling based our estimate of the lfs, where the increase is restricted by $1/k(\mathbf{x})$ at neighboring points $\mathbf{x}$.

of the surface , which is a global operation. For instance, unrelated opposite parts of the surface that are close to each other influence the location of the medial axis. To avoid this expensive pre-computation we instead have developed an estimate of the local feature size that is based on the surface curvature. This allows better performance for the resampling and also enables dynamical, local resamplings of parts of the surface.

Our estimate uses the inverse of the maximum principal curvature, which is as an upper bound for the local feature size (cf. Figure 8.6, left). For points on the shape that are closest to the boundaries of the medial axis the local feature size equals the inverse of the principal curvature, i.e. $lfs(\mathbf{x}) = 1\ k(\mathbf{x})$. For all other points the local feature size is determined by other parts of the surface, i.e. $lfs(\mathbf{x}) < 1/k(\mathbf{x})$. Thus, it can not be computed locally (as shown in Figure 8.6, middle). Therefore, we additionally use the estimated local feature size at nearby points to restrict the rate of increase of the estimate and thereby smoothing the transition from regions with high curvature to regions with low curvature. The result is illustrated in figure (see Figure 8.6, right).

This estimate of the local feature size is by no means perfect, but has shown to work well for our purpose – using this estimate certainly produces better sampling results than approaches that are purely based on the curvature.

The idea for the algorithm itself, is to place particles onto the surface that repel each other and move on the surface. These particles are called *floaters*. We choose the repulsion force of the floaters proportional to the estimated local feature size, so that the floater density will be higher in regions near the medial axis and lower in regions farther away from the medial axis. The relaxation process of the floaters is iteratively simulated until an equilibrium is reached, the final positions of the floaters then form the new sampling of the surface. For more details, please refer to [18].

Our lfs-estimation is not expected to produce satisfactory results for simple shapes

that have few features. On more complex shapes, the medial axis has more boundaries, especially in the 3D-case. Hence, a large part of the surface points are closest to boundary points of the medial axis, where the lfs is exactly determined by $1/k$. For such models our estimation produces good results, as can be seen in Figure 8.7. The corresponding timings and parameters are found in Table 8.1. Note, how well the samples distribe at transitions from low to high curvature. So far, we have only experimented with weighting functions of spherically support, although ellipsoids can be used, as well.

ρ	Initial floaters	Time	Iterations	Resulting samples	$\Delta_{avg}(\mathcal{S}, \mathcal{S}')$	$\Delta_{max}(\mathcal{S}, \mathcal{S}')$
0.3	10000	22s	54	38013 (26.7%)	$5.98 \cdot 10^{-5}$	0.00686
0.5	10000	16s	89	14897 (10.4%)	0.000222	0.00412

For $\rho = 0.5$ a lower time step was necessary to reach convergence ($\Delta t = 0.1$).

Table 8.1: Timings and number of iterations for completing a sampling of the torso model. The parameter ρ describes the target spacing of the samples in relation to the estimated local feature size. The values $\triangle_{avg}$ and $\triangle_{max}$ indicate the average and maximum distance of the original samples to the resulting surface.

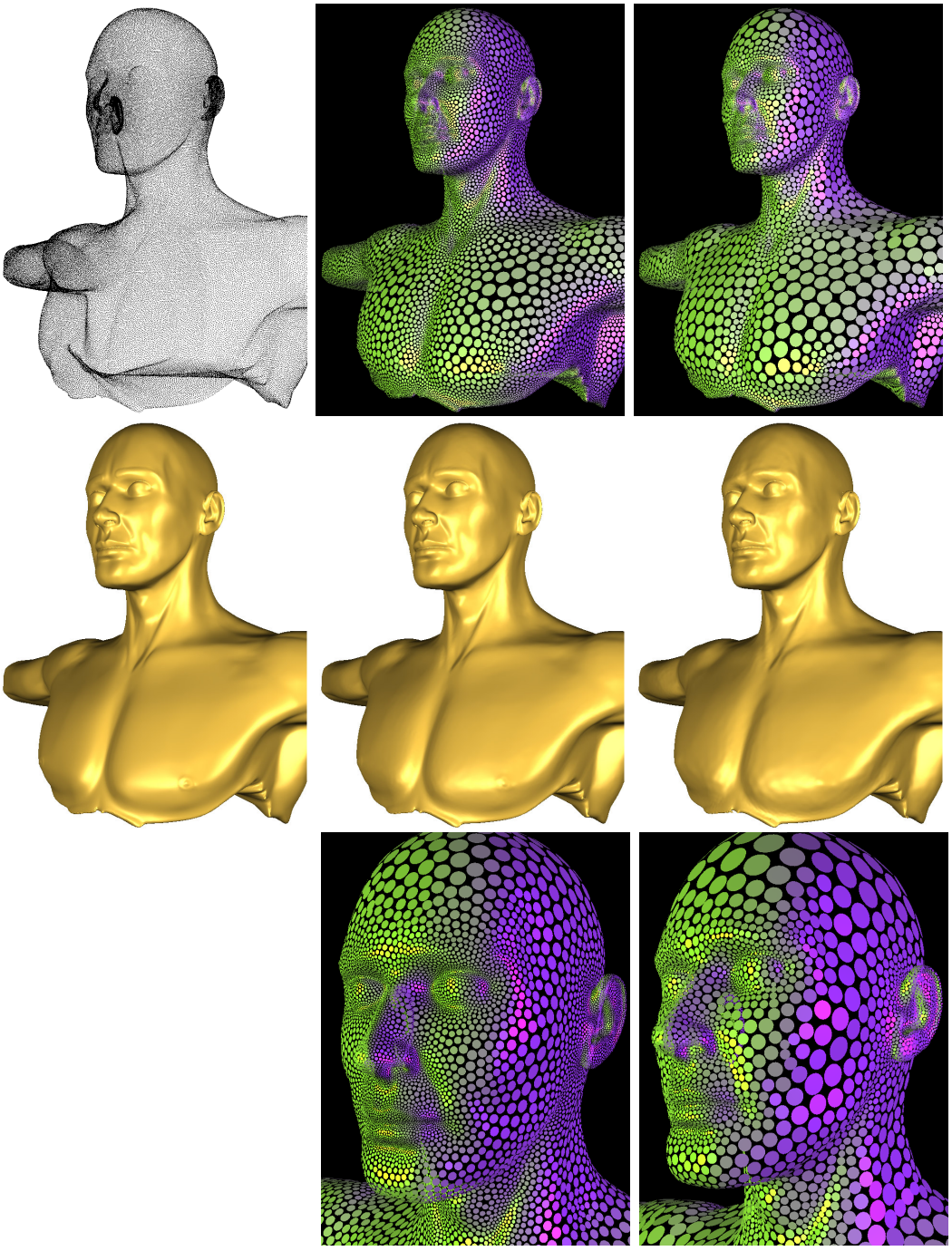

Figure 8.7: The Human torso model (courtesy of INRIA) defined by 142348 points and, from left to right, original, resamplings with $\rho = 0.3$ and $\rho = 0.5$. The middle row shows renderings of the surfaces defined by the corresponding samplings. The bottom row contains close up views of the resamplings.

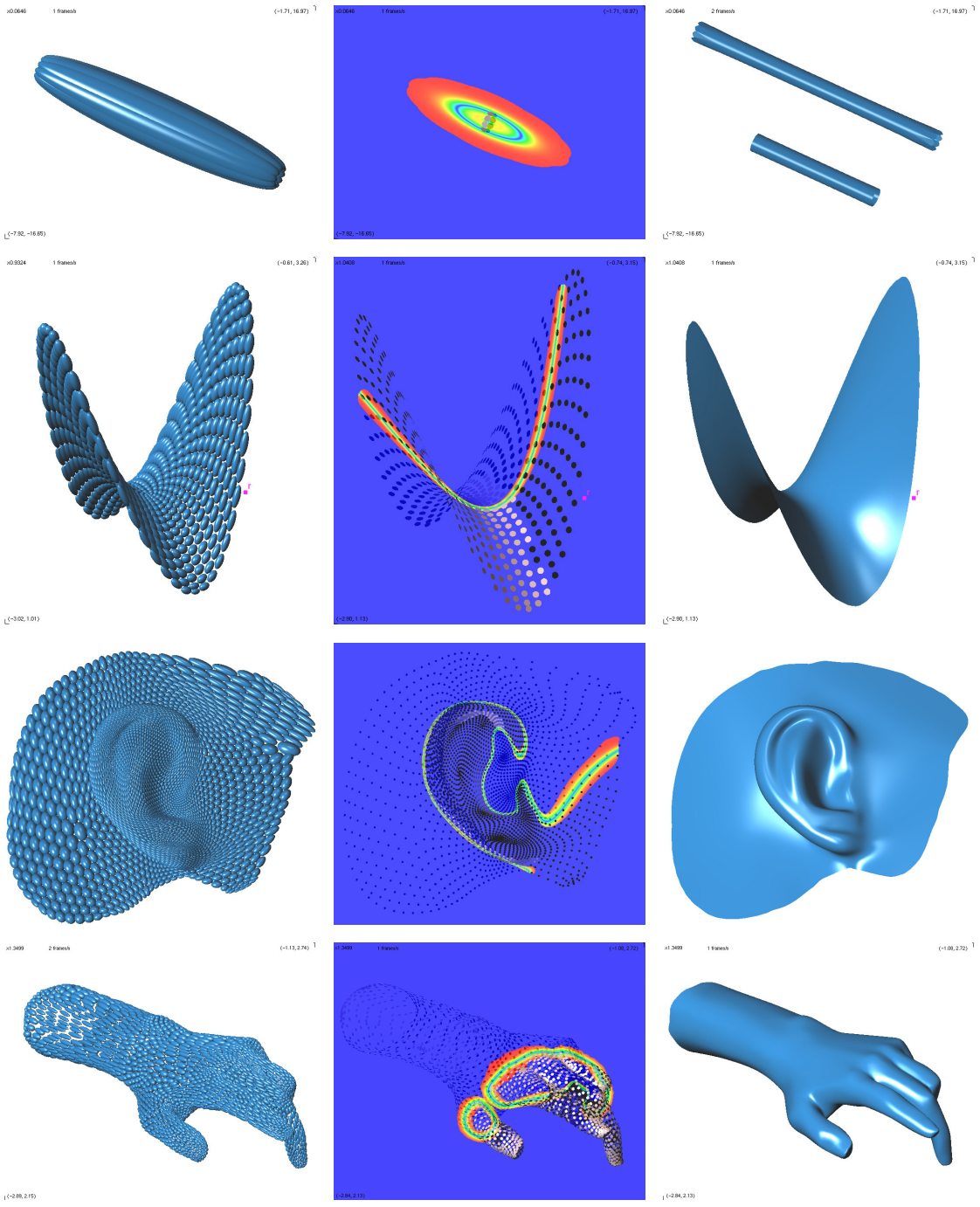

Figure 8.8: A series of models (cylinder, saddle, ear and hand), containing anisotropic features are represented by anisotropic ellipsoids (left column). For illustration purpose, the ellipsoids are scaled by factor 0.25, except for the cylinder model. The magnitude of the implicit function is visualized along a planar cut (middle column). The models are rendered using ray casting (right column). In the lower cylinder rendering, the cut-off threshold is set to 0.5, in order to exclude the endings, where the overlap of the support ellipsoid decreases.

Chapter 9

Piecewise Smooth Surfaces

The approach we presented so far as well as other point-based approaches [111, 83, 62] allow to represent smooth surfaces, only. The surface is usually defined (or can be interpreted) as the level set of a smooth function [68, 9, 4, 16]. Consequently, only closed smooth surfaces can be represented. The extension discussed in Section 6 additionally enables us to deal with boundaries that result implicitly in areas where samples are missing in tangential direction.

In this Chapter we present a modelling approach for high quality piecewise smooth surfaces based on point samples and a cell complex. The surface may have boundaries, sharp edges, corners, self intersections, or other features, and these features can be prescribed exactly.

We model a piecewise smooth surface as a cell complex. For a 2-surface embedded in 3-space this complex consists of surface patches, curve segments, and points (i.e. cells of dimensions 2,1, and 0). Each of the classes is represented by samples. Elements of different dimension are glued together based on connectivity information. This concept is inspired by the construction of cell complexes (or CW complexes, see [51] for a modern introduction), though we use a relaxed definition (see Section 9.1 for details).

Following the works of Levin [68], Alexa et al. [9], and Amenta & Kil [16] we define the surface as the set of stationary points under a projection The following contributions lead to our modeling framework based on point-sampled cell complexes:

1. We extend the projection operators to account for the connectivity among cells, i.e. lower-dimensional boundary cells are interpolated and the cell is contained within the boundary (Section 9.2). The geometry of the cell complex can then be defined as the stationary points of the projection.

2. Based on this basic modeling approach, we discuss tangent constraints for boundaries to allow modeling tangent continuous geometry across cells (Section 9.3).

The resulting projection procedure can be used in a ray tracing environment for surface visualization [1, 98].

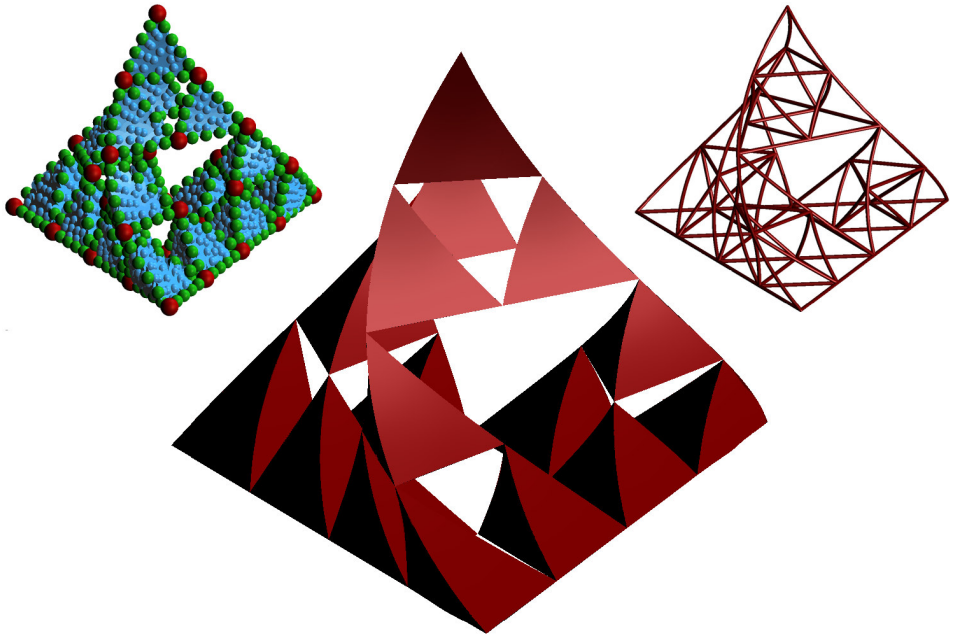

Figure 9.1: Example of a geometric model that can easily be represented with our approach. All curves as well as the surface patches are represented by point samples. Connectivity information is used to glue surface patches to curves and curves to points.

Some of the techniques we build upon are also used for reconstructing a surface from raw sample data, however, our goal is different: we aim at providing a versatile shape representation for modeling that allows prescribing features exactly. For that we expect these features (e.g. sharp edges or non-manifold points) to be represented explicitly. A preprocessing step using techniques for feature detection in point-sampled data [47, 84, 80, 40] could be used to fully automate or at least aid the generation of our representation from real-world sample data.

In that sense, our approach is comparable with other modeling approaches such as NURBS or subdivision surfaces. We briefly discuss the differences of the approaches and their implications in Section 9.4 and finally mention limitations and future work.

9.1 The cell complex

In a cell complex, each surface patch would be open and attached to curve segments forming a boundary; and each curve segment would have endpoints as boundary. This allows understanding the homotopy type of the set from the cells and their connectivity (the 0-cells correspond to critical points of a Morse function [51]). This interesting correspondence has been used in graphics for analyzing and representing topological

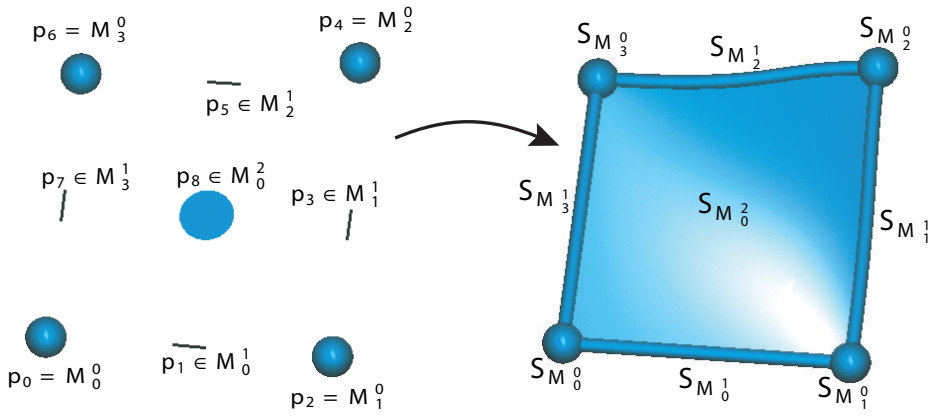

Figure 9.2: The representation of the point-sampled cell complex and the basic construction of the geometry by projection. Each d-cell $\mathcal{M}_i^d$ is represented by samples and defines its own geometry $\mathcal{S}_i^d$

properties of shapes [93, 50, 54, 78].

We use relaxed understanding of cell complexes by also including closed curves and surfaces (i.e. without boundary). This makes modeling some common shapes easier, such as a surface bounded by a closed curve (see Figure 9.3 for an unorientable surface bounded by one closed (knotted) curve), or smooth closed surfaces. If necessary for topological reasons, users could still restrict themselves to using only cells with boundary. In any case, independent surface patches meet in lines; independent lines meet in points.

This concept obviously generalizes to hyper-surfaces of arbitrary dimension, however, for clarity of the exposition we concentrate on 2-surfaces embedded in 3-space and sometimes illustrate properties on curves embedded in the plane.

The surface representation then consists of the (manifold) cells $\{\mathcal{M}_i^d\}$, where $d = \{0, 1, 2\}$ denotes the dimension of the cell with index i. The topology of the complex is represented by the connectivity among the cells; the geometry of each cell is represented by point samples in space. For our purposes it is convenient to simply think of $\mathcal{M}_i^d$ as a set containing the cells $\mathcal{M}_j^{d-1}$ attached to it as well as the points $\mathbf{p}_k \in \mathbb{R}^3$ defining its geometry (see Figure 9.2 for an illustration). In practice, the points might be equipped with additional information about the local normal and/or tangent direction.

The surface defined by this complex will be defined as the set of stationary points of a projection operator $P(\mathbf{x}), \mathbf{x} \in \mathbb{R}^3$, i.e.

$$\mathcal{S} = \{\mathbf{x} | P(\mathbf{x}) = \mathbf{x}\} \tag{9.1}$$

For the definition of this projection operator, we proceed as follows. We make use our general definition of closed manifolds without boundary of arbitrary dimension and restrict the locally approximated functions to points contained in a cell. This defines the

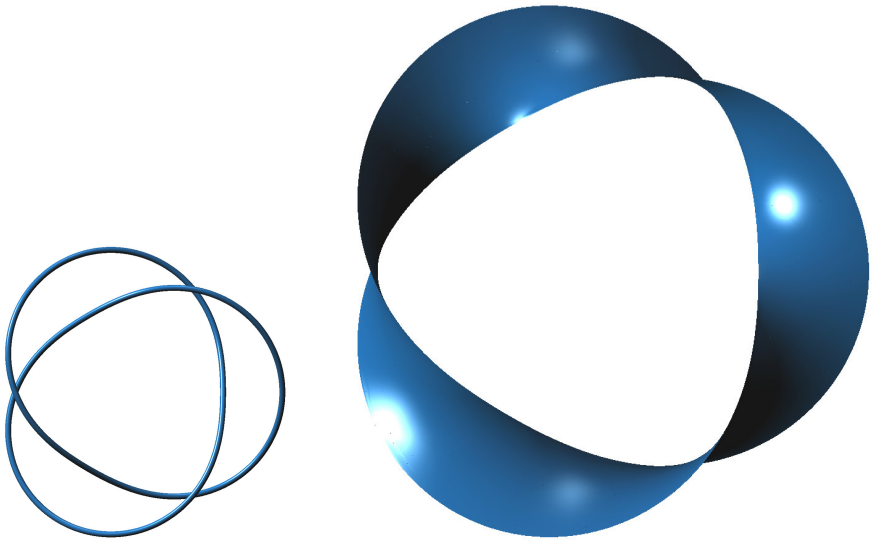

Figure 9.3: A simple closed curve (sampled with 160 points) represents the boundary of a closed surface (sampled with 560 points). The curve is here visualized as a generalized cylinder with constant radius. Note that this is not a cell complex because the boundary curve itself has no boundary (i.e. has no endpoints), however, we allow using closed cells without boundary for convenient modeling and give up the correspondence between homotopy of the shape and connectivity in the complex.

geometry of the individual cells. Then, we extend the corresponding projection operator to respect the connectivity structure of the complex.

9.2 Projecting onto the complex

The surface comprised by the cell complex is going to be defined by extending the projection operator. For this it is important to understand the additional requirements resulting from gluing the cells together:

1. A cell meets the cells on its boundary. This means, the geometry within the cell is not only influenced by its sample points but also by the cells it connects to. Furthermore, this influence has to be chosen so that the geometries of cells coincide. As an example, a curve in space has to interpolate an endpoint bounding it.

2. The connected cells are boundaries. Because the geometry of cells is defined by unconnected samples, it is not clear from the samples alone where the cell

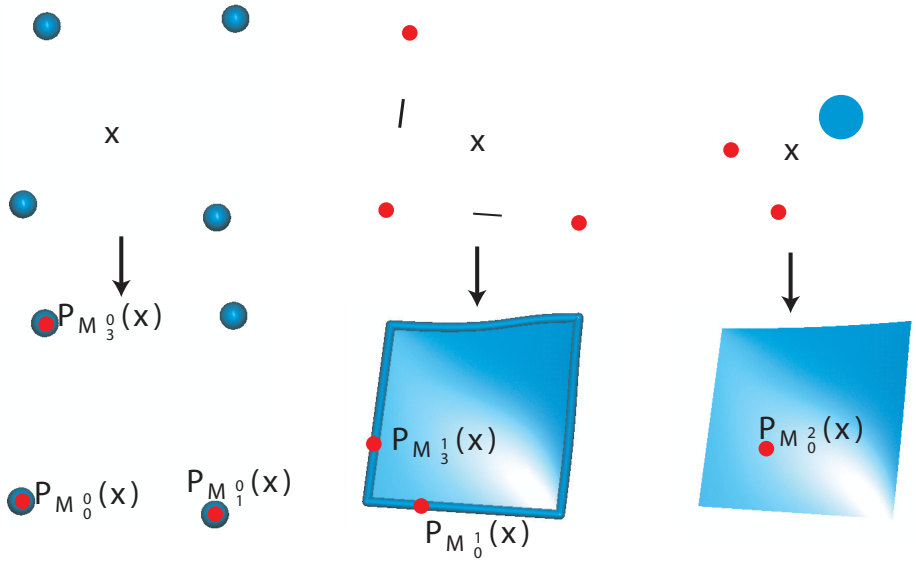

Figure 9.4: The representation of the point-sampled cell complex and the basic construction of the geometry by projection. Each cell $\mathcal{M}$ is represented by samples. The projection $P_{\mathcal{M}}(\mathbf{x})$ is computed inductively, by first projecting onto the 0-cells, then using the results (together with the samples in 1-cells) for projecting onto 1-cells, and so on. The geometry $\mathcal{S}_{\mathcal{M}}$ of each cell is defined as the stationary points $P_{\mathcal{M}}(\mathbf{x}) = \mathbf{x}$ of the projection.

ends. This should be defined by the boundary cells, e.g. a line should end in its endpoints.

The projection defined in Section 3.1 works for any closed cell without boundary. For cells with boundary, the projection is defined inductively, by taking into account the results of projecting onto any connected lower-dimensional cell:

1. Compute the projection onto all 0-cells M_i^0 supporting $\mathbf{x}$. This simply yields their locations. Let these locations be $\mathbf{q}_i^0(\mathbf{x})$.

2. Repeat the following step for all d-cells, $d > 0$: Collect all samples of d-cells supporting $\mathbf{x}$. For each cell $\mathcal{M}_j^d$ consider the collected samples it contains as well as the projections $\mathbf{q}_i^{d-1}(\mathbf{x})$ onto the $(d-1)$-cells on its boundaries. The computations detailed below will yield projections $q_j^d(\mathbf{x})$ onto the d-cells.

One sequence of steps (see Figure 9.4) is enough to decide whether a point is on the surface. If it is necessary to project points in space the sequence has to be repeated until convergence. Note that in a ray tracing environment the repeated evaluation on the ray is equivalent to this repetition.

9.2.1 Interpolating attached cells of lower dimension

The following procedure applies to cells of all dimensions and we use $\mathbf{q}_j(\mathbf{x})$ as an abbreviation for $\mathbf{q}_j^{d-1}(\mathbf{x})$, i.e. the projections onto the boundaries $\mathcal{M}_j^{d-1}$ of $\mathcal{M}_i^d$. To interpolate the neighboring cells, the computation of $\mathbf{c}_\mathcal{M}(\mathbf{x})$ needs to take into account the $\mathbf{q}_j(\mathbf{x})$ as follows:

$$\mathbf{c}_\mathcal{M}(\mathbf{x}) = \frac{\sum_i \theta(\|\mathbf{x}-\mathbf{p}_i\|)\mathbf{p}_i + \sum_j \omega(\|\mathbf{x}-\mathbf{q}_j(\mathbf{x})\|)\mathbf{q}_j(\mathbf{x})}{\sum_i \theta(\|\mathbf{x}-\mathbf{p}_i\|) + \sum_j \omega(\|x-\mathbf{q}_j(\mathbf{x})\|)} \tag{9.2}$$

Here, ω is a locally supported weight function that goes to infinity as its argument goes to zero. This leads to an interpolation of the projection $\mathbf{q}_j(\mathbf{x})$ as $\mathbf{x}$ gets closer to $\mathbf{q}_j(\mathbf{x})$ (see [67]). For reasons that are detailed in a technical report [8] we use the following function:

$$\omega(r) = \begin{cases} \left(\ln \frac{r}{h}\right)^2 - \left(\frac{r}{h}\right)^2 + 2\frac{r}{h} - 1 & 0 \le r \le h \\ 0 & r > h \end{cases} \tag{9.3}$$

With this choice of weight function, $\mathbf{q}(\mathbf{x}) = \mathbf{x}$ implies $\mathbf{c}_\mathcal{M}(\mathbf{x}) = \mathbf{x}$. Now, $\mathbf{c}(\mathbf{x}) = \mathbf{x}$ enforces $\mathbf{N}^\mathsf{T}(\mathbf{c}(\mathbf{x}) - \mathbf{x}) = \emptyset$, independent of $\mathbf{N}$ – so all points $\mathbf{q}(\mathbf{x}) = \mathbf{x}$ belong to the attached cell $\mathcal{M}_i^d$ independently of its approximated tangent frame at $\mathbf{x}$. Since the set $\{\mathbf{q}_j(\mathbf{x}) = \mathbf{x}\}$ actually represents the geometry of $\mathcal{M}_j^{d-1}$, we have enforced the required interpolation behavior.

Based on this modification a projection is computed, as described in Section 3.1, by repeatedly applying $Q(\mathbf{x})$ (see Eq. 3.1).

9.2.2 Restricting the cell by its boundaries

The basic idea is that $\mathcal{M}_i^d$ is bounded by $\mathcal{M}_j^{d-1}$ and exists only in half space through $\mathbf{q}_j$, namely the one that contains the local centroid of the samples in $\mathcal{M}_i^d$ (see Figure 9.5). For this, $\mathbf{q}_j$ defines a half-space as follows: Because the geometry is interpolated, $\mathbf{T}_{\mathbf{q}_j\mathbf{x}}\mathcal{M}_j^{d-1}$ (the tangent frame of the boundary in $\mathbf{q}_j(\mathbf{x})$) is a subspace of $\mathbf{T}_{\mathbf{q}_j\mathbf{x}}\mathcal{M}_i^d$ (the tangent frame of the cell in $\mathbf{q}_j(\mathbf{x})$). The complement $\mathbf{b}_j = \mathbf{T}_{\mathbf{q}_j\mathbf{x}}\mathcal{M}_i^d - \mathbf{T}_{\mathbf{q}_j\mathbf{x}}\mathcal{M}_j^{d-1}$ is a direction that is tangent to $\mathcal{M}_i^d$ but orthogonal to the bounding cell $\mathcal{M}_j^{d-1}$. This is the required direction of the half space that bounds $\mathcal{M}_i^d$ at $\mathbf{q}_j(\mathbf{x})$.

Let the centroid $\mathbf{c}_\mathcal{M}(\mathbf{x})$ be in the positive halfspace, i.e. $\mathbf{b}_j^\mathsf{T}(\mathbf{c}_\mathcal{M}(\mathbf{x}) - \mathbf{q}_j) > 0$. If the projection result $Q_\mathcal{M}^\infty(\mathbf{x})$ yields $\mathbf{b}_j^\mathsf{T}(Q_\mathcal{M}^\infty(\mathbf{x}) - \mathbf{q}_j) < 0$ then the projection is reset to $\mathbf{q}_j$. This test has to be performed independently for all $\mathbf{q}_j$. In case the positive halfspace conditions fails for several $\mathbf{q}_j$ the one closest to $\mathbf{x}$ is chosen (however, we assume that this situation is avoided by appropriate modeling, as it affects the smoothness of the resulting surface, see next section).

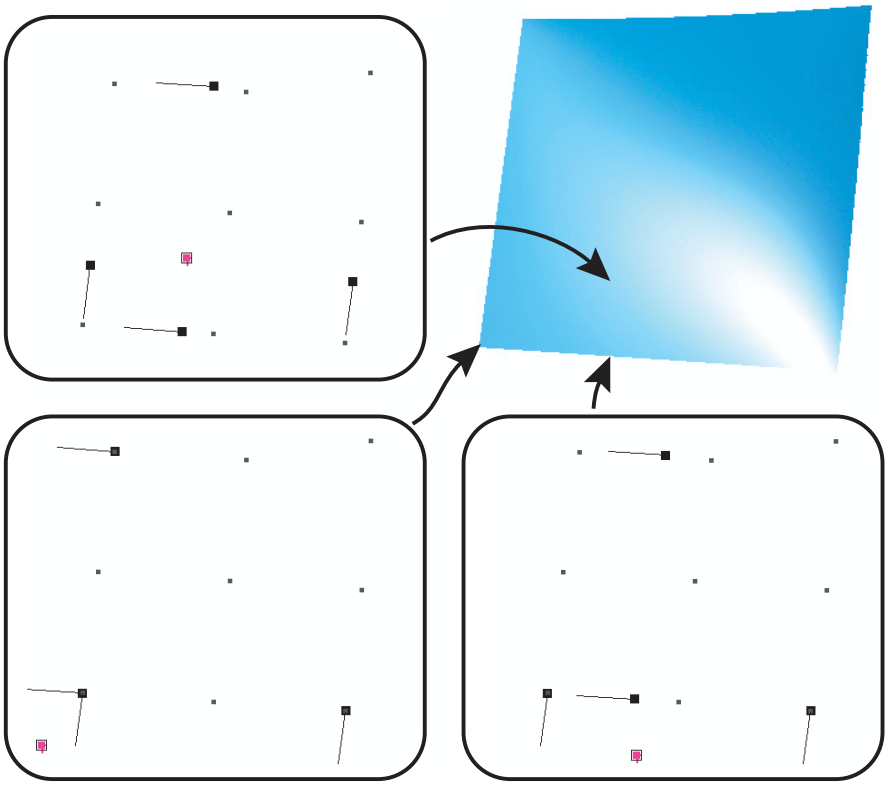

Figure 9.5: The projection onto cells is restricted to lie within a halfspace through the projection onto its boundary. The normal of the half space is computed as the direction that is tangent to the cell but not to its boundary.

The result of the projection can be defined as

$$P_{\mathcal{M}}(\mathbf{x}) = \begin{cases} Q^{\infty}_{\mathcal{M}}(\mathbf{x}) & \forall j \ \mathbf{b}_j^{\mathsf{T}}(Q^{\infty}_{\mathcal{M}}(\mathbf{x}) - \mathbf{q}_j) \geq 0 \\ \mathbf{q}_j & \mathbf{b}_j^{\mathsf{T}}(Q^{\infty}_{\mathcal{M}}(\mathbf{x}) - \mathbf{q}_j) < 0 \end{cases} \tag{9.4}$$

Note that the geometry of a cell is, thus, defined as the intersection of half spaces, i.e. it is should be convex. However, the geometry might also be concave, as long as the weighted centroid $\mathbf{c}_{\mathcal{M}}(\mathbf{x})$ of each point in the cell is also contained in the cell. Figures 9.7 (middle image), 9.9, 9.10 demonstrate that, in practice, smooth concave boundaries are no problem. The issue is different where boundaries meet non-smooth. Here, the boundary has to be convex. The reason for this difference is related to the continuity of the projection and its distance field, which we discuss next.

9.2.3 Continuity of the projection and support sizes

It has been shown that the set of stationary points of the projection is well defined [16] and smooth if tangents are computed based on eigenanalysis [29]. As our extended definition also involves the result of projection onto lower dimensional cells, the projection operator has to be a continous function on $\mathbb{R}^n$ for the surface to be smooth.

For the following argument we assume the projection is orthogonal[1]. Then the projection is orthogonal to the distance function of the cells.

For closed smooth cells without boundary this is a smooth function except for the critical points of the distance function. Thus, we require that the weight functions of the sample points are locally supported and that the union of local supports contains no critical points of the distance function. This is equivalent to the construction of r-samples, where the local sampling density is proportional to the local feature size, i.e. the distance to the medial axis. Because we know the geometry of each individual cell is smooth, we can require the sampling density to be large enough such that the local support of each sample point is smaller than the distance to the closest critical point on the distance function.

Note that computing individual projections for cells meeting in sharp features is the fundamental idea that makes it possible to satisfy this condition. A sharp feature meets with the medial axis and would be impossible to avoid the critical points of the distance function. This is why smooth concave boundaries are possible, while sharp concave corners would require special treatment.

For cells with boundary, the definition of $\mathbf{c}_{\mathcal{M}}(\mathbf{x})$ and $T_{\mathbf{x}}\mathcal{M}$ is a function of projections $\mathbf{q}_j(\mathbf{x})$ onto lower dimensional cells. Note that the projection onto 0-cells is constant and, therefore, smooth everywhere. For all other cells, the extended projection operator is equivalent to the case without boundary except for the case distinction in Eq. 9.4. To understand the resulting continuity we inspect the distance field: This is the combination of the distance field of the cell $\mathcal{M}_i^d$ and the bounding cell $\mathcal{M}_j^{d-1}$ joined at a hyper plane where all distances are equal. So the projection is at least C^0, but obviously not tangent continuous across the hyperplane. This is illustrated in Figure 9.6, where we have used the projection operator as defined so far for generating the left image.

The work of Amenta & Kil [16] suggests that the implicit function theorem applies to the defining function of the stationary points of the projection and, consequently, the surface inherits its smoothness from the functions it is constructed from. To make the surface smooth, we have to redefine the projection so that the result is smooth across the hyperplane. Remember that the result of the projection $Q^{\infty}_{\mathcal{M}}(\mathbf{x})$ is smooth – our idea for smoothing the projection is to 'blend in' the bounding behavior: Let h_j be the

[1] In our current implementation we only use the 'almost' orthogonal projection [7] without any noticable problems

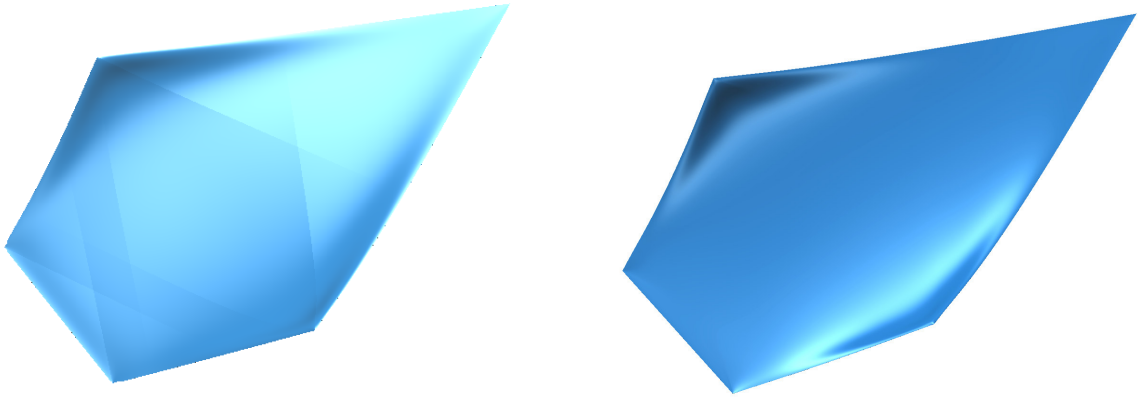

Figure 9.6: A simple shape constructed with one sample per element (5 points, 5 curves, 1 face). The bounding behavior of the projection operator results in the projection being not tangent continuous across the half planes defining the boundaries of the curves. This shows in tangent discontinuities of the surface if interior angles at the boundaries are larger than 90 degrees. A smoothed version of the bounding behavior increases the smoothness of the surface (see main text).

distance to hyper plane through $\mathbf{q}_j$

$$h_j = \mathbf{b}_j^{\mathsf{T}}(Q_{\mathcal{M}}^{\infty}(\mathbf{x}) - \mathbf{q}_j) \tag{9.5}$$

then in the case $h_j < 0$ we don't use $\mathbf{q}_j$, but blend it with $Q_{\mathcal{M}}^{\infty}(\mathbf{x})$ using a Gaussian weight function based on h_j^2. Thus, our operator becomes:

$$S_{\mathcal{M}}(\mathbf{x}) = \begin{cases} Q_{\mathcal{M}}^{\infty}(\mathbf{x}) & \forall j \; h_j \geq 0 \\ e^{-h_j^2} Q_{\mathcal{M}}^{\infty}(\mathbf{x}) + \left(1 - e^{-h_j^2}\right) \mathbf{q}_j & h_j < 0 \end{cases} \tag{9.6}$$

The visual result is shown in Figure 9.6.

Note that this modification results in giving up the projection property: No point in the half space $h_j < 0$ can project onto itself as the result of Eq. 9.6 is always biased towards $\mathbf{q}_j$. Consequently, this half space contains no geometry of the cell, as required. But the reference points $S_{\mathcal{M}}(\mathbf{x})$ generated for computing the geometry of attached cells are using a smooth function, resulting in a smooth surface.

Alltogether, we have constructed a definition of a piecewise smooth surface that meets continuous but not tangent-continuous in prescribed, piecewise smooth bounding curves. In the next section we will discuss variants for defining the behavior across boundaries.

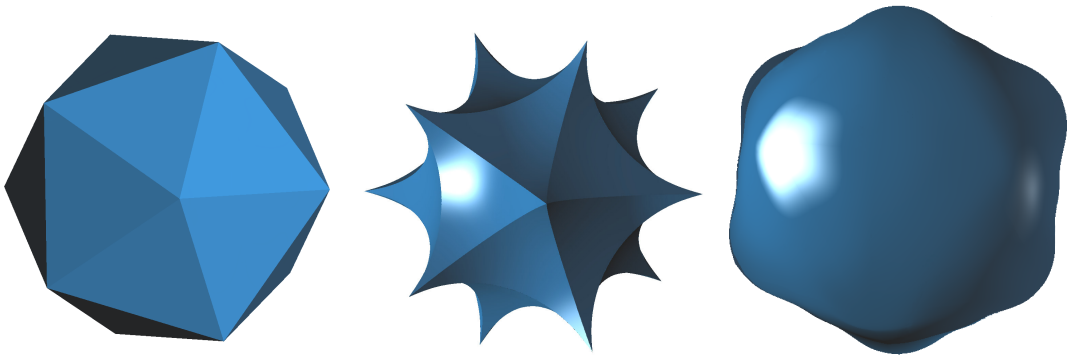

Figure 9.7: Several icosahedra represented with 4 samples per curve and 4×4 samples per face but different tangent constraints on the boundaries of each surface patch.

9.3 Constraints

With our definition so far, smooth cells meet in smooth boundaries, however the continuity across boundaries cannot be controlled. In many modeling situations, however, it is desirable to achieve a certain degree of continuity across boundaries.

Among the different possibilities we concentrate on tangent continuity. This requires the following constraints:

1. Interpolating a boundary cell with prescribed tangents for the attached cells, e.g., prescribing the tangents of a curve meeting a point or prescribing normals of a surface meeting a curve. This allows prescribing tangents for surface patches (see middle image of Figure 9.7, each surface patch has individual boundary constraints) or joining cells across boundaries with continuous tangents (see Figure 9.8).

2. Also prescribing tangent frames of a dimension higher than the attached cells, i.e. prescribing normals for surfaces in points to enforce tangent continuity across bounding curves meeting in a point (right image of Figure 9.7).

9.3.1 Prescribing tangents on boundaries

Remember that we already require the interpolation of the tangent frame of bounding cells. However, the tangent frame of a bounding cell has one dimension less and the remaining tangent direction would be free.

To prescribe this tangent we add it as additional information to the samples of a boundary. Then, this information is automatically interpolated when a cell meets its boundary (as the position is interpolated). For example, if a curve sample carries two tangent directions, these tangents are taken into consideration when the extended

projection is computed on any surface connecting to the curve. More formally, the samples of a boundary cell $\mathcal{M}_j^{d-1}$ are equipped with tangent frames $\{T_k\}$ of dimension d. The projection onto the cell yields the position $\mathbf{q}_j(\mathbf{x})$ as well as the combined tangent frame $T_{\mathbf{q}_j(\mathbf{x})}\mathcal{M}_j^{d-1}$. This tangent frame is taken into consideration when the tangent frame $T_{\mathbf{x}}M_i^d$ of an attached cell $\mathcal{M}_i^{d+1}$ is computed. Let $\mathbf{T}_j$ be a matrix representing $T_{\mathbf{q}_j(\mathbf{x})}\mathcal{M}_j^{d-1}$, then the tangent frame is extracted from the eigenvectors of the co-variance matrix

$$\mathbf{C}_{\mathcal{M}^d}(\mathbf{x}) = \sum_i \theta(\|\mathbf{x}-\mathbf{p}_i\|)(\mathbf{x}-\mathbf{p}_i)(\mathbf{x}-\mathbf{p}_i)^\mathsf{T} + \omega(\|\mathbf{x}-\mathbf{q}_j(\mathbf{x})\|)\mathbf{T}_j\mathbf{T}_j^\mathsf{T} \tag{9.7}$$

Note that this only ensures that the tangent frames used for the projection operator are interpolated. It is not immediately obvious that the prescribed tangents are actually interpolated by the geometry (see [7] for an explanation of the potential pitfalls). Our particular choice of weight functions ensures that tangents are interpolated [8].

9.3.2 Prescribing higher dimensional tangent frames

To ensure a tangent continuous surface across boundaries it is not enough to prescribe tangent frames in the cells attached to the surface (i.e. the curves). In addition, the tangents need also be interpolated across points bounding the curves. Again, these tangent constraints have to be added as additional information to the points.

The main idea for ensuring interpolation of this tangent frame is to modify the attached curves accordingly: Curves attached to a point with a tangent frame defining a surface need to meet the point with a tangent direction contained in that tangent plane. Note that this is different from prescribing the tangent direction of the curve in that point, as there remains a degree of freedom for the tangent direction within the tangent plane defined in the point.

So let the samples of $\mathcal{M}_j^{d-1}$ be equipped with tangent frames $\{T_k\}$, however, now they have dimension $d+1$. Again, they are combined to yield the tangent frame $T_{\mathbf{q}_j(\mathbf{x})}\mathcal{M}_j^{d-1}$. In theory, we could add the corresponding weighted matrix directly to the matrix of co-variances $\mathbf{C}_{\mathcal{M}^d}$ as before, as the largest eignevalues of this matrix would still correspond to eigenvectors in the tangent space of $\mathcal{M}_i^d$, since the tangent frames of this cell would contribute additional co-variance. In practice, however, we found the computations to be unstable when $\mathbf{x}$ gets close to $\mathbf{q}(\mathbf{x})$.

So, instead, we extract a tangent frame of the right dimension before adding it to the matrix of co-variances. First, we compute the tangent frame $T'_{\mathbf{x}}\mathcal{M}_i^d$ of $\mathcal{M}_i^d$ without considering $T_{\mathbf{q}_j(\mathbf{x})}\mathcal{M}_j^{d-1}$. Second, we project $T'_{\mathbf{x}}\mathcal{M}_i^d$ onto $T_{\mathbf{q}_j(\mathbf{x})}\mathcal{M}_j^{d-1}$ yielding $\hat{T}_{\mathbf{q}_j(\mathbf{x})}\mathcal{M}_j^{d-1}$, which now has the right dimension d. Third, the co-variance matrix is built as before, using $\hat{T}$ instead of T. Thus, in the case of interpolation (i.e. $\mathbf{q}(\mathbf{x}) = \mathbf{x}$) the 'natural' tangent frame of $\mathcal{M}_i^d$ projected onto the prescribed tangent frame is used.

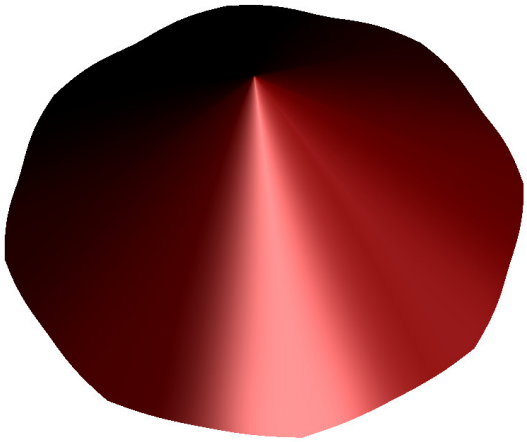

Figure 9.8: A cone modeled from 8 triangular cells (where each curve is sampled with one point and each face is sampled with three points) with prescribed normals on the edges and points (except for the tip, which has no tangent or normal constraints). Note that the edges on the rim are not tangent continuous as there are no tangent (only normal) constraints in the points on the rim. Also, while the surface is tangent continuous over edges everywhere, the edges are more visible towards the top of the cone, where the tangents vary quicker. This is a consequence of our uniform sampling assumption.

We describe the idea for the case of surfaces in $\mathbb{R}^3$: Let points and curves carry normals to prescribed the tangent frame of the surface. Then the tangents of the curves have to be orthogonal to the normals in the points as the curve meets the endpoint. Let the curve have tangent $\mathbf{T}(\mathbf{x})$ and the normal constraint in the endpoint be $\mathbf{n}$. The tangent added to the co-variance matrix is then $(\mathbf{n} \times \mathbf{T}(\mathbf{x})) \times \mathbf{n}$, as this is the projection of $\mathbf{t}$ into the tangent plane defined by $\mathbf{n}$.

9.4 Discussion

Using a cell complex for describing geometric models is not new. Shinagawa et al. [93] build shapes from the critical points of a morse function to ensure correct topology of the shape. Rossignac [86] extends the classic boundary representation idea to include non-manifold configurations. However, they are not concerned with how the geometry of the shape is actually represented. Our approach could be understood as a particular instance of their concepts.

A mesh is another instance of a cell complex, and quite commonly used for modeling shapes. In contrast to our approach, meshes are typically associated with piecewise linear surfaces. Using smooth geometry, many meshes could probably converted by extracting

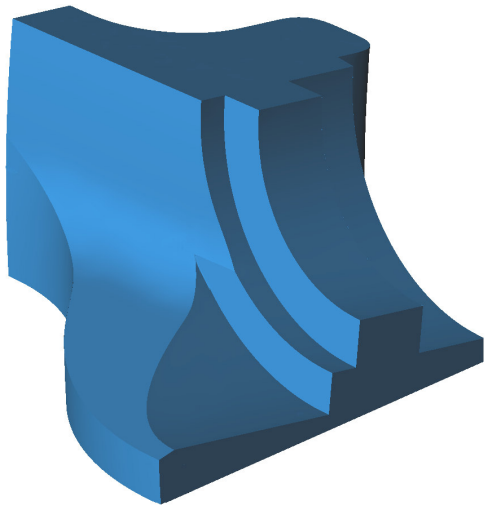

Figure 9.9: The well-known fandisk model has been converted to a sampled cell complex using the tagged sharp edges. Untagged vertices have been used as samples for the faces, tagged vertices are either samples on curves or sharp points.

a sub-complex of smooth pieces (see Figure 9.9 for the result of a converted mesh, where we have used the tagged sharp edges to define the curves of the complex). One possibility to adapt the geometric fidelity to the given viewing or modeling requirements are multi-resolution representations [56, 108, 63]. Depending on the number of cells in the complex, these approaches could also be interesting in our setting – and they would be easy to adapt.

The idea of gluing together surface patches also appears in approaches inspired by the construction of manifolds. Grimm & Hughes [44] use overlapping charts and smooth mappings to blend neighboring surfaces. Later, Ying & Zorin [107] refine this approach to achieve arbitrary smoothness. A main difference of our approach (also in comparison to other smooth surface constructions) is that a cell is always glued to a bounding cell of lower dimension (possibly with constraints) and not to neighboring cells of the same dimension.

The classic approach to piecewise smooth surfaces are splines, which might also be reconstructed from points [57]. Recent developments ease the insertion of constraints [91, 90] at arbitrary locations, and in that sense have something in common with our approach. In general, we see our requirement of uniform sampling as a limitation, and discuss this in the next Section.

Even more flexibility in terms of the topology and the constraints may be achieved through subdivision surfaces that interpolate boundary constraints [20, 22, 73, 66]. Judging from Nasri and Sabin's [77] taxonomy and the remarks on possible configurations for non-manifold subdivision [106], our framework seems to support a more general class

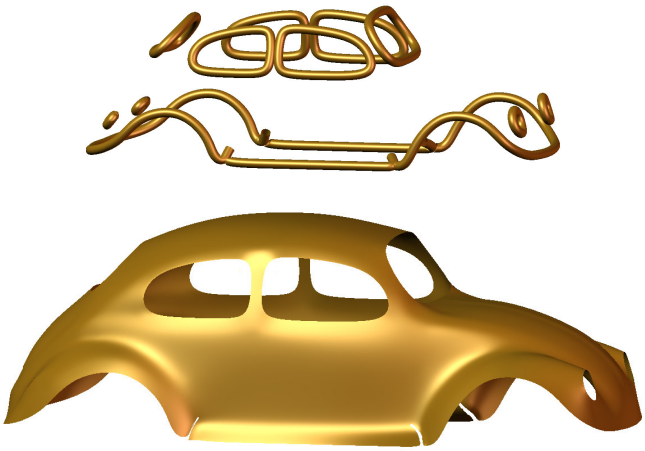

Figure 9.10: The boundary curves are a convenient way of trimming the surface. The upper picture shows the trimming (or boundary) curves rendered as cylinders. The lower pictures shows the trimmed surface.

of surfaces, and also generalizes easier to higher dimensions.

More important, however, in this comparison are the fundamental differences resulting from the domains in which the surface is defined. Splines are defined over a parameter domain, where values in the parameter domain map to coordinates in space. Subdivision surfaces are generated from a control mesh in space, and each face in the control mesh could be interpreted as the parameter domain of a corresponding part of the surface. In contrast, the point samples in our approach directly influence their neighborhood in space.

This difference becomes important in interactive modeling environments when the user picks a location on the surface, which has to be changed subsequently. Parametric representations require to first compute the corresponding parameter location and then to adapt the control net to reflect the required changes in space. This procedure typically involves numerical optimization and might be only approximate [65, 21] – in our approach ray-surface intersection and projection onto the surface are inherent to the surface definition, and changes defined in space can be directly reflected by adjusting the samples' locations or other information attached to them.

An example are trimming curves (see Figure 9.10 for a trimmed surface), which are defined by space curves attached to the smooth surface. I an interactive application, the user could define these curves by interactively drawing onto the surface, and the intersections with the surface would yield the point samples for the definition of the curve.

Chapter 10

Conclusions

This chapter concludes the thesis, by summarizing and discussing the main contributions and also by giving some directions for future work:

- **Definition of Curves and Surfaces:**

  Inspired by the MLS surface, we have presented an implicit definition of smooth curves and surfaces defined by points. The implicit function is composed of the local centroid of the input points and a tangent frame that reflects how the points are locally arranged. This makes it possible to describe manifolds of arbitrary dimension. In the special case of hyper-surfaces, the tangent frame can be represented by the only normal direction.

  Extremal surfaces are a generalization of our surface, in that the relation of the considered point in space and its local centroid can be replaced by an arbitrary energy function. However, we find that the local centroid is the natural choice in order to describe point set surfaces. Our surface shares the same properties with the MLS surface and extremal surfaces.

  The evaluation of the implicit function can be performed locally by only considering a small subset of the points, when compactly supported functions are used to weight the input points. We have discussed appropriate choices of weighting functions for maintaining sufficient smoothness of the resulting surface. In order to efficiently determine the relevant points, spacial data-structures have to be applied.

- **Operators for Projecting onto Curves and Surfaces:**

  For computing points of the curve or surface, we have presented several projection operators that perform very well. The algorithms converge very fast, typically within less than five iterations. Compared to the computation of the MLS projection procedure as proposed by Alexa et. al [9, 10] we are in the order of two

magnitudes faster. The reason is the strategy for establishing the reference plane: While they move the projected point $\mathbf{q}$ by alternating between rotation of the reference plane (including $\mathbf{q}$) around the input point and adjusting its distance, we rotate the reference plane around $\mathbf{q}$ and project onto it, instead. This way, much larger steps of $\mathbf{q}$ are performed, leading to a much better convergence behavior.

The way Amenta and Kil [16] define their projection is the same as ours, considering our particular energy function, i.e. the distance of $\mathbf{x}$ to $\mathbf{c}(\mathbf{x})$ is minimized along $\mathbf{n}$, where $\mathbf{x} - \mathbf{c}(\mathbf{x}) \perp \mathbf{n}(\mathbf{x})$. This is the projection of $\mathbf{x}$ onto the reference plane defined by $\mathbf{q}$ and $\mathbf{n}(\mathbf{q})$.

We have also presented an orthogonal version of our projection operator, and projections can also be performed onto curves – or more generally to manifolds of arbitrary dimension.

- **Ray-surface Intersection Operator:**

  We have presented a ray-surface intersection operator that in principle works according to our projection operator. Instead of projecting into direction $\mathbf{n}$, we restrict the search to the ray, by intersecting the ray and the approximation. Again, the algorithm converges very fast.

  We have also discussed alternative ray-surface intersection operators and how to successfully apply them to our surface. Sphere tracing is possible, provided that the normals are consistently oriented on the consecutive ray locations. Thereby, the intersection can be approached from both sides. The same applies to ray marching, where we have to detect a change of sign. Ray marching can be an efficient alternative, when the SIMD technique to sample a short ray segment is exploited. In this case, the ray has to be clipped against very tight voxels. Linear interpolation can be applied to increase precision, as the implicit values approximate the distance field very well.

- **Ray Tracing / Ray Casting Point Set Surfaces:**

  We have explained how to employ the ray-surface intersection algorithms for ray casting or ray tracing and discussed corresponding efficiency aspects. In order to obtain the frontmost intersection of the ray and the surface, we have to choose an appropriate initial location, which is sufficiently close to the surface. In case the ray misses the surface, the algorithm has to be restarted at the subsequent regions that potentially contain the intersection. Tightening these regions, leads to a considerably better performance, as fewer rays have to be examined, and the resulting ray-segments are shorter.

- **Spatial Data-Structures for Point-based Surface Representations:**

In order to find the relevant ray-segments for evaluating the ray-surface intersection algorithms, a suitable volume, bounding the surface, has to be build. We have proposed several types of bounding volumes, each covering a region around a sample. Together these volumes form a bounding volume of the surface. The choice depends on whether normals are given with the input points or not. For supplied normals, a flat bounding box is preferable, as it tightly encloses the surface. Otherwise, we propose bounding-spheres.

To efficiently query the bounding volume, we have to build a hierarchical data-structure. This is also necessary in the context of determining the relevant points for local evaluation of the implicit function. We have shown how to integrate these different requirements in a single framework. This was done both for static models and for dynamic models that are subject to continuous modifications.

The discussed acceleration techniques are sufficient to achieve interactive frame-rates during rendering by ray-casting or ray-tracing, which was shown by Wald [97]. However, this premises to build very tight kd-trees, which is not applicable for dynamically changing objects. Also, when dealing with static objects, a time consuming pre-process is often not desirable, e.g. during or after scanning.

- **Adaptive Sampling Exploiting Image- and Object-space Coherence:**

  We have presented a sampling framework for the fast display of intersectable shapes. It is adaptive in image space, however, it creates a representation with controlled error in object space. With this representation the rendering task can be off-loaded to current graphics hardware, which keeps the CPU free for computing ray surface intersections. In addition, the object space representation can be re-used in new views, leading to an effective way of exploiting coherence.

  Our approach is transparent to the surface definition actually used. It is expected to generally improve the performance of ray casting systems without complicating existing implementations. However, we are particularly interested in the application to high quality surface from point samples without the need to perform an expensive global surface reconstruction. Compared to other rendering approaches for point sets (such as splatting) the quality of rendered images (especially in close-up views) is better.

  An interesting extension of our system is to include shadow rays, or, more generally, secondary rays as in ray tracing. For a single static image, this would be no problem. Exploiting coherence over several views is more difficult - a possible solution is to split refinement criteria into view dependent and view independent.

  The system might also be distributed on several processors or displays. An interesting challenge would be to re-use quads as they transit from one processor/display to the next.

In our current implementation, we render valid front-facing quads, by activating OpenGL backface culling. Although this works well, we plan to experiment with more conservative culling methods in future versions of our software, as OpenGL renders a quad as soon as one vertex normal passes the front-facing test. We, however, would prefer discarding the quad as soon as one normal fails the front-facing test.

Artifacts potentially occur, when previously visible, valid quads become occluded by other parts of the scene, in transformed, subsequent frames. Since valid quads are preserved and then possibly fully occlude a query quad, new occluders are not detected. This could be solved by casting a single ray into the center of each query quad, thereby detecting closer occluders at the cost of a simple intersection with the spatial data structures. However, we have found that simply limiting the size of the quad-queue (as described above) and thereby limiting the life-span of valid quads, alleviates this problem and results in good surface approximations.

- **Feature Adaptive Representation:**

  We have generalized the weighting scheme typically used with Point Set Surfaces, which allows point-based models to represented in a feature adaptive way. More fundamental than it might appear at first sight, we propose that each sample carries its own weight function (i.e. local scale), rather than averaging local scales directly. Statistics demonstrate that only this change has the promise of improving the properties of the reconstructed surface, when the samples are not regular.

  Besides spherical weights per sample, we suggest to use ellipsoidal weights, in anisotropic surface areas. Surface samples can have cylindrical shape if the principal curvatures are different. This is not only convenient to keep the representation compact, it also restricts the space around complex features as much as possible, avoiding expensive calculations away from complex, and consequently densely sampled features. As far as we know, this is the first approach that integrates anisotropic, curvature dependent sampling into mesh-less representations.

- **Differential Geometry:**

  We have explicitly considered the normals of Point Set Surfaces and its variants. We have demonstrated that the normal to the approximating tangent frame is not the surface normal. Based on our implicit version of the surface description we have derived how to gain higher order information of the surface, i.e. we have shown how to compute exact surface normals and curvature. The exact normals allow computing orthogonal projections – we feel that these tools help to solidify the computational framework of Point Set Surfaces.

- **Bounded Curves and Surfaces:**

We have enhanced our basic manifold definition, in order to represent surfaces and curves with boundaries. Our approach is based on the distance of the considered point in space to its local centroid. This is advantageous because it reflects the distance to the input points very well. It is a smooth function that yields smooth boundaries, when setting a threshold. No explicit boundary information has to be provided for directly processing surfaces containing holes, etc. The only quantities that have to be considered, are computed during the basic evaluation, anyway. To avoid artifacts, adequate radii for bounding the surface have to be chosen. We have discussed how to derive these from the given threshold, assuming a uniform sampling.

Our approach can also be applied in the feature adaptive setting. In order to maintain the smoothness of the boundary, this requires a smooth adaptation the threshold according to the radii of the weighting functions. A drawback here is that the complexity of the boundary curve depends on the sampling density of the surface. Therefore, we find that the explicit boundaries, discussed in the context of piecewise smooth surfaces, are superior in terms of representing bounded surfaces. However, there are situations where the boundaries are not known, e.g. after scanning. Here, the implicit definition is advantageous.

- **Non-orientable Surfaces:**

  We have demonstrated that our surface also can be globally non-orientable. However, we are restricted to manifold surfaces, not allowing self-intersections. In the example of the Klein-bottle, we had to cut holes into the surface to avoid these situations. It is clear that the case of self-intersection cannot be handled fully automatically because it is inherently ambiguous.

- **Piecewise Smooth Surfaces:**

  The basic definition of curves and surfaces does not allow to describe surfaces that self-intersect, nor can features, such as sharp edges or cusps, be handled. Therefore, we have finally enhanced our manifold definition, to precisely describe the more general class of piecewise smooth surfaces – still in the setting of point-based representations. Inspired by cell complexes, we model surface patches, curve segments and points and glue them together based on explicit connectivity information.

  A considerable problem with our and other approaches to computing piecewise smooth surfaces is numerical robustness. Mathematically elegant ideas, such as interpolation enforced by singular weight functions or a surface defined by the zero set of an unsigned function, are notoriously ill-behaved in floating point implementations (which is what we currently use). A better understanding of the surface

or numerically more stable weight functions are necessary to make our approach stable enough for real-world applications.

We have experimented with enforcing tangent constraints on curves and points. While the data points are interpolated if singular weight functions are used, applying this interpolatory scheme has two deficiencies: it does not preserve convexity and again, it is numerically unstable. In addition, some kind of shape control would be desirable, as there are various shapes with different characteristics that interpolate the given tangents.

In our implementation we have only considered uniform sampling. Quite generally, this assumption seems to be a limiting factor. Adaptive sampling could be achieved in different ways: 1. Each cell uses its own sampling density. Since the surface can be arbitrarily subdivided into cells, this would enable us to define geometry at an appropriate scale. 2. Geometry could be adaptively sampled, e.g. using our feature adaptive representation.

The representation of geometry by samples in space and the basic primitives of projection and surface intersection are useful operators for interactive shape modeling. We are looking forward to a modeling system in the spirit of the Pointshop3D system [111, 83], amended with the cell complex structure and using the extended projection operators we have introduced here. This would be a versatile general purpose application for modeling shapes.

Appendix A

Curriculum Vitae

Anders Adamson was born in September 6, 1973 in Vallentuna, Sweden. He received his computer science diploma from the Technische Universität Darmstadt, Germany in April 2002. His dimploma thesis was awarded with the 2002 INI-GraphicsNet Award for Thesis Excellence. Since October 2002 he has worked as a research and teaching assistant in the Interactive Graphics Systems Group, headed by Prof. José L. Encarnação and Prof. Dieter W. Fellner, Department of Computer Science at the Technische Universität Darmstadt, under the supervision of Prof. Marc Alexa. In December 2006 he won the Best Paper Award in the Category Research within the INI-GraphicsNet Foundation.

Publications

Articles in International Journals

- Adamson, A. and Alexa, M. (2006). Point-Sampled Cell Complexes. *ACM Transactions on Graphics (Proceedings of SIGGRAPH 2006)*, 25 (3): 671-680.

- Adamson, A. and Alexa, M. (2006). Anisotropic Point Set Surfaces. *Computer Graphics Forum*, 25 (3): 717-724 (previously published at Afrigraph 2006)

- Marc Alexa, Anders Adamson. Interpolatory Point Set Surfaces - Convexity and Hermite Data. Accepted for publication in *ACM Transactions on Graphics*.

Full Papers in Refereed Conference Proceedings

- Anders Adamson, Marc Alexa and Andrew Nealen. (2005). Adaptive Sampling of Intersectable Models Exploiting Image and Object-space Coherence. In *Proceedings of the ACM SIGGRAPH 2005 Symposium on Interactive 3D Graphics and Games (SI3D'05)*, pp. 171-178

- Anders Adamson and Marc Alexa. Approximating Bounded, Non-orientable Surfaces from Points. In *Proceedings of Shape Modeling and Applications 2004 (SMI'04)*, pp. 243-252

- Marc Alexa and Anders Adamson (2004). On Normals and Projection Operators for Surfaces Defined by Point Sets. In *Proceedings of Eurographics Symposium on Point-based Graphics (SPBG'04)*, pp. 149-156

- Anders Adamson and Marc Alexa. Approximating and Intersecting Surfaces from Points. In *Proceedings of the 2003 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing (SGP'03)*, pp. 245-254

- Anders Adamson and Marc Alexa. Ray Tracing Point Set Surfaces. In *Proceedings of Shape Modeling International 2003 (SMI'03)*, pp. 272-279

Bibliography

[1] Bart Adams, Richard Keiser, Mark Pauly, Leonidas J. Guibas, Markus Gross, and Philip Dutre. Efficient Raytracing of Deforming Point-Sampled Surfaces. *Computer Graphics Forum*, 24(3):677–684, 2005.

[2] Anders Adamson and Marc Alexa. Approximating and Intersecting Surfaces from Points. In *Proceedings of the Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 230–239. Eurographics Association, Jun 2003.

[3] Anders Adamson and Marc Alexa. Ray Tracing Point Set Surfaces. In M.S. Kim, editor, *SMI '03: Proceedings of Shape Modeling International 2003*, pages 272–279, Los Alamitos, May 2003. IEEE Computer Society.

[4] Anders Adamson and Marc Alexa. Approximating Bounded, Non-orientable Surfaces from Points. In *SMI '04: Proceedings of Shape Modeling Applications 2004*, pages 243–252. IEEE Computer Society, 2004.

[5] Anders Adamson, Marc Alexa, and Andrew Nealen. Adaptive Sampling of Intersectable Models Exploiting Image and Object-space Coherence. In *SI3D '05: Proceedings of the 2005 symposium on Interactive 3D graphics and games*, pages 171–178, New York, NY, USA, 2005. ACM Press.

[6] M. Alexa, M. Gross, M. Pauly, H. Pfister, M. Stamminger, and M. Zwicker. *Point-Based Computer Graphics*. SIGGRAPH'04 Course Notes #6, SIGGRAPH-ACM publication, 2004.

[7] Marc Alexa and Anders Adamson. On Normals and Projection Operators for Surfaces Defined by Point Sets. In Marc Alexa, Markus Gross, Hanspeter Pfister, and Szymon Rusinkiewicz, editors, *Proceedings of Eurographics Symposium on Point-based Graphics*, pages 149–156. Eurographics, 2004.

[8] Marc Alexa and Anders Adamson. Interpolatory point set surfaces - convexity and hermite data, 2006. manuscript.

[9] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Point Set Surfaces. In *Proceedings of the conference on Visualization '01*. IEEE Computer Society, 2001.

[10] Marc Alexa, Johannes Behr, Daniel Cohen-Or, Shachar Fleishman, David Levin, and Claudio T. Silva. Computing and rendering point set surfaces. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):3–15, 2003.

[11] Pierre Alliez, David Cohen-Steiner, Olivier Devillers, Bruno Levy, and Mathieu Desbrun. Anisotropic polygonal remeshing. *ACM Transactions on Graphics. Special issue for SIGGRAPH conference*, pages 485–493, 2003.

[12] Nina Amenta, Marshall Bern, and David Eppstein. The crust and the beta-skeleton: Combinatorial curve reconstruction. *Graphic Models and Image Processing*, 60(2 of 2):125–135, 1998.

[13] Nina Amenta, Marshall Bern, and Manolis Kamvysselis. A new voronoi-based surface reconstruction algorithm. *Proceedings of SIGGRAPH 98*, pages 415–422, July 1998.

[14] Nina Amenta, Sunghee Choi, and Ravi Kolluri. The power crust, unions of balls, and the medial axis transform. *International Journal of Computational Geometry and its Applications*, 19(2-3):127–153, 2001.

[15] Nina Amenta and Yong J Kil. The domain of a point set surfaces. *Eurographics Symposium on Point-based Graphics*, 1(1):139–147, June 2004.

[16] Nina Amenta and Yong Joo Kil. Defining Point-set Surfaces. *ACM Transactions on Graphics*, 23(3):264–270, 2004.

[17] Marco Attene and Michela Spagnuolo. Automatic surface reconstruction from point sets in space. *Computer Graphics Forum*, 19(3):457–465, 2000.

[18] Dominik Bathon and Anders Adamson. Technical report, 2006.

[19] Larry Bergman, Henry Fuchs, Eric Grant, and Susan Spach. Image rendering by adaptive refinement. In *Proceedings of the 13th annual conference on Computer graphics and interactive techniques*, pages 29–37. ACM Press, 1986.

[20] Henning Biermann, Adi Levin, and Denis Zorin. Piecewise smooth subdivision surfaces with normal control. In *Proceedings of ACM SIGGRAPH 2000*, Computer Graphics Proceedings, Annual Conference Series, pages 113–120, July 2000.

[21] Henning Biermann, Ioana Martin, Fausto Bernardini, and Denis Zorin. Cut-and-paste editing of multiresolution surfaces. *ACM Transactions on Graphics*, 21(3):312–321, July 2002.

[22] Henning Biermann, Ioana M. Martin, Denis Zorin, and Fausto Bernardini. Sharp features on multiresolution subdivision surfaces. *Graphical Models*, 64(2):61–77, March 2002.

[23] J. D. Boissonnat and S. Oudot. Provably good surface sampling and approximation. In *SGP '03: Proceedings of the 2003 Eurographics/ACM SIGGRAPH symposium on Geometry processing*, pages 9–18, Aire-la-Ville, Switzerland, Switzerland, 2003. Eurographics Association.

[24] Jean-Daniel Boissonnat. Geometric structures for three-dimensional shape representation. *ACM Transactions on Graphics*, 3(4):266–286, 1984.

[25] Jean-Daniel Boissonnat and Steve Oudot. Provably good sampling and meshing of surfaces. *Graph. Models*, 67(5):405–451, 2005.

[26] Mario Botsch and Leif Kobbelt. High-quality point-based rendering on modern GPUs. In *Proceedings of Pacific Graphics 2003*, pages 335–343, 2003.

[27] Mario Botsch, Michael Spernat, and Leif Kobbelt. Phong splatting. In *Eurographics Symposium on Point-Based Graphics 2004*, pages 25–32, 2004.

[28] Mario Botsch, Andreas Wiratanaya, and Leif Kobbelt. Efficient high quality rendering of point sampled geometry. In *Rendering Techniques 2002: 13th Eurographics Workshop on Rendering*, pages 53–64, June 2002.

[29] Peer-Timo Bremer and John C. Hart. A sampling theorem for mls surfaces. In *Symposium on Point - Based Graphics 2005*, pages 47–54, June 2005.

[30] Thibaut Brusseaux and Anders Adamson. Technical report, 2004.

[31] Loren Carpenter. The a -buffer, an antialiased hidden surface method. In *SIGGRAPH '84: Proceedings of the 11th annual conference on Computer graphics and interactive techniques*, pages 103–108, New York, NY, USA, 1984. ACM Press.

[32] David Cohen-Steiner, Pierre Alliez, and Mathieu Desbrun. Variational shape approximation. *ACM Transactions on Graphics*, 23(3):905–914, August 2004.

[33] Carsten Dachsbacher, Christian Vogelgsang, and Marc Stamminger. Sequential point trees. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 657–662, New York, NY, USA, 2003. ACM Press.

[34] Bruno Rodrigues de Araujo and Joaquim Armando Pires Jorge. Curvature dependent polygonization of implicit surfaces. In *SIBGRAPI '04: Proceedings of the Computer Graphics and Image Processing, XVII Brazilian Symposium on (SIBGRAPI'04)*, pages 266–273, Washington, DC, USA, 2004. IEEE Computer Society.

[35] Tamal K. Dey, Samrat Goswami, and Jian Sun. Extremal surface based projections converge and reconstruct with isotopy, 2005. manuscript.

[36] Tamal K. Dey and Piyush Kumar. A simple provable algorithm for curve reconstruction. In *Proceedings of the Tenth Annual ACM-SIAM Symposium on Discrete Algorithms*, pages 893–894, N.Y., January 17–19 1999. ACM-SIAM.

[37] Tamal K. Dey and Jian Sun. An adaptive mls surface for reconstruction with guarantees. In *ACM Symposium on Geometry Processing*, pages 43–52, 2005.

[38] Herbert Edelsbrunner and Ernst P. Mücke. Three-dimensional alpha shapes. In *VVS '92: Proceedings of the 1992 workshop on Volume visualization*, pages 75–82, New York, NY, USA, 1992. ACM Press.

[39] Shachar Fleishman, Daniel Cohen-Or, Marc Alexa, and Cláudio T. Silva. Progressive point set surfaces. *ACM Trans. Graph.*, 22(4):997–1011, 2003.

[40] Shachar Fleishman, Daniel Cohen-Or, and Cláudio T. Silva. Robust moving least-squares fitting with sharp features. *ACM Transactions on Graphics*, 24(3):544–552, August 2005.

[41] Michael Garland and Paul S. Heckbert. Surface simplification using quadric error metrics. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, pages 209–216, August 1997.

[42] Bernd Gärtner. Fast and robust smallest enclosing balls. In *ESA '99: Proceedings of the 7th Annual European Symposium on Algorithms*, pages 325–338, London, UK, 1999. Springer-Verlag.

[43] M. Gopi, S. Krishnan, and C. T. Silva. Surface reconstruction based on lower dimensional localized delaunay triangulation. In M. Gross and F. R. A. Hopgood, editors, *Computer Graphics Forum (Eurographics 2000)*, volume 19(3), 2000.

[44] Cindy M. Grimm and John F. Hughes. Modeling surfaces of arbitrary topology using manifolds. In *Proceedings of SIGGRAPH 95*, Computer Graphics Proceedings, Annual Conference Series, pages 359–368, August 1995.

[45] J. P. Grossman and William J. Dally. Point sample rendering. In *9th Eurographics Workshop on Rendering*, pages 181–192, 1998.

[46] Gael Guennebaud and Mathias Paulin. Efficient screen space approach for Hardware Accelerated Surfel Rendering . In *Vision, Modeling and Visualization , Munich, 19/11/03-21/11/03*, pages 485–495. IEEE Signal Processing Society, 2003.

[47] S. Gumhold, X. Wang, and R. McLeod. Feature extraction from point clouds. In *Proc. 10th International Meshing Roundtable*, pages 293–305, Sandia National Laboratories, Newport Beach, CA, 2001.

[48] John C. Hart. Ray Tracing Implicit Surfaces. Technical Report EECS-93-014, WSU, 1993.

[49] John C. Hart. Sphere tracing: Simple robust antialiased rendering of distance-based implicit surfaces. In *SIGGRAPH 93 Modeling, Visualizing, and Animating Implicit Surfaces course notes*, pages 14–1 to 14–11. 1993.

[50] John C. Hart. Using the CW-complex to represent the topological structure of implicit surfaces and solids. In *Proc. Implicit Surfaces '99*, pages 107–112, Sept. 1999.

[51] Allan Hatcher. *Algebraic Topology*. Cambridge University Press, Cambridge, UK, 2002.

[52] Vlastimil Havran. *Heuristic Ray Shooting Algorithms*. Ph.d. thesis, Department of Computer Science and Engineering, Faculty of Electrical Engineering, Czech Technical University in Prague, November 2000.

[53] Paul S. Heckbert. Fundamentals of texture mapping and image warping. Master's thesis, 1989.

[54] Masaki Hilaga, Yoshihisa Shinagawa, Taku Kohmura, and Tosiyasu L. Kunii. Topology matching for fully automatic similarity estimation of 3d shapes. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 203–212, August 2001.

[55] R. W. Hockney and J. W. Eastwood. *Computer simulation using particles*. Taylor & Francis, Inc., Bristol, PA, USA, 1988.

[56] Hugues Hoppe. Progressive meshes. In *Proceedings of SIGGRAPH 96*, Computer Graphics Proceedings, Annual Conference Series, pages 99–108, August 1996.

[57] Hugues Hoppe, Tony DeRose, Tom Duchamp, Mark Halstead, Hubert Jin, John McDonald, Jean Schweitzer, and Werner Stuetzle. Piecewise smooth surface reconstruction. *Proceedings of SIGGRAPH 94*, pages 295–302, July 1994.

[58] Hugues Hoppe, Tony DeRose, Tom Duchamp, John McDonald, and Werner Stuetzle. Surface reconstruction from unorganized points. In *SIGGRAPH '92: Proceedings of the 19th annual conference on Computer graphics and interactive techniques*, pages 71–78, New York, NY, USA, 1992. ACM Press.

[59] Doug L. James and Dinesh K. Pai. BD-Tree: Output-sensitive collision detection for reduced deformable models. *ACM Transactions on Graphics*, 23(3), August 2004.

[60] Aravind Kalaiah and Amitabh Varshney. Differential point rendering. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 139–150, London, UK, 2001. Springer-Verlag.

[61] Aravind Kalaiah and Amitabh Varshney. Modeling and rendering of points with local geometry. *IEEE Transactions on Visualization and Computer Graphics*, 9(1):30–42, 2003.

[62] Leif Kobbelt and Mario Botsch. A Survey of Point-based Techniques in Computer Graphics. *Computers & Graphics*, 28(6):801–814, 2004.

[63] Leif Kobbelt, Swen Campagna, Jens Vorsatz, and Hans-Peter Seidel. Interactive multi-resolution modeling on arbitrary meshes. In *Proceedings of SIGGRAPH 98*, Computer Graphics Proceedings, Annual Conference Series, pages 105–114, July 1998.

[64] Ravikrishna Kolluri. Provably good moving least squares. In *ACM-SIAM Symposium on Discrete Algorithms*, 2005. to appear.

[65] Daniel Kristjansson, Henning Biermann, and Denis Zorin. Approximate boolean operations on free-form solids. In *Proceedings of ACM SIGGRAPH 2001*, Computer Graphics Proceedings, Annual Conference Series, pages 185–194, August 2001.

[66] Adi Levin. Combined subdivision schemes for the design of surfaces satisfying boundary conditions. *Computer Aided Geometric Design*, 16(5):345–354, 1999.

[67] David Levin. The approximation power of moving least-squares. *Math. Comput.*, 67(224):1517–1531, 1998.

[68] David Levin. Mesh-independent surface interpolation. In *Geometric Modeling for Data Visualization*. Springer, 2003.

[69] Marc Levoy. Efficient ray tracing of volume data. *ACM Transactions on Graphics*, 9(3):245–261, 1990.

[70] Marc Levoy, Kari Pulli, Brian Curless, Szymon Rusinkiewicz, David Koller, Lucas Pereira, Matt Ginzton, Sean Anderson, James Davis, Jeremy Ginsberg, Jonathan Shade, and Duane Fulk. The digital michelangelo project: 3d scanning of large statues. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 131–144, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[71] Marc Levoy and Turner Whitted. The use of points as display primitives. Technical report, CS Departement, University of North Carolina at Chapel Hill, 1985.

[72] Erik Lindholm, Mark J. Kligard, and Henry Moreton. A user-programmable vertex engine. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 149–158, New York, NY, USA, 2001. ACM Press.

[73] N. Litke, A. Levin, and P. Schröder. Fitting subdivision surfaces. In *IEEE Visualization 2001*, pages 319–324, October 2001.

[74] William E. Lorensen and Harvey E. Cline. Marching Cubes: A High Resolution 3d Surface Construction Algorithm. In *Proceedings of the 14th annual conference on Computer graphics and interactive techniques*, pages 163–169. ACM Press, 1987.

[75] William R. Mark, R. Steven Glanville, Kurt Akeley, and Mark J. Kilgard. Cg: a system for programming graphics hardware in a c-like language. In *SIGGRAPH '03: ACM SIGGRAPH 2003 Papers*, pages 896–907, New York, NY, USA, 2003. ACM Press.

[76] Matthias Müller, Bruno Heidelberger, Matthias Teschner, and Markus Gross. Meshless deformations based on shape matching. In *SIGGRAPH '05: ACM SIGGRAPH 2005 Papers*, pages 471–478, New York, NY, USA, 2005. ACM Press.

[77] Ahmad H. Nasri and Malcolm A. Sabin. Taxonomy of interpolation constraints on recursive subdivision surfaces. *The Visual Computer*, 18(5/6):382–403, 2002.

[78] Xinlai Ni, Michael Garland, and John C. Hart. Fair morse functions for extracting the topological structure of a surface mesh. *ACM Transactions on Graphics*, 23(3):613–622, August 2004.

[79] NVIDIA. NV_occlusion_query extension, 2002.

[80] Yutaka Ohtake, Alexander G. Belyaev, Marc Alexa, Greg Turk, and Hans-Peter Seidel. Multi-level partition of unity implicits. *ACM Transactions on Graphics*, 22(3):463–470, July 2003.

[81] Yutaka Ohtake, Alexander G. Belyaev, and Hans-Peter Seidel. 3D scattered data approximation with adaptive compactly supported radial basis functions. In *Shape Modeling International 2004*, pages 31–39, Genova, Italy, June 2004.

[82] Mark Pauly, Markus Gross, and Leif P. Kobbelt. Efficient simplification of point-sampled surfaces. In Robert Moorhead, Markus Gross, and Kenneth I. Joy, editors, *Proceedings of the 13th IEEE Visualization 2002 Conference (VIS-02)*, pages 163–170, Piscataway, NJ, October 27– November 1 2002. IEEE Computer Society.

[83] Mark Pauly, Richard Kaiser, Leif Kobbelt, and Markus Gross. Shape modeling with point-sampled geometry. *ACM Transactions on Graphics*, 22(3), 2003.

[84] Mark Pauly, Richard Keiser, and Markus Gross. Multi-scale feature extraction on point-sampled surfaces. *Computer Graphics Forum*, 22(3):281–290, September 2003.

[85] Hanspeter Pfister, Matthias Zwicker, Jeroen van Baar, and Markus Gross. Surfels: surface elements as rendering primitives. In *SIGGRAPH '00: Proceedings of the 27th annual conference on Computer graphics and interactive techniques*, pages 335–342, New York, NY, USA, 2000. ACM Press/Addison-Wesley Publishing Co.

[86] Jarek Rossignac. Structured topological complexes: a feature-based api for non-manifold topologies. In *SMA '97: Proceedings of the Fourth Symposium on Solid Modeling and Applications*, pages 1–9, May 1997.

[87] Szymon Rusinkiewicz and Marc Levoy. QSplat: A multiresolution point rendering system for large meshes. In Kurt Akeley, editor, *Siggraph 2000, Computer Graphics Proceedings*, pages 343–352, New York, NY, USA, 2000. ACM Press / ACM SIGGRAPH / Addison Wesley Longman.

[88] Stephan R. Sain. Multivariate locally adaptive density estimation. *Computational Statistics and Data Analysis*, (39):165–186, 2002.

[89] Gernot Schaufler and Henrik Wann Jensen. Ray tracing point sampled geometry. In *Proceedings of the Eurographics Workshop on Rendering Techniques 2000*, pages 319–328, London, UK, 2000. Springer-Verlag.

[90] Thomas W. Sederberg, David L. Cardon, G. Thomas Finnigan, Nicholas S. North, Jianmin Zheng, and Tom Lyche. T-spline simplification and local refinement. *ACM Transactions on Graphics*, 23(3):276–283, August 2004.

[91] Thomas W. Sederberg, Jianmin Zheng, Almaz Bakenov, and Ahmad Nasri. T-splines and t-nurccs. *ACM Transactions on Graphics*, 22(3):477–484, July 2003.

[92] Chen Shen, James F. O'Brien, and Jonathan R. Shewchuk. Interpolating and approximating implicit surfaces from polygon soup. *ACM Transactions on Graphics*, 23(3):896–904, August 2004.

[93] Yoshihisa Shinagawa, Tosiyasu L. Kunii, and Yannick L. Kergosien. Surface coding based on morse theory. *IEEE Computer Graphics & Applications*, 11(5):66–78, September 1991.

[94] Renben Shu and Alan Liu. A fast ray casting algorithm using adaptive isotriangular subdivision. In *Proceedings of the 2nd conference on Visualization '91*, pages 232–238. IEEE Computer Society Press, 1991.

[95] Marc Stamminger and George Drettakis. Interactive sampling and rendering for complex and procedural geometry. In *Proceedings of the 12th Eurographics Workshop on Rendering Techniques*, pages 151–162, London, UK, 2001. Springer-Verlag.

[96] Richard Szeliski and David Tonnesen. Surface modeling with oriented particle systems. *Computer Graphics*, 26(2):185–194, 1992.

[97] Ingo Wald. *Realtime Ray Tracing and Interactive Global Illumination*. PhD thesis, Computer Graphics Group, Saarland University, 2004. Available at http://www.mpi-sb.mpg.de/~wald/PhD/.

[98] Ingo Wald and Hans-Peter Seidel. Interactive Ray Tracing of Point Based Models. In *Proceedings of 2005 Symposium on Point Based Graphics*, 2005.

[99] Michael Wand, Matthias Fischer, Ingmar Peter, Friedhelm Meyer auf der Heide, and Wolfgang Strasser. The randomized z-buffer algorithm: interactive rendering of highly complex scenes. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 361–370, New York, NY, USA, 2001. ACM Press.

[100] Michael Waschbüsch, Stephan Würmlin, Eduard C. Lamboray, Felix Eberhard, and Markus Gross. Progressive compression of point-sampled models. In *Eurographics/IEEE Symposium on Point-Based Graphics*, pages 95–102. Eurographics, 2004.

[101] Holger Wendland. Piecewise polynomial, positive definite and compactly supported radial functions of minimal degree. In *Advances in Computational Mathematics 4*, pages 389–396, 1995.

[102] Martin Wicke, Denis Steinemann, and Markus Gross. Efficient animation of point-based thin shells. In *Proceedings of Eurographics '05*, pages 667–676, 2005.

[103] Andrew P. Witkin and Paul S. Heckbert. Using particles to sample and control implicit surfaces. *Computer Graphics*, 28(Annual Conference Series):269–277, 1994.

[104] Jianhua Wo and Leif Kobbelt. Optimized sub-sampling of point sets for surface splatting. *Computer Graphics Forum*, 23(3), August 2004.

[105] Cliff Woolley, David Luebke, Benjamin Watson, and Abhinav Dayal. Interruptible rendering. In *SI3D '03: Proceedings of the 2003 symposium on Interactive 3D graphics*, pages 143–151, New York, NY, USA, 2003. ACM Press.

[106] Lexing Ying and D. Zorin. Nonmanifold subdivision. In *IEEE Visualization 2001*, pages 325–331, October 2001.

[107] Lexing Ying and Denis Zorin. A simple manifold-based construction of surfaces of arbitrary smoothness. *ACM Transactions on Graphics*, 23(3):271–275, August 2004.

[108] Denis Zorin, Peter Schröder, and Wim Sweldens. Interactive multiresolution mesh editing. In *Proceedings of SIGGRAPH 97*, Computer Graphics Proceedings, Annual Conference Series, pages 259–268, August 1997.

[109] M. Zwicker, J. Rsnen, M. Botsch, C. Dachsbacher, and M. Pauly. Perspective accurate splatting. In *In Proceedings of Graphics Interface 2004.*, 2004.

[110] Matthias Zwicker. *Continuous reconstruction, rendering, and editing of point-sampled surfaces*. Ph.d. thesis, Institute for Scientific Computing, ETH Zrich, September 2003.

[111] Matthias Zwicker, Mark Pauly, Oliver Knoll, and Markus Gross. Pointshop 3d: an interactive system for point-based surface editing. *ACM Transactions on Graphics*, 21(3):322–329, 2002.

[112] Matthias Zwicker, Hanspeter Pfister, Jeroen van Baar, and Markus Gross. Surface splatting. In *SIGGRAPH '01: Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, pages 371–378, New York, NY, USA, 2001. ACM Press.

Index