EUROGRAPHICS 2004 / M.-P. Cani and M. Slater
(Guest Editors)

Volume 23 (2004), Number 3

Optimized Sub-Sampling of Point Sets for Surface Splatting

Jianhua Wu Leif Kobbelt

Computer Graphics Group, RWTH Aachen, Germany

Abstract

Using surface splats as a rendering primitive has gained increasing attention recently due to its potential for high-performance and high-quality rendering of complex geometric models. However, as with any other rendering primitive, the processing costs are still proportional to the number of primitives that we use to represent a given object. This is why complexity reduction for point-sampled geometry is as important as it is, e.g., for triangle meshes. In this paper we present a new sub-sampling technique for dense point clouds which is specifically adjusted to the particular geometric properties of circular or elliptical surface splats. A global optimization scheme computes an approximately minimal set of splats that covers the entire surface while staying below a globally prescribed maximum error tolerance ε*. Since our algorithm converts pure point sample data into surface splats with normal vectors and spatial extent, it can also be considered as a surface reconstruction technique which generates a hole-free piecewise linear* C^{-1} *continuous approximation of the input data. Here we can exploit the higher flexibility of surface splats compared to triangle meshes. Compared to previous work in this area we are able to obtain significantly lower splat numbers for a given error tolerance.*

1. Introduction

Point-based geometry representations have gained quite some attention recently due to their conceptual simplicity [GPA*03]. Individual point samples can be computed or measured on any kind of surface description and a sufficiently dense set of samples already provides an effective surface approximation that is suitable, e.g., for high-quality rendering [PZBG00, RL00]. The major advantage over "classical" geometry representations is that point sets do not require any connectivity information. This significantly simplifies their processing since no topological consistency conditions have to be satisfied.

In order to guarantee a visually continuous appearance of the displayed objects, the concept of purely point-based representations is usually generalized to splat-based representations [ZPBG01], i.e., infinitesimal points are replaced by ellipses or rectangles, either in object space or in image space. Due to their intuitive geometric interpretation, especially object space techniques, like *surface splatting*, have been used widely in applications ranging from high-quality rendering and level-of-detail handling of complex models to interactive shape design [ZPBG01, ZPKG02, PKKG03].

Although surface splats require the derivation of attributes such as orientation and spatial extent for each splat, the conceptual simplicity of point-based representations is largely preserved. However, in a strict sense, we are dealing with a piecewise linear surface representation just like triangle meshes with the important difference that the linear pieces join in a C^{-1} fashion rather than C^0.

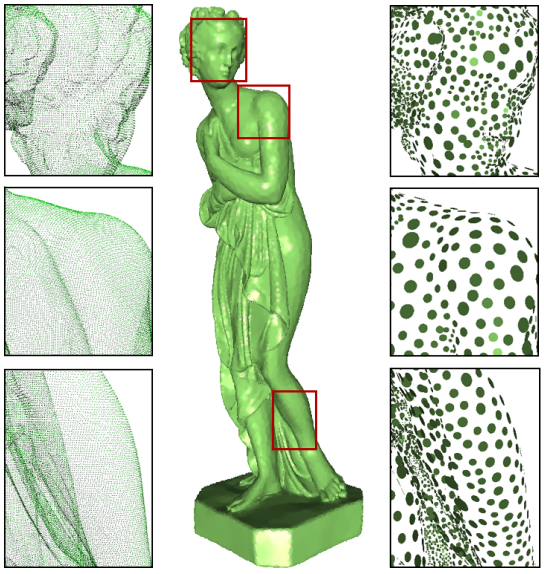

Figure 1: *Optimized sub-sampling of the Iphigenie (left,* 352*K points) using* 30181 *circular splats. The error tolerance is set to* 0.05% *of the bounding box diagonal. The center figure is rendered with EWA-filtered splats and the right zoom-in figures show the sample density and distribution.*

On the other hand, in most approaches the generation of surface splat models still has the flavor of a sampling procedure since the position and normal information is considered to be associated with the *center* of each splat while the spatial extent (radius for circular splats and minor/major axes for elliptical ones) is just needed to bridge the gap between two neighboring samples. Our approach in this paper, in contrast, takes the full geometry of the circular or elliptical object space splats into account when carefully placing samples on a given surface such that the collection of all splats forms a hole-free representation of the input geometry.

Even if surface splats can be processed extremely fast by exploiting the programmable features of current graphics hardware [DVS03, BK03], we would like to minimize the number of splats that are needed to represent a given object since the processing time is nevertheless proportional to the number of geometric primitives. This is why geometry simplification techniques have been developed for splat-based representations, most of them being derived from well-established mesh decimation or hierarchical clustering schemes [GGK02].

In this paper, we present a sub-sampling algorithm which is especially designed for splat-based representations and which makes use of their flexibility. Since no topological consistency conditions between neighboring splats have to be observed, we can apply an unconstrained global optimization technique. By this we can considerably reduce the splat count for a given model while still observing a prescribed error tolerance. In contrast to previous approaches we are not focussing merely on the relation between splat *centers* but rather consider each splat as a *linear surface segment* with finite spatial extent.

1.1. Related Work

Using points as rendering primitives was first introduced by Levoy et al. [LW85] and later followed and improved by [GD98, PZBG00, RL00, ZPBG01, WFP*01, BWK02] aiming at efficient and high-quality rendering of complex models. Most recently, various applications dealing with point-based geometry have been intensively discussed including shape acquisition and authoring [MPN*02], high quality rendering [ZPBG01, RPZ02, BSK04], (re-) sampling and simplification [SD01, PGK02], as well as as shape editing [ZPKG02, PKKG03]. See [GPA*03] for a complete survey.

Pure point representations are discrete samplings of the input geometry and hence proper reconstruction filters have to be applied in order to enable hole-free rendering. Approaches applied in image space simply render "large" points as squares or circles [GD98, SD01, KV03b]. Object space approaches use (disk-shaped or elliptical) surface splats [ZPBG01, RPZ02, BWK02, Paj03], quadratic [KV03a], or even higher-order [ABF*03, OBA*03] patches to reconstruct the surface geometry. In this paper we are using circular or elliptical object space splats, since they seem to provide the best quality/performance trade-off when implemented on programmable graphics hardware [RPZ02, BK03, ZRB*04].

Many existing sub-sampling methods focus only on the relation between splat centers and hence require extra effort to estimate the actual spatial extent of splats to fill the gap between samples. Linsen [Lin01] and Alexa et al. [ABF*03] adopted greedy schemes to iteratively remove samples from the input point cloud. The greedy nature of these algorithms cannot guarantee a globally uniform point distribution. Moenning et al. [MD03] use fast marching farthest point sampling for point set simplification. While it is simple and efficient, reliable error control is not supplied and seems not trivial to embed. In [PGK02], Pauly et al. adapted various mesh simplification techniques to simplify point-sampled geometry which, however, cannot take an a-priori approximation error tolerance into account. Their pure greedy simplification produces results with small a-posteriori error but also with non-uniform sampling density. This is why they post-optimize the result with a particle simulation scheme which, alas, increases the approximation error. Our scheme uses a relaxation scheme that takes approximation error into account and uses the complete anisotropic geometry of the elliptical splats – not just their centers. In this sense it differs significantly from previous isotropic sampling approaches like [PGK02] or [SAG03].

Other approaches [RL00, BWK02, Paj03, KV03b] to splat simplification are based on hierarchical clustering schemes. The input points are rearranged in a hierarchical spatial partitioning data structure and splats are created for every node by analyzing the local surface properties. This technique is simple and fast but since there is no optimization strategy involved, the results are usually overly conservative and tend to contain lots of redundant splats.

2. Overview

Let a surface S be given by a set of sample points $P = \{\mathbf{p}_i\}$ which are sufficiently dense in the sense that they form an r-sample of S for some value $r < 1$ [ABK98]. If r is chosen small enough then there exists a constant k such that fitting a least squares plane to any point $\mathbf{p}_i$ and its k nearest neighbors yields a reliable estimate of the surface normal direction at $\mathbf{p}_i$. A consistent normal orientation can be propagated via a minimum spanning tree for the point set P [HDD*92]. We denote by $N_k(\mathbf{p}_i)$ the set of k nearest neighbors to $\mathbf{p}_i$ measured by Euclidian distance and the graph $\mathcal{N} = (P, E)$ represents this non-symmetric *neighborhood relation* where the edge (i, j) belongs to E iff $\mathbf{p}_j \in N_k(\mathbf{p}_i)$. The actual distance $d_i = \|\mathbf{p}_i - \mathbf{p}_k\|$ to the k-th neighbor can be used as an estimate for the local sampling density and we associate a *surface area element* $\omega_i = \pi d_i^2$ with each sample point $\mathbf{p}_i$. The graph structure of $\mathcal{N}$ can be computed most efficiently by using a hierarchical binary space partition [Sam94].

Our goal is to find a minimum set of surface splats $T = \{t_j\}$ that approximates P with an error below some prescribed tolerance ε. Depending on the application, the user can choose if *circular* or *elliptical* splats should be used. Elliptical splats can adapt to the local curvature of the surface usually leading to sparser sets T but they need more storage space per splat. A circular splat t_j is given by its center $\mathbf{c}_j$, its normal vector $\mathbf{n}_j$, and its radius r_j. For elliptical splats we replace the radius r_j by two additional vectors $\mathbf{u}_j$ and $\mathbf{v}_j$ to define the major and minor axes.

For every input sample $\mathbf{p}_i$ we measure its distance to T by orthogonal projection onto the splats t_j, i.e.,

$$\text{dist}(\mathbf{p}_i, T) \;=\; \text{dist}(\mathbf{p}_i, t_j) \;=\; |\mathbf{n}_j^T\,(\mathbf{p}_i - \mathbf{c}_j)|$$

if

$$\Big\| (\mathbf{p}_i - \mathbf{c}_j) - \mathbf{n}_j^T\,(\mathbf{p}_i - \mathbf{c}_j)\,\mathbf{n}_j \Big\|^2 \;\le\; {r_j}^2 \qquad (1)$$

for circular splats or

$$\Big(\mathbf{u}_j^T\,(\mathbf{p}_i - \mathbf{c}_j)\Big)^2 \;+\; \Big(\mathbf{v}_j^T\,(\mathbf{p}_i - \mathbf{c}_j)\Big)^2 \;\le\; 1 \qquad (2)$$

for elliptical splats. If $\mathbf{p}_i$ projects into the interior of several splats, we choose the minimum distance. If (1) or (2) do not hold for any t_j we set $\text{dist}(\mathbf{p}_i, T) = \infty$.

For a given set of splats T and an error tolerance ε, conditions (1) and (2) imply a *coverage relation* $\mathcal{C}_\varepsilon \subset P \times T$ which includes all pairs $(\mathbf{p}_i, t_j)$ for which (1) or (2) holds and $\text{dist}(\mathbf{p}_i, t_j) \le \varepsilon$. We define the *surface patch* $Q_j = \mathcal{C}_\varepsilon(*, t_j)$ corresponding to a splat t_j as the set of all samples $\mathbf{p}_i$ for which the relation $(\mathbf{p}_i, t_j) \in \mathcal{C}_\varepsilon$ holds. The *area* of the patch Q_j is given by

$$\Omega_j \;:=\; \sum_{\mathbf{p}_i \in Q_j} \omega_i .$$

The optimized sub-sampling task can now be formulated as the *minimum dominating set* problem [CLRS01] for the 2-colorable graph $(P \cup T, \mathcal{C}_\varepsilon)$ whose connectivity is defined by the coverage relation $\mathcal{C}_\varepsilon$. Since the dominating set problem is known to be NP-hard, we have to find an approximate optimization algorithm.

Our algorithm proceeds in three steps which we explain in the following subsections. First, a maximum splat t_i is computed for each input sample $\mathbf{p}_i$ whose size is limited by the prescribed error tolerance ε. These splats are centered at $\mathbf{p}_i$ in the sense that $\mathbf{p}_i$ projects to the center $\mathbf{c}_i$ but we do not require $\mathbf{p}_i = \mathbf{c}_i$. This relaxed condition turns out to be crucial for the effective reduction of the splat count. Notice that for extremely complex input data P we can apply simple predecimation [PGK02] before generating the initial splats to speed up the computation. This does not have significant impact on the final result.

From the initial set of splats, we select a subset which savely covers the whole surface. This is done by a greedy procedure where the selection criterion guarantees that neighboring splats have sufficient overlap to provide a hole-free approximation of the input surface S. Since the error tolerance ε has been taken into account in the splat generation step, any selection of splats t_j such that the union of their corresponding patches Q_j completely covers P automatically satisfies the approximation tolerance.

In the third step, the greedy solution is further improved by a global relaxation procedure. The idea is to iteratively replace subsets of splats by new sets that have fewer elements or at least a better splat distribution. Notice that for circular splats, *isotropic* distribution is optimal while for elliptical splats *anisotropic* distribution according to minimum and maximum curvature is optimal. Since our algorithm takes minimum and maximum curvature into account when creating elliptical splats and since the relaxation is controlled by mutual overlap of the splats, we can guarantee to always produce a near-optimal splat distribution.

3. Initial Splat Generation

Initially we generate a circular splat $t_i = (\mathbf{c}_i, \mathbf{n}_i, r_i)$ for each sample $\mathbf{p}_i$ in the (possibly pre-decimated) input set P. Optionally this initial splat can then be extended into an elliptical splat $t_i = (\mathbf{c}_i, \mathbf{n}_i, \mathbf{u}_i, \mathbf{v}_i)$ in the next step (cf. Sec. 3.1).

Starting with a seed point $\mathbf{p}_i$ we first estimate the local normal direction $\mathbf{n}_i$ by fitting a least squares plane to $\mathbf{p}_i$ and its k nearest neighbors [Jol86]. Then we grow the splat by adding neighboring sample points in the order of their projected distances (1) to $\mathbf{p}_i$. For each new point $\mathbf{p}_j$ we compute the signed distance

$$h_j \;=\; \mathbf{n}_i^T\,(\mathbf{p}_j - \mathbf{p}_i)$$

and the growing stops as soon as the interval $[h_{\min}, h_{\max}]$ becomes larger than 2ε. The center of the splat is then set to

$$\mathbf{c}_i \;=\; \mathbf{p}_i + \frac{h_{\min} + h_{\max}}{2}\,\mathbf{n}_i$$

and the radius is set to

$$r_i \;=\; \Big\| (\mathbf{p}_j - \mathbf{c}_i) - \mathbf{n}_i^T\,(\mathbf{p}_j - \mathbf{c}_i)\,\mathbf{n}_i \Big\|$$

where $\mathbf{p}_j$ has the largest projected distance (1) before the prescribed error tolerance is violated (cf. Fig. 2).

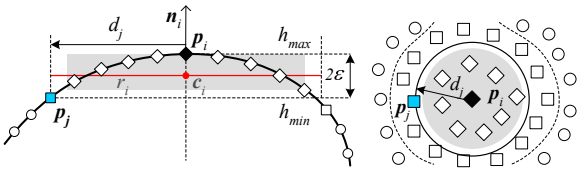

Figure 2: *Growing a splat t_i initially centered at $\mathbf{p}_i$. Symbols ◇, □ and ○ stand for* conquered, front *and* un-covered *samples respectively. The left figure shows a view in tangent direction and the right figure is viewed in normal direction.*

The splat growing procedure can be implemented quite efficiently by breadth first traversal of the neighborhood graph $\mathcal{N}$ in a "fast marching" manner. Since we do not update the normal direction $\mathbf{n}_i$ while the splat is growing, there is no need to update the ordering of the neighbor samples or to recompute the signed distance interval $[h_{\min}, h_{\max}]$. Notice that re-fitting a least squares plane to the conquered sample points after the splat growing reduces the $\|\cdot\|_2$-error but it might actually increase the $\|\cdot\|_\infty$-error and hence could violate the prescribed error tolerance ε.

3.1. Elliptical Splats

After a maximum *circular* splat t_i seeded at the point sample $\mathbf{p}_i$ is generated, it is optionally possible to continue the growing procedure into the minimum curvature direction to obtain an *elliptical* splat which better adapts to the local anisotropic surface curvature while still keeping the error tolerance (cf. Fig. 4). In addition to the center $\mathbf{c}_i$ and normal $\mathbf{n}_i$ we need two non-normalized tangent vectors $\mathbf{u}_i$ and $\mathbf{v}_i$ to define the major and minor axis of the elliptical splat according to (2).

In order to find a robust estimate of the (normalized) principal directions $\gamma_{\min}$ and $\gamma_{\max}$, we use the shape operator of [CSM03]. Since this operator is defined for triangle meshes, we triangulate the surface patch Q_i by projecting the corresponding sample points into the supporting plane of the splat and computing a 2D Delaunay triangulation.

With r_i being the radius of the initial circular splat t_i, we define the minor axis of the elliptical splat by $\mathbf{v}_i = \gamma_{\max}/r_i$.

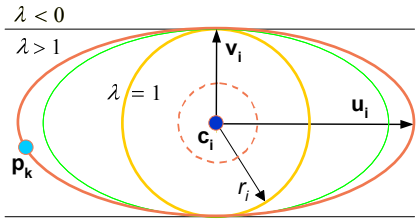

Figure 3: *A circular splat with center* $\mathbf{c}_i$ *and radius* r_i *is extended into an elliptical splat with semi-axes* $\mathbf{u}_i$ *and* $\mathbf{v}_i$.

Let $\sqrt{\lambda}$ be the (unknown) aspect ratio of the resulting elliptical splat then we can order the neighboring samples $\mathbf{p}_j$ by increasing aspect ratio, i.e., by

$$\lambda_j := \frac{(\gamma_{\min}^T(\mathbf{p}_j - \mathbf{c}_i))^2}{r_i^2 - (\gamma_{\max}^T(\mathbf{p}_j - \mathbf{c}_i))^2}$$

where we do only consider samples for which the denominator is positive. We continue the splat growing procedure described in the last section but this time we add the neighboring samples $\mathbf{p}_j$ ordered by increasing λ_j. Again, the growing procedure stops when the interval $[h_{\min}, h_{\max}]$ becomes larger than 2ε and if $\mathbf{p}_j$ is the last added neighbor before the error tolerance is violated, the major axis of the elliptical splat becomes

$$\mathbf{u}_i = \frac{1}{r_i\sqrt{\lambda_j}}\gamma_{\min}.$$

Figure 4: *A torus approximated with error tolerance* 0.2% *by* 510 elliptical *splats (left and center) and by* 734 circular *splats (right).*

3.2. Hole-free Approximation

In the selection and optimization steps we will choose a subset of *active* splats from the initially generated set of *candidate* splats. During the process, we have to verify if the current subset of splats actually covers the whole surface S. This is difficult in general since we only have a discrete set of sample points. A much simpler condition is to check if each sample point $\mathbf{p}_i \in P$ is covered by at least one splat. Testing the latter simply requires to iterate over all active splats, tag all covered samples based on the relation $\mathcal{C}_\varepsilon$, and eventually check if there are un-tagged sample points left.

However, as shown in Fig. 5, covering all sample points $\mathbf{p}_i$ by active splats is not sufficient to guarantee a hole-free approximation since holes can appear inbetween the sample points. Hence we have to modify the definition of the patch which is savely covered by a splat t_i.

After the growing procedure for a splat t_i stops, the set of conquered sample points defines a local surface patch Q_i which projects into the interior of the splat defined by the splat center $\mathbf{c}_i$ and its radius r_i or by its minor and major axes $\mathbf{u}_i$ and $\mathbf{v}_i$ respectively. As a by-product of the 2D Delaunay triangulation within the supporting plane of the splat (cf. Sec. 3.1) we can compute the *convex hull* of the projected sample points which also lies completely in the interior of the splat due to convexity. Now we define the *interior* patch $\bar{Q}_i$ as the subset of Q_i which excludes all the boundary vertices, i.e., all samples that lie on the convex hull.

If we restrict our coverage relation $\mathcal{C}_\varepsilon$ to the interior coverage $\bar{\mathcal{C}}_\varepsilon$ then the test if all sample points $\mathbf{p}_i$ are covered by the current selection of active splats is a conservative estimate for the condition that the whole surface S is approximated in a hole-free manner. Here we assume that the initial sampling is sufficiently dense. Severely undersampled regions should be considered as intentional holes in the surface S.

Since the hole-detection as well as all the following operations are only based on the coverage relation $\bar{\mathcal{C}}_\varepsilon$, we can

simply consider splats as sets of samples and we do not have to distinguish between circular and elliptical splats in the rest of the algorithm.

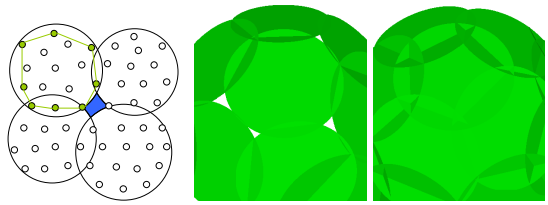

Figure 5: *The left figure shows that all sample points are covered but the small blue region between the samples remains as a hole. The middle and right figures show splat-based models with and without holes for the same error threshold. The green points on the left indicate the boundary points $Q_j \setminus \bar{Q}_j$ of that splat.*

4. Greedy Selection

From the set of candidate splats T we select an *active* subset T' that covers the whole surface S, i.e., which satisfies

$$P = \bigcup_{t_i \in T'} \bar{Q}_i.$$

according to our above discussion of hole-free approximation. Since finding the minimum subset T' is NP-hard [CLRS01] we have to find an approximate solution. As with most geometrical optimization problems, we find that a greedy selection strategy provides a suitable solution.

Similar to [DDSD03] we rank the splats according to their incremental surface area contribution. For every candidate splat $t_j \in T$ we sum the area elements ω_i associated with those sample points $\mathbf{p}_i \in \bar{Q}_j$ that are not already covered by previously selected splats. Then in each step we select the splat which adds the maximum surface area to the active set and update the area contribution of the remaining candidates.

To implement the area contribution update, we explicitly store the relation $\bar{\mathcal{C}}_\varepsilon$ two times. Once organized by patches $\bar{Q}_j$ and once organized by sample points $\mathbf{p}_i$. When a splat t_j is selected, we iterate over all samples $\mathbf{p}_i \in \bar{Q}_j$ and subtract the area ω_i from each patch that overlaps $\mathbf{p}_i$. By this, the computation costs for the update only depend on the size of the splats and the degree of overlap but not on the total number of splats in T.

5. Global Relaxation

Even if it can be proven theoretically that the greedy selection produces a splat set T' which is close to the optimum [Hoc95, Hro01], there are still situations where some splats are redundant (cf. Fig. 6). Moreover, the local decision about which splat to select usually causes a disturbing non-uniformity of the splat distribution (cf. Fig. 7). This has to be overcome by some global optimization scheme where we can exploit the fact that splat-based surface representations do not have to respect any consistency requirements. Hence we can add and remove splats in arbitrary order as long as we preserve a full hole-free coverage of the input samples.

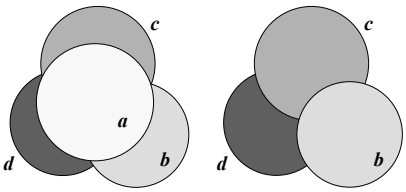

Figure 6: *Greedy selection produces redundant splats. Splats a, b, c and d are selected consecutively based on their incremental area contribution (left). Eventually a's area is fully covered by the remaining splats and could be removed (right). A greedy scheme cannot detect such situations since no backtracking beyond earlier decisions is possible.*

Our global relaxation procedure mimics the behavior of repulsing particles on the surface [Tur91]. Since we do not have a continuous surface representation we have to use the neighborhood structure which is encoded in the connectivity of the neighborhood graph $\mathcal{N}$. This neighborhood relation between sample points can be transferred to splats since each splat t_i is uniquely associated with the sample point $\mathbf{p}_i$ which has been used as a seed when growing t_i. In this sense we misuse the notation $N_k(t_i)$ to refer to those splats t_j whose seed points $\mathbf{p}_j$ are in $N_k(\mathbf{p}_i)$.

The local movement of a splat-particle t_i is achieved by removing t_i from the active set and replacing it with another splat $t_j \in N_k(t_i)$. The choice of the new splat t_j is controlled by a local relaxation force. In contrast to previous approaches [Tur91, PGK02], we derive this force by taking the complete splat geometry into account and do not simply consider the relation between splat centers.

We use two operations to improve the splat distribution and to remove redundant splats. In the first operation we iterate over all active splats and check if there is another splat in the vicinity that has less overlap with its neighbors. In the second operation, we check for each splat if it can be removed, i.e., if the hole resulting from its removal can be re-covered by locally "moving" the neighboring splats. The overall optimization procedure alternates between relaxation phases and redundant splat removal phases. Since the convergence in our experiments turned out to be quite fast, we are usually applying each phase just once. More iterations can still improve the result slightly but the effect per computation time is decreasing.

We introduce the *kernel* of a splat $t_i \in T'$ as the set

$$K_i := \bar{Q}_i \setminus \bigcup_{t_j \in T', j \neq i} \bar{Q}_j \qquad (3)$$

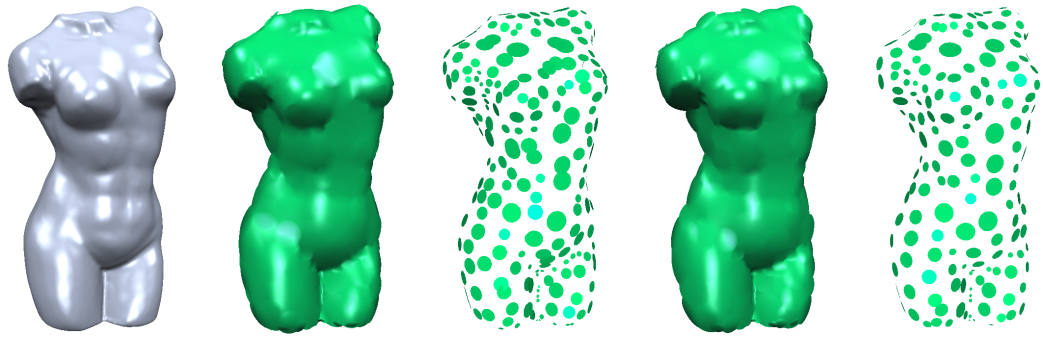

Figure 7: *A Female-torso model (left, 171K sample points) is approximated by 422 circular splats after greedy selection (middle two). Global relaxation further reduces the number of splats to 333 (right two). The figures show both, EWA-filtered splats for approximation quality and smaller splats for distribution quality. The error tolerance is $\varepsilon = 0.47\%$ of the bounding box diagonal. Notice the improved splat distribution after the global relaxation step.*

which consists of all sample points that are only covered by t_i and not by any other *active* splat t_j. The kernel is critical to preserve the hole-free property of the active splat set T' during global relaxation: if we remove t_i we have to make sure that K_i is covered again after the relaxation.

Let $V_j = \bar{C}_\varepsilon(\mathbf{p}_j, *) \cap T'$ be the set of currently *active* splats that cover a given sample point $\mathbf{p}_j$. Then the kernel K_i for a splat t_i consists of all points $\mathbf{p}_j \in \bar{Q}_i$ such that V_j contains just one element $\{t_i\}$. In practice it turns out that it is more efficient to pre-compute the kernels K_i once in the beginning and then to update them after each relaxation step. The same holds for the coverage sets V_j.

5.1. Relaxation

With this operation we are improving the regularity of the splat distribution. Since local non-uniformity in the sparse set T' is indicated by large overlap of neighboring splats, we iterate over all active splats and try to replace each splat $t_i \in T'$ by another candidate splat $t_j \in T \setminus T'$ such that the hole-free property is preserved.

Using the kernel notation for splats, we find that the set

$$U_i \;:=\; \left\{ t_j \in N_k(t_i) \,|\, K_i \subset \bar{Q}_j \right\} \quad \subset \quad T \setminus T' \qquad (4)$$

defines the set of choices for hole-free relaxation. Notice that U_i cannot contain an active splat $t_j \in T'$ due to the definition (3) of the kernel.

The overlap area between two splats is measured by summing the surface area elements of all samples that are covered by both splats

$$\text{overlap}(t_i, t_j) \;:=\; \sum_{\mathbf{p}_l \in \bar{Q}_i \cap \bar{Q}_j} \omega_l .$$

Hence in each step, the relaxation replaces a splat $t_i \in T'$ by another splat $t_j \in U_i$ which minimizes the *maximum* overlap to any other active splat. Notice that it can turn out that t_i already is a local optimum. In this case we simply keep t_i active.

The relaxation loop over all active splats is performed several times until convergence (no more splat replacements take place) or until a maximum number of iterations is reached (10 to 15 in all our experiments).

To obtain the set U_i we check for each $t_j \in N_k(t_i)$ if it is not active and if $K_i \subset \bar{Q}_j$. Then for each element $t_j \in U_i$, we find its maximum overlap to any other active splat t_l from the set

$$W_j \;:=\; \bigcup_{\mathbf{p}_m \in \bar{Q}_j} V_m ,$$

where $\text{overlap}(t_j, t_l)$ is computed by accumulating the area elements ω_m for all sample points $\mathbf{p}_m \in \bar{Q}_j$ whose coverage set V_m contains t_l.

By using hash-tables to associate splats t_i with their indices i, the complexity of these local operations depends only on the number of samples per splat and not on the total number of splats.

Since the relaxation force is defined in terms of mutual overlap area, it turns out that the density of splats depends on their sizes. Hence, for circular splats, the relaxation generates a uniform and isotropic splat distribution and for elliptical splats the distribution is anisotropic according to the orientation and aspect ratio of the splats (cf. Fig 4).

5.2. Removing Redundant Splats

If the kernel K_i of an active splat t_i is empty then this splat can be removed without creating a hole in the splat-based surface approximation. If the kernel is not empty we can still try to replace neighboring splats in a relaxation procedure such that the hole K_i which is caused by removing t_i is covered by those splats. Here we can exploit the full flexibility of splat-based representations since no topological restrictions have to be taken into account.

We iterate over all active splats in T' and in each step we tentatively remove one splat t_i. In order to fill the resulting hole K_i, we apply a relaxation operator to all currently active splats $t_j \in W_i$ that have a non-empty overlap with t_i. The restriction to this set W_i is not necessary but it accelerates the implementation since we can use the quick overlap computation procedure sketched in the last section. If after the relaxation the hole K_i could not be closed, we backup to the splat configuration before the tentative removal of t_i and continue with the next splat.

The relaxation force in this case favours splats that cover a maximum portion of the hole $K = K_i$. Just like in (4), we define for each splat $t_j \in W_i$ the set of possible choices U_j. Among those choices, we pick the splat t_l that maximizes overlap(t_l, K). After each relaxation step we have to update the remaining hole $K \leftarrow K \setminus \bar{Q}_l$ to account for the local change. Notice that the definition of U_j automatically guarantees that no other holes can appear during relaxation.

6. Results and Comparisons

We have applied our sub-sampling technique to a wide range of different unstructured point-set models obtained from laser scanning to demonstrate the quality and performance. In all experiments we set the k-neighborhood to 10 such that the only user-defined input parameter is the error tolerance ε. The optimized splat sub-sampling so works fully automatic.

In Table 1 we report the overall running times of the sub-sampling algorithm measured on a Pentium4 2.8GHz CPU, 2GB memory system. Note that the additional time used for elliptical splats compared to the circular ones mainly results from computing the local surface curvature tensors. Due to the exact $\|\cdot\|_\infty$ error control in the splat generation phase and the global optimization in the relaxation phase, the running times are higher than, e.g., [PGK02] where only approximate error measures and a pure greedy approach is used. Nevertheless the timings are acceptable as a pre-processing step. By applying another round of relaxation and redundant splat removal we can usually reduce the output complexity by another $1-2\%$ but this further increases the computation time.

Table 2 shows the timings and splat counts for the Igea model (cf. Fig. 13) with different error tolerances ε. The timings are measured separately for the three phases of the algorithm. As one expects, the computing time for the splat generation increases for larger error thresholds. This comes from the fact that while the *number* of candidate splats depends on the number of input samples, the *size* of these splats depends on the tolerance. Conversely, the time for the global relaxation increases with smaller error threshold since it depends on the *number* of active splats and not so much on their *size*. Adding all timings together, it turns out that the overall computing costs have a global minimum for some intermediate error tolerance (cf. Fig 8). If ε becomes larger, the splat generation dominates the computing costs and if ε becomes smaller, the relaxation will then dominate.

model	#points	ε(%)	type	time(s)	#splats
Charlemagne	598386	0.1	C	935	17445
Buddha	546730	0.1	C	592	13249
			E	1048	9405
Dragon	437645	0.2	C	354	4450
Iphigenie	351750	0.05	C	663	30181
			E	1207	20714
Max	199169	0.2	C	189	1271
Female-torso	171094	0.2	C	236	861
			E	394	698
Igea	134345	0.2	C	116	1621
Horse	50697	0.3	C	27	1138
			E	42	898
Bunny	34835	0.3	C	14	1371
Torus	30000	0.2	C	12	734
			E	19	510

Table 1: *Running-times of the algorithm on various models. The 'type' stands for the splat shape: (C)ircular or (E)lliptical.*

ε (%)	times (s)			# splats	
	create	greedy	relax	greedy	relax
0.8	352	73	26	242	218
0.4	149	38	24	658	580
0.2	72	12	31	1864	1621
0.1	34	7	46	4856	4285
0.05	20	4	85	11813	10737
0.02	11	3	192	31154	29847

Table 2: *Timings and (circular) splat numbers after the greedy selection and global relaxation for the Igea model with decreasing error tolerance* ε.

The rate by which the number of splats increases for decreasing error threshold ε in Table 2 indicates that our piecewise linear C^{-1} continuous surface reconstruction scheme in fact has linear precision and hence quadratic approximation order. This justifies that splat-based geometry representations and triangle meshes are equally powerful in approximating freeform geometry. Figures 1, 7, 9, 12 and 13 demonstrate the visual quality of the splat-based representations. Whether the models are globally smooth or have many small features, our sub-sampling algorithm always produces splat-based approximations that have high visual fidelity, uniform sample density, no holes, and are guaranteed to stay within the prescribed error tolerance.

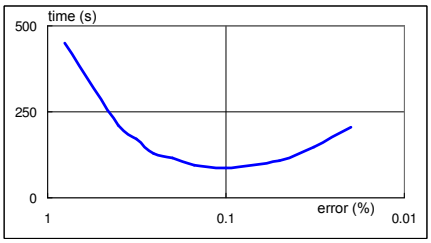

Figure 8: *Overall computing costs vs. error tolerance* ε *for the Igea model (cf. Table 2).*

6.1. Elliptical vs. Circular Splats

We first compare the approximation power of two types of splats. Since a torus has a simple geometry with clear curvature changes, we use it as a benchmark object in Fig. 4. Our results on other less synthetic objects are similar but sometimes overlayed by effects caused by fine geometric detail.

We found that for a given error tolerance ε, the required number of elliptical splats is usually about 30% less than the number of circular splats (cf. Fig. 9). Besides the additional computation time during the splat generation phase, elliptical splats also require slightly more per-pixel computation during rendering. However, as demonstrated in Fig. 13, the improved visual quality when using Phong shading [BSK04] definitely justifies the use of elliptical splats.

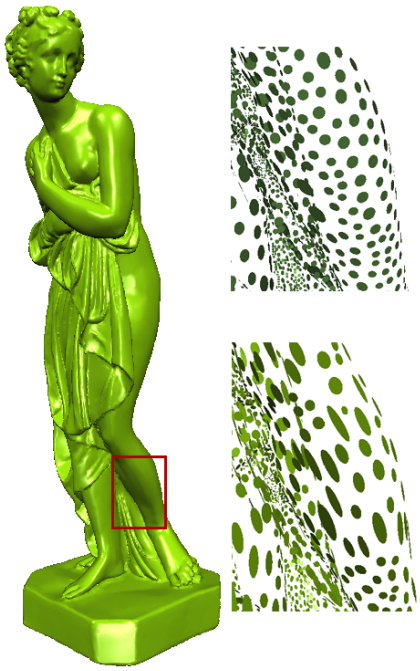

Figure 9: *The Iphigenie model approximated by* 20714 *elliptical splats with error tolerance* $\varepsilon = 0.05\%$ *(left). The right figures show its zoomed-in views of* 30181 *circular splats (top) and fewer elliptical ones (bottom).*

6.2. Comparing to Point Cloud Simplification

We compare the results of our algorithm to the greedy iterative point cloud simplification scheme proposed in [PGK02] using circular splats as [PGK02] only considers circular ones. By using the technique of [Paj03], we derive splat radii for [PGK02]'s output and then compute the $\|\cdot\|_\infty$ approximation error from the input point cloud to the output splats.

Fig. 10 shows the results of the greedy point cloud simplification and of our algorithm on the Torus model. With about three times the computation time, our algorithm produces a splat-based model with the same number of splats but with a significantly smaller approximation error (which comes from the superior splat distribution). The more uniform distribution of the splats in our case also equalizes the sizes of the splats and hence improves the visual quality. If we allow for an equally high error tolerance then our global relaxation scheme can reduce the splat count by one third. This indicates how far away from the optimal solution the greedy solution is.

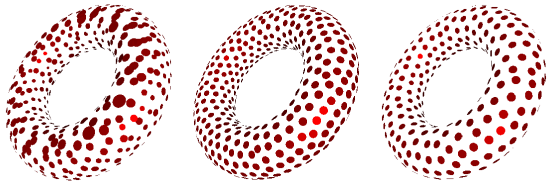

Figure 10: *Same number of* 734 *circular splats generated by point cloud simplification [PGK02] (left) and by our algorithm (center) with respective running time* 4.5 *sec and* 12 *sec and approximation error* 0.29% *and* 0.2%. *The right figure shows our sub-sampling result at 0.29% error using only 493 splats created in 16 seconds.*

6.3. Comparing to Mesh Simplification

We also compare our results to the well-established greedy mesh simplification algorithm [GH97]. Triangle meshes have the advantage over circular splats that they can adjust anisotropically to the local minimum and maximum curvature while disk-shaped splats can only adjust isotropically to the maximum curvature. On the other hand, splat representations are more flexible since they do not have to satisfy C^0 continuity conditions between the linear pieces. Our experiments indicate that the two advantages somewhat balance each other but the flexibility of splat-based representation (even circular ones) almost always leads to better visual quality as well as to lower $\|\cdot\|_\infty$ approximation error when using the same number of primitives (cf. Fig. 11).

By comparing Fig. 11 with Fig. 4 we see that elliptical splats yield the same visual quality while using about 30% fewer primitives.

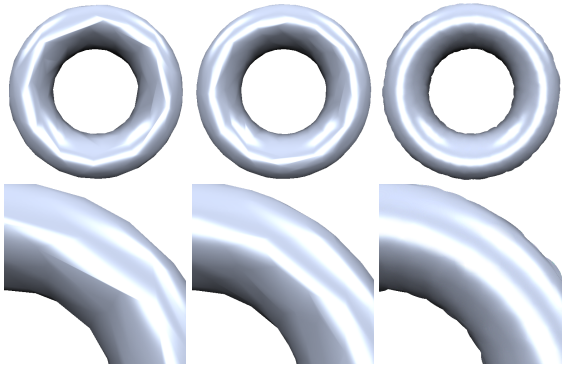

Figure 11: *Torus simplified to meshes with* 367 *vertices and* 734 *faces (left),* 734 *vertices and* 1468 *faces (center) by the greedy algorithm [GH97] and sub-sampled with* 734 *circular splats by our algorithm (right). The approximation errors are 0.66%, 0.38% and 0.2% respectively. The models are rendered with Phong-shaded triangles and filtered Phong splats [BSK04].*

7. Conclusions

In this paper we presented a sub-sampling scheme that converts a dense set of point samples into a sparse set of circular of elliptical object-space splats that provide a hole-free approximation of the original data up to a prescribed error tolerance ε. The scheme achieves comparably low splat counts and uniform splat distribution by applying a global optimization scheme. The scheme is not as fast as pure greedy approaches but the increased computation costs are paid off by a considerably improved output quality.

In the future we are planning to improve the algorithm into various directions: The initial splat candidate generation could be accelerated significantly by quantizing the least squares normals and then saving computation time by generating splats with similar orientation simultaneously. Also, the splat count could be further reduced by using a less conservative estimate for the hole-free property. This would only marginally change our algorithm since all forces are only defined by set-operations on the coverage relation $\bar{\mathcal{C}}_\varepsilon$. Another issue is the proper handling of sharp features to exploit the full functionality of recent high quality splat renderers such as [ZRB*04, BSK04].

References

[ABF*03] ALEXA M., BEHR J., FLEISHMAN S., COHEN-OR D., LEVIN D., SILVA C. T.: Computing and rendering point set surfaces. *IEEE Transaction on Visualization and Computer Graphics 9*, 1 (2003), 3–15.

[ABK98] AMENTA N., BERN M., KAMVYSSELIS M.: A new voronoi-based surface reconstruction algorithm. In *Proceedings SIGGRAPH 1998* (1998), pp. 415–421.

[BK03] BOTSCH M., KOBBELT L.: High-quality point-based rendering on modern GPUs. In *Proceedings of Pacific Graphics 2003* (2003), pp. 335–343.

[BSK04] BOTSCH M., SPERNAT M., KOBBELT L.: Phong splatting. In *Preprint of Submission* (2004).

[BWK02] BOTSCH M., WIRATANAYA A., KOBBELT L.: Efficient high quality rendering of point sampled geometry. In *Eurographics Workshop on Rendering* (2002).

[CLRS01] CORMEN T. H., LEISERSON C. E., RIVEST R. L., STEIN C.: *Introduction to Algorithms*. MIT Press, 2001.

[CSM03] COHEN-STEINER D., MORVAN J.-M.: Restricted delaunay triangulations and normal cycle. In *19th Annual ACM Symposium on Computational Geometry* (2003), pp. 312–321.

[DDSD03] DÉCORET X., DURAND F., SILLION F. X., DORSEY J.: Billboard clouds for extreme model simplification. *ACM Transactions on Graphics 22*, 3 (SIGGRAPH 2003), 689–696.

[DVS03] DACHSBACHER C., VOGELGSANG C., STAMMINGER M.: Sequential point trees. *ACM Transactions on Graphics 22*, 3 (SIGGRAPH 2003), 657–662.

[GD98] GROSSMAN J. P., DALLY W. J.: Point sample rendering. In *Eurographics Rendering Workshop 1998* (1998), pp. 181–192.

[GGK02] GOTSMAN C., GUMHOLD S., KOBBELT L.: Simplification and compression of 3d-meshes. In *Tutorials on Multiresolution in Geometric Modeling* (2002), Springer.

[GH97] GARLAND M., HECKBERT P. S.: Surface simplification using quadric error metrics. In *Proceedings ACM SIGGRAPH 1997* (1997), pp. 209–216.

[GPA*03] GROSS M., PFISTER H., ALEXA M., DACHSBACHER C., PAULY M., BAAR J. V., STAMMINGER M., ZWICKER M.: Point-based computer graphics. In *EUROGRAPHICS 2003 Tutorial T1* (2003).

[HDD*92] HOPPE H., DEROSE T., DUCHAMP T., MCDONALD J., STUETZLER W.: Surface reconstruction from unorganized points. *Computer Graphics (SIGGRAPH 1992) 26*, 2 (1992), 71–78.

[Hoc95] HOCHBAUM D. S.: *Approximation Algorithms for NP-Hard Problems*. PWS Publishing Company, 1995.

[Hro01] HROMKOVIC J.: *Algorithmics for Hard Problems*. Springer, 2001.

[Jol86] JOLLIFFE I.: *Principle Component Analysis*. Spring-Verlag, 1986.

[KV03a] KALAIAH A., VARSHNEY A.: Modeling and rendering of points with local geometry. *IEEE Transaction on Visualization and Computer Graphics 9*, 1 (2003), 30–42.

[KV03b] KALAIAH A., VARSHNEY A.: Statistical point geometry. In *Eurographics Symposium on Geometry Processing* (2003), pp. 107–115.

[Lin01] LINSEN L.: Point cloud representation. In *Technical report No. 2001-3* (2001), Faculty for Computer Science, Universitaet Karlsruhe.

[LW85] LEVOY M., WHITTED T.: The use of points as a display primitive. In *Technical Report TR 85-022* (1985), University of North Carolina at Chapel Hill.

[MD03] MOENNING C., DODGSON N. A.: A new point cloud simplification algorithm. In *Proceedings 3rd IASTED Conference on Visualization, Imaging and Image Processing* (2003).

[MPN*02] MATUSIK W., PFISTER H., NGAN A., ZIEGLER R., MCMILLAN L.: Acquisition and rendering of transparent and refractive objects. In *Eurographics Workshop on Rendering (EGRW)* (2002), pp. 267–278.

[OBA*03] OHTAKE Y., BELYAEV A., ALEXA M., TURK G., SEIDEL H.-P.: Multi-level partition of unity implicits. *ACM Transactions on Graphics 22*, 3 (SIGGRAPH 2003), 463–470.

[Paj03] PAJAROLA R.: Efficient level-of-details for point based rendering. In *Proceedings IASTED Computer Graphics and Imaging* (2003).

[PGK02] PAULY M., GROSS M., KOBBELT L.: Efficient simplification of point-sampled surfaces. In *Proceedings IEEE Visualization 2002* (2002), pp. 163–170.

[PKKG03] PAULY M., KEISER R., KOBBELT L., GROSS M.: Shape modeling with point-sampled geometry. *ACM Transactions on Graphics 22*, 3 (SIGGRAPH 2003), 641–650.

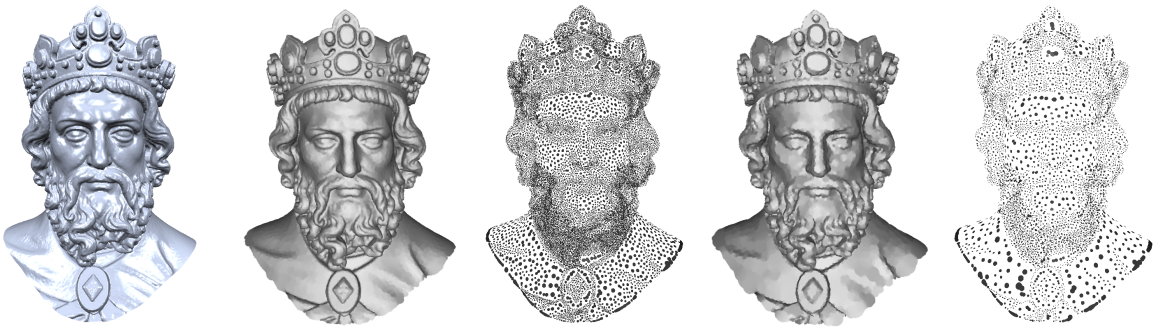

Figure 12: *The Charlemagne model (left, 598K points) approximated with disk-shaped splats by setting the relative error threshold to* $\varepsilon = 0.03\%$ *(middle two, 66401 splats) and* $\varepsilon = 0.1\%$ *(right two, 17445 splats).*

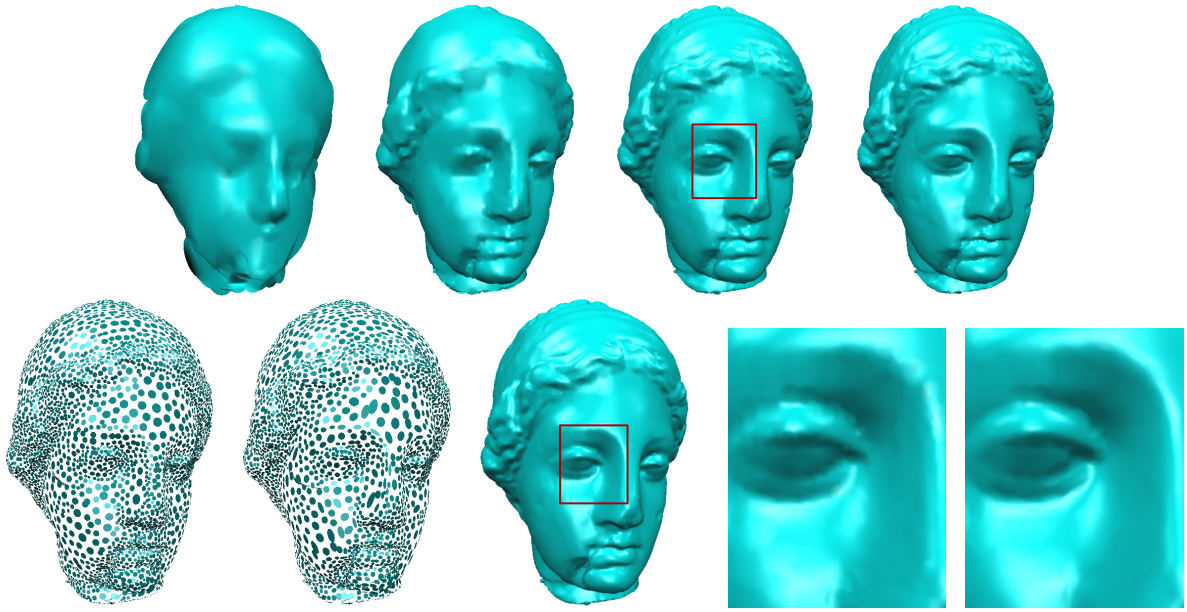

Figure 13: *From left to right in top row are the sub-sampled Igea models with* 218*,* 1621*,* 10737*, and* 29847 *filtered circular splats and decreasing error* ε *= 0.8%, 0.2%, 0.05%, 0.02% respectively. The bottom row shows approximations with* 10737 *circular splats (left) and* 8556 *elliptical splats (middle two) at same error* 0.05% *and their respective zoom-in views (right two). Notice the improved visual quality of elliptical splats (far right).*

[PZBG00] PFISTER H., ZWICKER M., BAAR J. V., GROSS M.: Surfels: Surface elements as rendering primitives. In *Proceedings SIGGRAPH 2000* (2000), pp. 335–342.

[RL00] RUSINKIEWICZ S., LEVOY M.: Qsplat: A multiresolution point rendering system fo large meshes. In *Proceedings SIGGRAPH 2000* (2000), pp. 343–352.

[RPZ02] REN L., PFISTER H., ZWICKER M.: Object space EWA surface splatting: A hardware accelerated approach to high quality point rendering. In *Eurographics 2002 Conference Proceedings* (2002), pp. 461–470.

[SAG03] SURAZHSKY V., ALLIEZ P., GOTSMAN C.: Isotropic remeshing of surfaces: a local parameterization approach. In *Proceedings of 12th International Meshing Roundtable* (2003).

[Sam94] SAMET H.: *The Design and Analysis of Spatial Data Structures*. Addison-Wesley, 1994.

[SD01] STAMMINGER M., DRETTAKIS G.: Interactive sampling and rendering for complex and procedural geometry. In *Proceedings Eurographics Rendering Workshop 2001* (2001), pp. 151–162.

[Tur91] TURK G.: Generating textures on arbitrary surfaces using reaction-diffusion. *Computer Graphics (SIGGRAPH 1991) 25*, 4 (1991), 289–298.

[WFP*01] WAND M., FISCHER M., PETER I., AUF DER HEIDE F. M., STRASSER W.: The randomized z-buffer algorithm: Interactive rendering of highly complex scenes. In *Proceedings ACM SIGGRAPH 2001* (2001), pp. 361–370.

[ZPBG01] ZWICKER M., PFISTER H., BAAR J. V., GROSS M.: Surface splatting. In *Proceedings SIGGRAPH 2001* (2001), pp. 371–378.

[ZPKG02] ZWICKER M., PAULY M., KNOLL O., GROSS M.: Pointshop 3d: An interactive system for point-based surface editing. *ACM Transactions on Graphics 21*, 3 (SIGGRAPH 2002), 322–329.

[ZRB*04] ZWICKER M., RÄSÄNEN J., BOTSCH M., DACHSBACHER C., PAULY M.: Perspective accurate splatting. In *Graphics Interface 2004* (2004).