Intrinsic point cloud simplification

Carsten Moenning and Neil A. Dodgson
Computer Laboratory, University of Cambridge, Cambridge CB3 0FD, UK
{cm230,nad}@cl.cam.ac.uk

Abstract

Modelling and visualisation methods working directly with point-sampled geometry have developed into attractive alternatives to more traditional mesh-based surface processing. In this paper, we consider a vital step in any point-based surface processing pipeline, point cloud simplification. Building upon the intrinsic point cloud simplification idea put forward in Moenning and Dodgson [13], we obtain a simplification algorithm allowing for intuitive density control and satisfying a set of important requirements unsupported by existing simplification techniques. The algorithm operates efficiently and gives a point set density guarantee. It supports both sub- and resampling of the input point set and allows for uniform and user-controlled feature-sensitive simplification. It can further deal with non-uniformly distributed point sets and point-sampled geometry featuring illegitimate holes of simple complexity. The algorithm is inherently progressive and supports the generation of multiresolution representations in the form of levels of detail. We are primarily concerned with describing the conceptual framework of our intrinsic approach and show its viability by giving a number of application examples using massive data sets.

Keywords: Point cloud simplification, Point-sampled geometry processing, Fast Marching level set methods

1 Introduction

Modern 3D data acquisition devices produce point sets of enormous density due to submillimeter measurement precision. Surface reconstruction algorithms either fail to cope with the inherent redundancy of these point sets or produce highly dense surface meshes. To facilitate meaningful further mesh-based processing, these meshes require mesh simplification algorithms which are frequently as time and memory demanding as the preceding surface reconstruction step. By simplifying the point cloud first and, if required, generating the surface mesh from the simplified point set, the surface reconstruction problem is accelerated significantly and the mesh simplification step is avoided altogether. Alternatively, surface reconstruction and thus mesh-based processing and its inherent maintenance overhead may be completely replaced by more efficient point-based modelling and visualisation algorithms at little or no loss in quality. In either case, the simplification of the input point set represents a vital first processing step.

In [13], we suggest to approach the problem of point cloud simplification intrinsically using geodesic Voronoi diagrams [14] and recent advances in Fast Marching methods for point clouds [12]. In this paper, we substantially extend this idea thereby obtaining a simplification method satisfying a set of important requirements unsupported by existing simplification techniques. We are primarily concerned with discussing the conceptual framework of the extended algorithm. We begin by formally stating the problem, then we survey related work and outline our contribution. Section 2 presents our farthest point sampling technique for point-sampled geometry which is at the heart of the simplification algorithm discussed in Section 3. Section 4 indicates the viability of our approach by providing application examples for a number of massive data sets and a first set of experimental results. Section 5 concludes the paper.

1.1 Problem statement

Given a set of samples $P_{N_1} = \{p_1, p_2, \ldots, p_{N_1}\}$ acquired from a smooth, compact two-manifold surface embedded in $\mathbb{R}^3$, simplify P_{N_1} to a point set P_{N_2} of target model size $N_2 < N_1$ subject to user-controlled refinement condition $\rho > 0$. The input point set does not have to be uniformly distributed. Furthermore, any noise is not required to be smoothed out in a pre-processing step.

Our aim was to develop a dedicated point cloud simplification algorithm allowing for simple density control whilst satisfying a set of important requirements. These are a point set density guarantee to support meaningful further processing, memory and execution efficiency and support for both uniform and user-designated feature-sensitive simplification. The algorithm should not be restricted to the generation of subsets of the input point cloud but support its resampling as well. Meeting this requirement would also help to satisfy the final requirement of being able to deal meaningfully with non-uniformly distributed input point sets and illegitimate holes in the form of undersampled regions.

1.2 Related Work

For the sake of brevity, we focus on dedicated point cloud simplification algorithms, with an eye to how they perform against the set of requirements listed in the previous section.

Dey et al. [6] were among the first to present a dedicated point cloud simplification algorithm which exploits the particular structure of 3D Voronoi cells of a densely distributed input point set both to detect oversampled regions and to determine candidate points for removal. Subsequent point decimation observes a user-controlled density condition. Due to the use of the medial axis-related local feature size concept [2], their method is inherently sensitive to changes in local curvature estimates. The algorithm does not support adaptive decimation driven by changes in a measure other than or in addition to local curvature. It is restricted to the generation of a subset of the input point set and cannot handle non-uniformly distributed point clouds or point sets featuring (illegitimate) holes. It requires the computation and maintenance of 3D Voronoi diagrams and therefore tends to be computationally and memory demanding.

Linsen's [11] simplification method for point sets associates each input point with an information content measure and iteratively deletes points with lowest entropy. The information content measure represents a weighted sum of local curvature, non-uniformity and colour variation computed in the candidate point's k nearest neighbourhood enhanced by a maximum angle criterion. The simplification algorithm is simple and effective but does not give any density guarantee and is limited to the generation of point cloud subsets. Input clouds may therefore be simplified to prohibitively unevenly distributed point sets and non-uniformly distributed input point sets will necessarily result in non-uniformly distributed output point sets. The resampling of the input cloud, which may be necessary in either case to support any effective further processing, is not addressed in the paper.

Alexa et al. [1] "Moving Least Squares" (MLS) technique locally approximates a smooth two-dimensional manifold surface by bivariate polynomials fitted with the help of weighted least squares. As part of their point decimation scheme, they judge each input point's importance by its distance from its projection onto the MLS surface computed from all input points other than the point under consideration. Those points exhibiting the smallest distance are considered redundant and are removed iteratively. Flexible feature-sensitive simplification is not supported. Similar to the techniques discussed above, this method produces a subset of the input point cloud and may require resampling to avoid any excessive non-uniformity in the simplified point set.

In Moenning and Dodgson [13], we present an intrinsic coarse-to-fine point cloud simplification algorithm supporting execution efficient uniform and feature-sensitive simplification driven by any combination of point weights reflecting changes in local curvature, colour, etc. The progressive nature of its uniform version inherently supports the generation of multiresolution representations of the input point set in the form of levels of detail. The method cannot deal with non-uniformly distributed input sets, does not give any density guarantee and does not allow for the generation of simplified point sets other than true subsets of the input data.

Pauly et al. [15] adapt various widely used mesh simplification techniques to the point cloud simplification scenario. Their iterative simplification method is reported to produce the best results in terms of average geometric accuracy but does not allow for simple control of point set density and requires relatively expensive pre-computations. Particle simulation is found to represent the next-best method in terms of approximation accuracy and the best choice in terms of point set density control but is generally computationally demanding. Uniform incremental clustering is computationally efficient but is reported to produce the highest approximation error and is not naturally extensible to simplification sensitive to measures other than changes in local curvature. Similarly, hierarchical clustering is memory and execution efficient but even in its adaptive version yields point sets of approximation error only slightly lower than that introduced by the method performing poorest on this criterion, uniform incremental clustering. The methods support surface resampling but do not come with any density guarantee.

In summary, while all of the simplification algorithms discussed above meet a subset of the requirements listed in 1.1, none satisfies them all. The following section summarises the contribution of our approach towards providing such an algorithm.

1.3 Our contribution

By building upon the intrinsic approach towards point cloud simplification put forward in [13], we obtain an enhanced algorithm allowing for efficient coarse-to-fine simplification with an easy to control, guaranteed density. The algorithm supports the uniform and adaptive simplification of uniformly distributed point sets and point clouds featuring non-excessive non-uniformity and/or illegitimate holes of simple complexity in the form of undersampled regions. Adaptivity is supported in the form of any combination of (positive) point weights either computed on-the-fly or imported in the form of pre-computed importance maps. The algorithm can be used to generate either a subset of the point cloud or a resampled simplified point set. It inherently supports progressive transmission, the generation of multiple levels of detail and selective refinement. This algorithm represents the first point cloud simplification technique combining simple density control with the above set of desirable features.

The algorithm's execution and memory efficiency directly results from its intrinsic nature due to the use of the optimal extended Fast Marching method for the computation of geodesic distance maps across point clouds introduced in [12]. We exploit these powerful techniques for the incremental computation of (discrete) geodesic Voronoi diagrams. As outlined in the following section, this allows us to locate both farthest point samples and nearest neighbours efficiently. This is in contrast to the frequent use of computationally and memory demanding three-dimensional extrinsic (Euclidean) Voronoi diagrams/Delaunay triangulations for similar purposes [2; 3; 6]. In the following, we discuss the concepts underpinning our approach in detail.

2 Intrinsic farthest point sampling of point-sampled geometry

Our simplification algorithm is based on the idea of progressive intrinsic farthest point sampling of a surface in point cloud form. The algorithm exploits the observation that intrinsic farthest point sampling can be shown to be closely related to the incremental computation of a geodesic Voronoi diagram. We therefore begin this section by briefly reviewing the notion of geodesic Voronoi diagrams. This is followed by the discussion of the link between progressive intrinsic farthest point sampling and incremental geodesic Voronoi diagram computation. As part of our simplification algorithm, we perform the discretised version of this Voronoi diagram computation by using the extended Fast Marching concept put forward by [12]. We therefore conclude this section with a brief summary of this important method.

2.1 Geodesic Voronoi diagrams

Given a finite number n of distinct data sites $P := \{p_1, p_2, \ldots, p_n\}$ on a smooth, compact manifold M, define the bisector of p_i, $p_j \in P$, $p_i \neq p_j$, as geodesically equidistantial loci with respect to p_i, p_j, i.e. $L(p_i, p_j) = \{q \in M | d_M(p_i, q) = d_M(p_j, q)\}$, where $d_M(p, q)$ denotes the length of the geodesic from p to q, p, $q \in M$. Let the dominance region of p_i, $D(p_i, p_j)$, denote the region of M containing p_i bounded by $L(p_i, p_j)$. The Voronoi region of p_i with respect to point set P, $V(p_i, P)$, is given by $V(p_i, P) = \bigcap_{p_j \in P, p_j \neq p_i} D(p_i, p_j)$ and consists of all points for which the geodesic distance to p_i is smaller than the geodesic distance to any other $p_j \in P$. We define the bounded Voronoi region, $BV(p_i, P)$, as the conjunction of $V(p_i, P)$ with the domain. The boundary shared by a pair of Voronoi cells is called a Voronoi edge. Voronoi edges meet at Voronoi vertices. For every Voronoi vertex v, there exist at least three points in P which are geodesically equidistant from v. The bounded Voronoi diagram of P, $BVD(P)$, is given by

$$BVD(P) = \bigcup_{p_i \in P} \partial BV(p_i, P) \tag{1}$$

where $\partial BV(p_i, P)$ denotes the boundary of $BV(p_i, P)$.

We approximate a geodesic *BVD* using weighted geodesic distance maps. The *BVD* is generated following a expanding waves view. That is, in analogy to the dropping of pebbles into still water, circular fronts move across the surface from the point of impact. The locations where wave fronts meet or leave the domain define the bounded surface Voronoi diagram of the points of impact. See Figure 1 for a triangular mesh-based example produced using public domain software [16]. Wave propagation is discretised and simulated accurately by the Fast Marching level set-based viscosity solution of the Eikonal equation [17]. This way concentric circles are computed on the surface in the form of discrete geodesic offsets and Voronoi edges and vertices are obtained as loci of intersection between geodesic offset curves. The following section shows the close relationship of this Voronoi diagram concept with the notion of progressive (intrinsic) farthest point sampling.

Figure 1: Wave propagation for the incremental computation of geodesic Voronoi diagrams and thus progressive intrinsic (red) farthest sample point localisation of 10, 11 and 12 sample sites on a triangulated surface (from left to right).

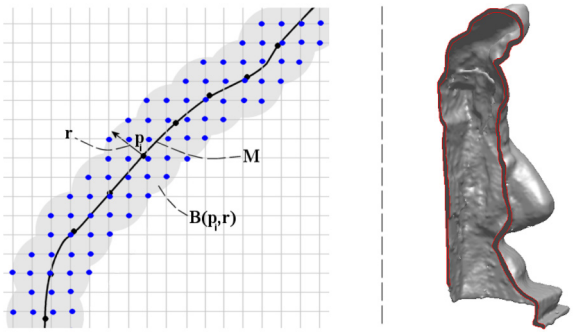

Figure 2: Geodesic distance mapping using extended Fast Marching for point clouds operates in an offset band consisting of the union of balls $B(p_i,r)$ centred at (black) points p_i of the surface M (left). Only those (blue) grid points falling inside the offset band are considered during processing. Cross-sectional view of a constant radius offset band for the Michelangelo Youthful data set (right).

2.2 Farthest point sampling

Farthest point sampling was introduced in an image sampling context by Eldar et al. [7] and is based on the intuitively appealing idea of repeatedly placing the next sample point in the middle of the least-known area of the sampling domain.

To see the close relationship of the farthest point sampling principle with incremental geodesic Voronoi diagram construction, note that the point farthest away from the current set of sample sites, S, is represented by the centre of the largest geodesic circle empty of any site $s_i \in S$. Due to the propagation of circular fronts from the s_i outwards during the computation of their Voronoi regions, this "largest empty circle" problem is solved by a vertex of the (bounded) geodesic Voronoi diagram of S, $BVD(S)$ [5]. Using this close link with Voronoi diagrams, farthest point samples may be generated progressively by incrementally constructing a geodesic Voronoi diagram across the sampling domain (Figure 1). Whilst previous work exploited this link to compute farthest point samples on triangular and implicit surfaces, we propose its extension to the point cloud case using the extended Fast Marching method of [12] discussed next.

2.3 Fast Marching for geodesic distance mapping across point clouds

In the following, we summarise the extension of the well-known original Fast Marching level set method [10; 17; 18] to the case of surfaces in point cloud form introduced in [12]. Our review is necessarily terse, presenting just the key results. For full details, see the original paper [12].

Let $P_n = \{p_1, p_2, \ldots, p_n\}$ denote a set of points acquired from a smooth, compact manifold surface M in $m \geq 3$ dimensions. Define the r-offset Ω_P^r as the union of Euclidean balls with radius r centred at points $p_i \in P$

$$\Omega_{P_n}^r := \bigcup_{i=1}^{n} B(p_i, r) = \{x \in \mathbb{R}^m : d(p_i, x) \leq r\},$$

for all i and where $d(.,.)$ denotes the Euclidean distance function (Figure 2). To approximate the weighted intrinsic distance map originating from a source point $q \in M$ on M, Mémoli and Sapiro [12] suggest to compute the Euclidean distance map in $\Omega_{P_n}^r$. That is

$$|\nabla_M T_M(p)| = F(p), \tag{2}$$

for $p \in M$ and with boundary condition $T_M(q) = 0$ is approximated by

$$|\nabla T_{\Omega_{P_n}^r}(p)| = \tilde{F}(p), \tag{3}$$

for $p \in \Omega_{P_n}^r$ and boundary condition $T_{\Omega_{P_n}^r}(q) = 0$. $\tilde{F}$ represents the (smooth) extension of the propagation speed F on M into $\Omega_{P_n}^r$. $T(p)$ denotes the arrival time at p of the front originating from q and ∇_M and ∇ represent the intrinsic and the Euclidean gradient operator respectively. The problem of computing an intrinsic distance map is therefore transformed into the problem of computing an extrinsic (Euclidean) distance map in the tubular neighbourhood $\Omega_{P_n}^r$ around the surface, i.e. in an Euclidean manifold with boundary. In [12], the authors prove uniform probabilistic convergence between these two distance maps and thus show that the approximation error between the intrinsic and extrinsic distance maps is of the same theoretical order as that of the Fast Marching algorithm for both noise-free and noisy point clouds (assumming noise to be bounded from above by r). The Fast Marching method can therefore be used to approximate the solution to (3) in a computationally optimal manner and without the need for any prior surface reconstruction by only slightly modifying the original Cartesian Fast Marching technique to deal with bounded spaces. The relatively straightforward implementation of this technique consists of, firstly, computing the offset band $\Omega_{P_n}^r$, followed by the application of standard Cartesian Fast Marching restricted to $\Omega_{P_n}^r$. For more implementational details, see [12]. Memory efficiency is achieved by only considering those grid points falling inside the offset band (Figure 2). This important technique is utilised in our point cloud simplification algorithm presented next.

3 Intrinsic point cloud simplification

We first present the point cloud *subsampling* algorithm extending the idea in [13], followed by the discussion of its extension to the *resampling* case. We then relax the assumption of uniformly densely distributed point clouds and describe our approach towards dealing with the resulting issues. The section concludes with a simple proof for the density guarantee given by our simplification algorithm.

3.1 Intrinsic point cloud subsampling

Our algorithm for the subsampling of uniformly dense input point clouds is summarised in the following.

The grid points which make up $\Omega_{P_n}^r$, and those grid points only, for a fixed r and a given grid resolution are computed in a one-off pre-processing step. For bounds on r to avoid an intersecting boundary or an unconnected domain, see Mémoli and

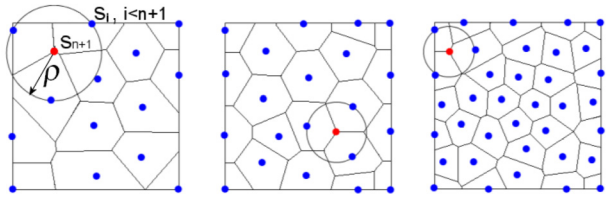

Figure 3: *Effect of different values of* ρ. The user controls the maximum distance from the next (red) farthest point candidate to the sample set S (blue points), i.e. the radius ρ of the largest empty circle in the domain of the simplified point set. This in turn bounds interpoint distances.

Sapiro [12]. During the pre-processing step, the input point set is bucketed in this offset grid and an initial sample point is selected randomly. The actual subsampling algorithm adds one new point to the set S of sample points at each step and proceeds as follows.

---

1. Initialisation: The algorithm starts with reading in the offset grid data and initial sample point, $s_1 \in M$. The grid points enclosing s_1 are initialised with their analytic distance from s_1. We generate a second, s_2, and third, s_3, starting sample by repeatedly selecting the point farthest away in the geodesic distance map approximations of s_1 and s_1, s_2 respectively. Once $|S| \geq 3$, the algorithm constructs the initial discrete geodesic bounded Voronoi diagram, $BVD(S)$, by simultaneously propagating fronts from the initial sample points outwards. The vertices of the geodesic $BVD(S)$ are given by those input points closest to grid points entered by three or more propagation waves (or two for points on the domain boundary) and are therefore obtained as a by-product of the propagation process. The vertices' arrival times are inserted into a max-heap data structure.

2. Sampling: The algorithm proceeds by extracting the root from the max-heap. This yields the next farthest point sample in the form of the input point closest to the root's grid location. This sample is inserted into $BVD(S)$ by resetting its arrival time to zero and propagating a front away from it. The front will continue propagating until it hits (grid) points featuring lower arrival times and thus belonging to neighbouring Voronoi regions. The arrival times of updated grid points are updated correspondingly in the min-heap using back pointers. New and obsolete Voronoi vertices are inserted or removed from the max-heap respectively. The algorithm continues extracting the root from the max-heap until the sample point budget has been exhausted and/or the refinement condition has been met. This sampling technique is particularly easily made adaptive by allowing $F(p)$ to vary with any (positive) point weights either computed on-the-fly or imported in the form of pre-computed importance maps.

The refinement condition is formulated in the form of a user-controlled density condition $\rho > 0$ which we show in Section 3.4 to bound the distance between sample points. More specifically, the simplified point set is refined until the next farthest point candidate's distance map value is no longer at least as large as the user threshold indicating that S has become sufficiently dense (Figure 3). As an alternative to the selection of a global value for ρ, the density condition can be formulated as a function of, for example, local object properties provided $\rho > 0$ holds true throughout.

As regards the algorithm's computational and memory efficiency, extracting the root from, inserting into, updating and removing from the max- and/or min-heap with subsequent re-heapifying are $O(\log W_1)$ and $O(\log W_2)$ operations respectively, where W_1 represents the number of elements in the max-heap and W_2 denotes the number of elements in the min-heap. W_1 and W_2 are $O(N)$, N representing the number of grid points in the offset band $\Omega^r_{P_n}$. The accessing of existing max- and min-heap entries is $O(1)$ due to the use of back pointers from the grid to the heaps. The detection of the Voronoi vertices is a by-product of the $O(N \log N)$ front propagation. This $O(N \log N)$ process is performed up to N_2 times, where N_2 represents the output size. N_2 is $O(N)$ yielding a running time of $O(N^2 \log N)$. It is important to note in this context that N will generally be relatively small and that the size of W_2 decreases with increasing sample size moving min-heap re-heapifying closer to $O(1)$ and making the $O(N \log N)$ front propagation close to linear in complexity thereby yielding considerably more favourable runtime behaviour in practice. This observation is confirmed by the experimental results reported in Section 4. The algorithm's memory requirements are relatively low due to the consideration and storage of grid points located in the offset band only. The actual requirements correspondingly vary proportionally with the size of the offset radius and inversely with the grid spacing.

3.2 Intrinsic point cloud resampling

Fast Marching farthest point sampling of a uniformly distributed point cloud yields a subsample. If resampling of the point cloud is required instead, local surface approximation and a corresponding projection operator are needed. We use a variant of Alexa et al. [1] MLS method, briefly summarised below, to fit a bivariate polynomial to a local neighbourhood of input points for sample s_i to be projected on. Apart from this additional projection operation, steps 1. and 2. of the simplification algorithm remain unchanged.

The MLS procedure involves two weighted least squares computations, the first of which estimates the normal n of a local support plane H for s_i by minimising the weighted distances of the points in the k nearest neighbourhood, NN_{s_i}, of s_i

$$\min n^T U n,$$

with $||n = 1||$ and $U = \{u_{vw}\} \in \mathbb{R}^{3x3}$ representing a weighted covariance matrix

$$u_{vw} = \sum_{j=1}^{k} \theta_j (p_{j_v} - s_{i_v})(p_{j_w} - s_{i_w}),$$

with $p_j \in NN_{s_i}$ and $\theta_j = e^{-d(p_j,s_i)^2/h^2}$; d denotes Euclidean distance and h represents a global scale parameter. The value of h will usually reflect any prior knowledge of global sampling density such as the sampling resolution of the device used to acquire the point geometry. Eigenanalysis of U yields the principal components of NN_{s_i} in the form of orthogonal eigenvectors e_1, e_2, e_3 and corresponding real eigenvalues λ_1, λ_2, λ_3 spanning a local covariance ellipsoid. Provided that $\lambda_1 \leq \lambda_2 \leq \lambda_3$, λ_1 describes the points' covariance along the local surface normal and e_1 may be chosen as local normal estimate [9]. Subsequently, this estimate can be used to move H closer to the underlying surface. Alternatively, given that the input point set is known to be (close to) noise-free, H passes through s_i and any modification of H's position is not required.

As part of the second weighted least squares computation, the k nearest neighbours of s_i are projected onto the support plane H. Their local 2D coordinates in an orthonormal coordinate system centred at the projection q_i of s_i onto H are used to fit a bivariate polynomial to the points in the neighbourhood. The weights used in the regression now reflect the distance of the points in NN_{s_i} to q_i instead of s_i, i.e. $\theta_j = e^{-d(p_j,q_i)^2/h^2}$. The final location of the farthest point sample s_i is then given by the projection of q_i onto the fitted polynomial.

Thus far, we have been assuming that the input point set is uniformly dense. The following section addresses the issues arising from dealing with point clouds of variable density.

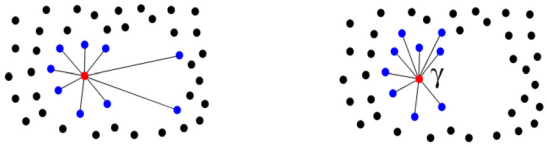

Figure 4: Enhanced k nearest neighbourhood (left) of a (red) point p. By controlling for violations of a preset maximum angle γ between successive neighbours, the anisotropic distribution of points around p is taken into account and interpoint spaces are bridged. The corresponding standard k nearest neighbourhood is shown on the right.

3.3 Non-uniformly distributed point clouds

When dropping the assumption of uniformly distributed input point sets, a number of problems arise. Firstly, continuing to use a global feature size parameter h for MLS approximation will result in poorly fitting regressions. Secondly, the radii of the offset balls, $B(p_i, r_i)$, $p_i \in P$, need to be adjusted to account for local density variations and undersampled regions, i.e. illegitimate holes. Finally, it is generally important to note the limited usefulness of standard k nearest neighbourhoods in this context. In the following, starting with the discussion of an enhanced k nearest neighbourhood concept, we describe our approach towards dealing with these issues.

3.3.1 Enhanced k nearest neighbourhood

It is well-known that standard k nearest neighbourhoods are ill-suited for collecting proximity information when dealing with point sets of variable density [8]. We therefore follow [11] and control for a maximum angle between successive neighbours around an input point p to ensure a spherical distribution of neighbours all around p when determining local proximity (Figure 4).

We compute the enhanced neighbourhood, NN_p, for an input point p, by first growing NN_p until it is comprised of a preset minimum number of neighbours k. Our experimental results indicate a choice of $8 \leq k\ (\leq 18)$ for the neighbourhood size to be well-suited for most cases. We then approximate the local normal vector by first computing the positive semi-definite weighted covariance matrix $C = \{c_{vw}\} \in \mathbb{R}^{3x3}$, of the points $p_i \in NN_p$ around their centroid c_{NN_p},

$$c_{vw} = \sum_{j=1}^{k} \theta_j (p_{j_v} - c_{NN_p})(p_{j_w} - c_{NN_p}),$$

with θ_j as in (4) below and $c_{NN_p} = 1/k \sum_{i=1}^{k} p_i$, $k = |NN_p|$. This definition of c_{NN_p} needs to be adapted when dealing with relatively strongly non-uniformly distributed point sets. In this case, c_{NN_p} may simply be defined as the weighted centroid $c_{NN_p} = \sum_{i=1}^{k} w_i p_i$, provided $w_i = \theta_i / \sum_{j=1}^{k} \theta_j$ with $\theta_i \geq 0$, for all i [4].

Similar to the MLS technique discussed above, provided the eigenvalues of C are both sufficiently dissimilar and $\lambda_1 \leq \lambda_2 \leq \lambda_3$ holds, the eigenvector associated with λ_1 is taken as estimate of the local normal vector; otherwise, the approximation is unreliable and the growing of NN_p resumes.

The normal vector estimate is subsequently used to project the p_i into the local support plane. The algorithm determines the 2D coordinates of the p_i in a local orthonormal coordinate system across the plane and centred at p. We transform these coordinates into polar coordinates and control for the difference in polar angle between successive neighbours in counter-clockwise order around origin p. If none of the angles between successive neighbours is found to be larger than a certain threshold, a valid enhanced k nearest neighbourhood has been found; otherwise, the algorithm continues growing the neighbourhood until replacement neighbours have been found which do not violate the angular threshold. This assumes that we are dealing with a closed manifold. In the case of manifolds with boundary, border points may be detected as outlined in Section 3.3.3.

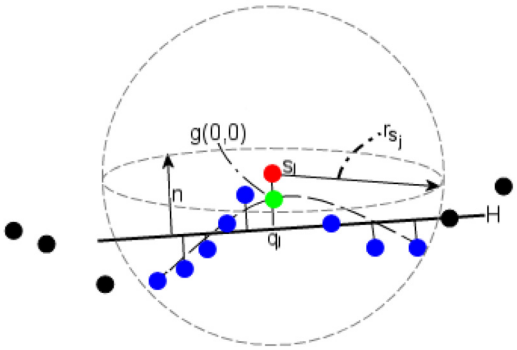

Figure 5: *Local surface approximation and projection* for non-uniformly distributed points. Sample s_j (red) is projected onto a bivariate polynomial $g(x_i, y_i)$ locally fit to (blue) points $p_i \in NN_{s_j}$ weighted by a quadratic B-spline with its parameter scaled by r_{s_j}.

As discussed next, this neighbourhood concept is used both to replace h by localised weights in the resampling regressions and to automatically determine offset ball radii in a pre-processing step.

3.3.2 Localised weighting for adaptive MLS approximation

Using a global scale parameter such as h when processing a non-uniformly distributed point set will cause poorly fitting regressions and localised weighting for supporting adaptive MLS approximation [15] needs to be used instead. To allow for local changes in point density, our experimental results suggest to set θ_j to the following quadratic B-spline $B(t)$ centred at s_i,

$$\theta_j = B\left(\frac{3d(p_j, s_i)}{2r_{s_i}}\right), \tag{4}$$

with support radius r_{s_i} representing the radius of the sphere centred at s_i. As illustrated in Figure 5, the sphere contains the input points p_j forming the enhanced k nearest neighbourhood of s_i.

3.3.3 Automatic determination of offset ball radii and hole-filling

We compute an adaptive offset band by determining variable offset ball radii, r_i, during the one-off pre-processing step. We start with establishing adjacency information by computing the enhanced k nearest neighbourhood, NN_p, of the input point p under consideration. Once this local proximity information is available, the algorithm determines the Euclidean distance d between p and its neighbour $q \in NN_p$ farthest away from p. Provided $d(p,q)$ is larger than any radius currently associated with p and q, $d(p,q)$ is the new radius of both the offset balls centred at p and q. The corresponding grid vertices are included in the band.

This first approach does not pay any attention to the risk of connecting different parts of the underlying surface. It may further bridge legitimate holes on manifolds with boundary. To avoid these problems, the maximum permissible radius d is bounded locally as discussed in [12]. If, due to the size of a hole, the algorithm cannot

determine such a value locally, a user-controlled (global) maximum radius is considered instead. If no spherical neighbourhood can be detected for this radius either, p either represents a border point or an undersampled region has been encountered which is too large to be bridged by NN_p subject to the global radius maximum. In this case, the user is asked interactively to label it either as a border point or as an illegitimate hole by setting a new (local) radius maximum.

Following this pre-processing step, a tubular neighbourhood is available to the simplification algorithm across undersampled regions. When processing such regions, there do not exist any input points near samples located in parts of the offset band which bridge illegitimate holes. These holes are filled by our adaptive MLS-based point cloud resampling technique. During resampling, a bivariate polynomial is fitted across the hole with the help of the surrounding input points in the sample's enhanced k nearest neighbourhood. The sample is then projected onto this polynomial (Figures 5, 6). Eigenanalysis of the local neighbourhood can be used to observe local topology more closely as outlined in [9].

Although this approach supports the processing of the type of moderately non-uniformly distributed point cloud data frequently observed in practice, it does not allow for the processing of strongly non-uniformly distributed point clouds or the filling of complex or sizeable holes. Note in this context that irrespective of the particular neighbourhood concept used, any such constellation prevents the meaningful computation of geodesic distances due to the lack of sufficient information regarding the underlying surface.

3.4 Point set density guarantee

We give a simple guarantee in the spirit of Eldar et al. [7] in terms of the user-controlled refinement condition ρ on the density of point sets produced by our simplification algorithm.

Definition 1 As discussed in Section 2.2, the centre of the largest empty circle coincides at any one stage of the sampling process with a vertex v of $BVD(S)$. Define r_{max} as the radius of this circle at the end of the simplification process, i.e. $r_{max} = \max_{s\in S} d_M(v,s) = \rho$, where $d_M(p,q)$, as defined above (Section 2.1), represents the geodesic distance between the points p and q on the surface M. Denote as r^j_{max}, the radius of the largest circle empty of the first j sample points. Without loss of generality, we assume that 1(a) below is not violated by the initial set of three sample points.

Theorem 1
(a) For the distance between points $s_i, s_j \in S$, $i \neq j$, $d_M(s_i,s_j) \geq \rho$.
(b) For any pair of neighbouring points $s_i, s_j \in S$, $d_M(s_i,s_j) \leq 2\rho$.

Proof
(a) Note that with increasing sample size, r^j_{max} will not increase,

$$r_{max} \leq r^j_{max}, \tag{5}$$

for $j < n = N$. Let s_i, s_j denote two sample points on the geodesic circle of v with j sampled after i. By the definition of the algorithm, s_j is placed at a vertex with distance $r^{j-1}_{max} = \max_{s\in S^{j-1}} d_M(v,s)$ from S^{j-1}. Thus, for any s_i with $i < j$, $d_M(s_i,s_j) \geq r^{j-1}_{max}$. From (5), $r_{max} \leq r^{j-1}_{max}$ so that $d_M(s_i,s_j) \geq r_{max} = \rho$. □

(b) Consider a Voronoi vertex v shared by two neighbouring sample points s_i, s_j. By definition, v is equally far away from both s_i and s_j. Thus, by triangle inequality, $d_M(s_i,s_j) \leq 2d_M(v,s_i)$. Since $d_M(v,s_i) \leq r_{max} = \rho$, $d_M(s_i,s_j) \leq 2\rho$. □

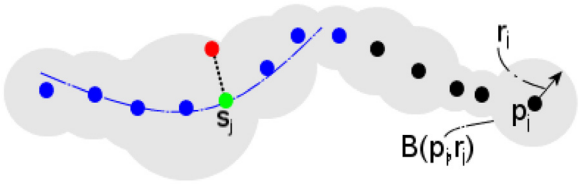

Figure 6: *Variable offset ball radii and hole-filling*. The offset radii r_i are adapted to changes in sample density. Simple holes are filled by projecting sample s_j from its grid location (red) in the bridging offset band onto the approximated surface (blue curve) generated by adaptive MLS (Figure 5).

4 Experimental results and applications

To highlight the features of our algorithm, we apply our simplification algorithm to a number of massive data sets (Figure 7 bottom) and present results for both uniform and feature-sensitive sub- and resampling. We also give a hole-filling example and illustrate the algorithm's inherent support of level of detail-generation. The section concludes with a first set of performance results for a number of different data sets. Each data set was processed on a 1.0GHz AMD machine with 1GB of memory.

Figure 9 shows the distributions and surface reconstructions of the Michelangelo Day and Michelangelo Dawn data sets both uniformly subsampled to 1% of their size. The simplified point sets are irregularly uniformly distributed. This results in cluster- and hole-free coverage of the domain so that high quality further processing such as surface reconstruction and its rendering is supported.

We exploit both this favourable distribution property and the algorithm's progressive nature to produce levels of detail of the Michelangelo Youthful data set (Figure 10). This feature can be utilised for, amongst other things, the progressive transmission of 3D content.

Automatic adaptive offset band generation and uniform resampling are used to deal with holes/non-uniformity in a Buddha data set acquired using a Minolta VIVID 900 laser range scanner. The adaptive offset band bridges the relatively simple holes in the geometry thereby allowing for the upsampling of the local geometry by projecting samples in the offset band onto local polynomials fitted across the holes. As a result and as illustrated in Figure 11, despite the irregularity of the acquired point set, a uniformly simplified representation fully supporting any further processing such as hole-free surface reconstruction is produced.

We estimate local changes in mean curvature by eigenanalysis-based local surface variation [15] to drive curvature-sensitive subsampling of the Venus model; Figure 8. This is achieved by varying the speed of front propagation with the curvature estimates. The point distributions clearly follow local curvature changes despite the rough nature of this estimate. Provided the speed remains strictly positive throughout, adaptive sampling may be driven by any other or additional adaptivity measure. Excessive irregularity of the resulting point set preventing any meaningful further processing can be avoided by enforcing a globally rather than locally defined refinement condition ρ.

Finally, as indicated in figure 7, the algorithm's efficiency is only moderately affected by substantial increases in input or output model size. Due to the consideration of offset band grid points only, only a fraction of the available 1GB of memory was used by the algorithm at any one point. This is in contrast to grid-based techniques which discretise a point set's bounding box at the expense of prohibitively large memory demands for relatively small point sets by today's standards.

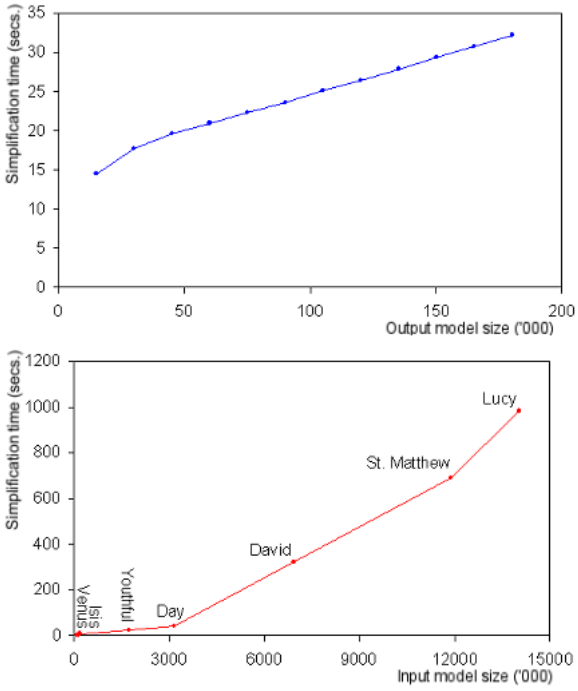

Figure 7: *Uniform simplification execution efficiency* as function of output (top) and input model size (bottom). For the former, the Venus data set (134345 points) was simplified to different output sizes. For the latter, each input model was uniformly subsampled to 1% of its size.

5 Conclusion

We present an intrinsic point cloud simplification algorithm with density guarantee. The algorithm supports efficient uniform and user-designated feature-sensitive sub- and resampling. It can further deal with illegitimate holes of simple complexity and its coarse-to-fine nature inherently allows for the generation of level-of-detail multiresolution representations and progressive transmission of 3D content. The technique put forward in this paper represents the first point cloud simplification algorithm combining simple control of guaranteed density with the above set of desirable features. We were primarily concerned with presenting the algorithm's conceptual framework and showing its viability using a number of massive data sets. The presentation of detailed experimental and comparative results including the quantitative analysis of the approximation error introduced by the algorithm and the generation of ellipsoidal rather than spherical offset bands is left to future work.

Acknowledgements

We gratefully acknowledge the permission to access the Michelangelo data sets granted by the Stanford Computer Graphics group. The Venus data set was obtained from the Cyberware Inc. website at http://www.cyberware.com/samples/index.html. Surfaces were reconstructed from point sets and rendered using Paraform Inc.'s Points2Polys software (http://www.paraform.com/ppdl/).

References

[1] ALEXA, M., BEHR, J., COHEN-OR, D., FLEISHMAN, S., LEVIN, D., AND SILVA, T. 2001. Point set surfaces. In *Proc. 12th IEEE Visual. Conf.*, 21–28.

[2] AMENTA, N., AND BERN, M. 1998. Surface reconstruction by voronoi filtering. In *Proc. 14th ACM Symp. on Comput. Geom.*, 39–48.

[3] BOISSONNAT, J.-D., AND CAZALS, F. 2003. Provably good surface sampling and approximation. In *Proc. SGP'03*, 246–265.

[4] BUSS, S. R., AND FILLMORE, J. P. 2001. Spherical averages and applications to spherical splines and interpolation. *ACM Trans. on Graph. 20*, 2, 95–126.

[5] CHEW, L. P., AND DRYSDALE, R. L. 1985. Voronoi diagrams based on convex distance functions. In *Proc. 1st ACM Symp. on Comput. Geom.*, 235–244.

[6] DEY, T. K., GIESEN, J., AND HUDSON, J. 2001. Decimating samples for mesh simplification. In *Proc. 13th Canadian Conf. on Comput. Geom.*, 85–88.

[7] ELDAR, Y., LINDENBAUM, M., PORAT, M., AND ZEEVI, Y. Y. 1997. The farthest point strategy for progressive image sampling. *IEEE Trans. on Image Proc. 6*, 9, 1305–1315.

[8] FLOATER, M. S., AND REIMERS, M. 2001. Meshless parameterization and surface reconstruction. *Computed Aided Geometric Design 18*, 2, 77–92.

[9] GUMHOLD, S., WANG, X., AND MCLEOD, R. 2001. Feature extraction from point clouds. In *Proc. 10th Int. Meshing Roundtable*, 293–305.

[10] HELMSEN, J., PUCKETT, E. G., COLLELA, P., AND DORR, M. 1996. Two new methods for simulating photolithography development in 3d. In *Proc. SPIE Microlithography IX*, 253–261.

[11] LINSEN, L. 2001. Point cloud representation. In *CS Tech. Rep. 2001-3, Universität Karlsruhe, Germany*.

[12] MÉMOLI, F., AND SAPIRO, G. 2003. Distance functions and geodesics on point clouds. *TR 1902, IMA, University of Minnesota, USA*.

[13] MOENNING, C., AND DODGSON, N. A. 2003. A new point cloud simplification algorithm. *Proc. 3rd Int. Conf. on Visualization, Imaging and Image Processing*, 1027–1033.

[14] OKABE, A., BOOTS, B., AND SUGIHARA, K. 2000. *Spatial Tessellations - Concepts and Applications of Voronoi Diagrams, 2nd ed.* John Wiley & Sons, Chicester, UK.

[15] PAULY, M., GROSS, M., AND KOBBELT, L. P. 2002. Efficient simplification of point-sampled surfaces. In *Proc. 13th IEEE Visual. Conf.*, 163–170.

[16] PEYRÉ, G., AND COHEN, L. 2003. Geodesic re-meshing and parameterization using front propagation. In *Proc. 2nd IEEE Workshop on Variational, Geometric and Level Set Methods in Computer Vision*.

[17] SETHIAN, J. A. 1996. Theory, Algorithms, and Applications of Level Set Methods for Propagating Interfaces *Acta Numerica 5*, 309–395.

[18] TSITSIKLIS, J. N. 1995. Efficient algorithms for globally optimal trajectories. *IEEE Trans. on Automatic Control 40*, 1528–1538.

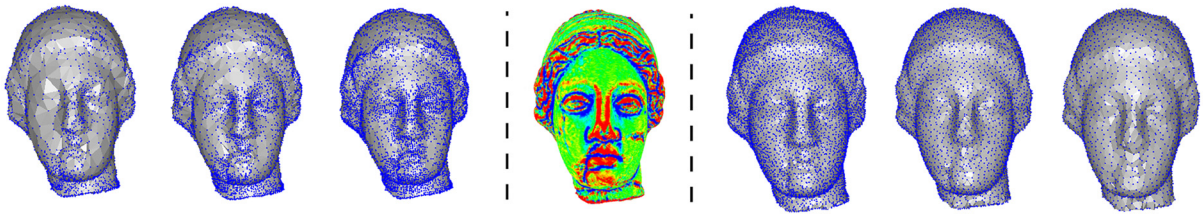

Figure 8: Purely adaptively subsampled Venus point sets driven by changes in local surface variation estimates (left half) vs. uniformly resampled Venus point sets (right half). The model's mean curvature plot is shown in the middle (red - high, yellow - medium, green - low curvature). From centre out, the point sets correspond to 90.0%, 95.0% and 97.5% simplification.

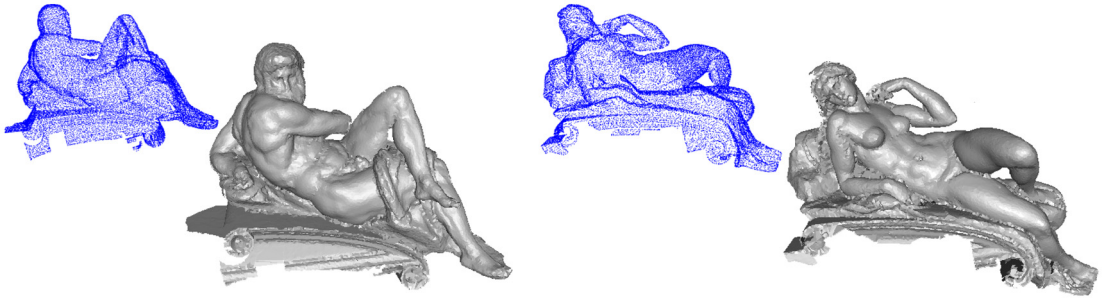

Figure 9: Examples for the quality of the point distributions generated by our algorithm: The Michelangelo Day (left) and Dawn (right) data sets uniformly subsampled to 1% of their original size and their corresponding surface reconstructions. (This Figure is best viewed on-screen).

Figure 10: Levels of detail of the Michelangelo Youthful data set produced from point sets generated by progressive uniform resampling of the original point cloud to 0.25%, 0.5%, 1.0% and 5.0% of its size (from left to right).

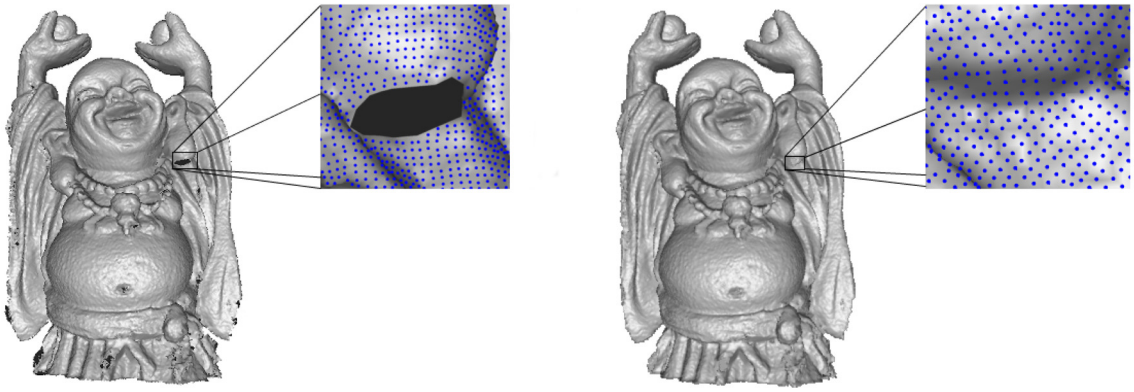

Figure 11: Simple hole-filling example for a Buddha data set acquired using a laser range scanner. Since subsampling of the data set would retain the holes (left), the point cloud was uniformly resampled to 10.0% of its size instead (right) by computing an adaptive offset band and subsequent projection of sample points inside that band onto a local surface approximation.