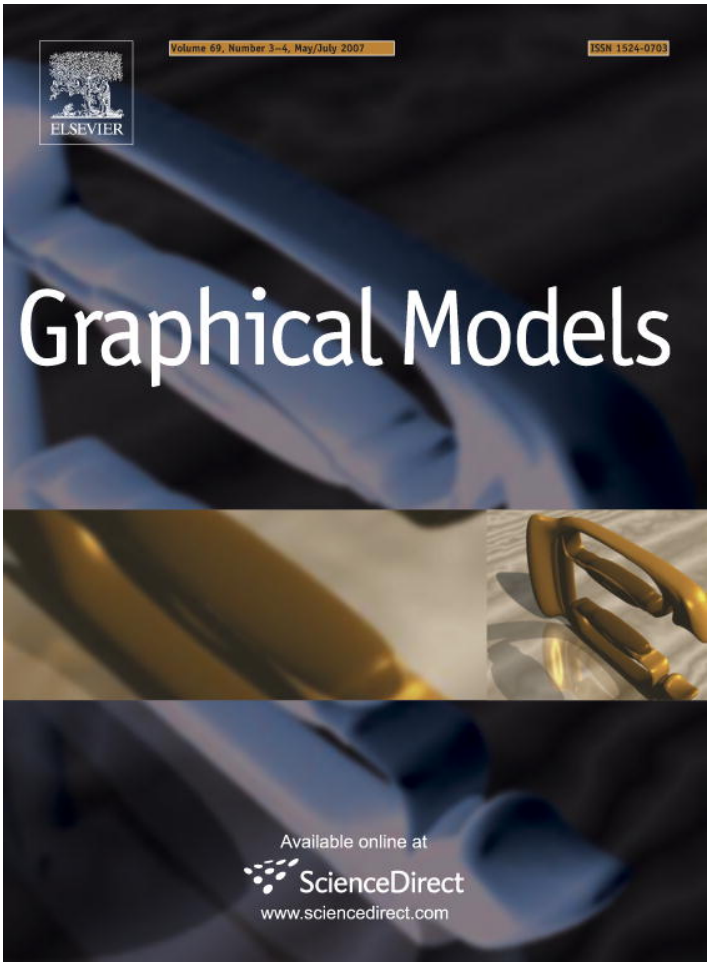

Available online at www.sciencedirect.com
ScienceDirect
Graphical Models 69 (2007) 160–179
Graphical Models
www.elsevier.com/locate/gmod

Meshless geometric subdivision

C. Moenning [a,*,1,2], F. Mémoli [b,2], G. Sapiro [b,2], N. Dyn [c], N.A. Dodgson [a]

[a] Computer Laboratory, University of Cambridge, 15 JJ Thomson Avenue, Cambridge CB3 0FD, UK
[b] Electrical and Computer Engineering Department, University of Minnesota, Minneapolis, MN 55455, USA
[c] School of Mathematical Sciences, Tel Aviv University, Tel Aviv 69978, Israel

Received 6 December 2005; received in revised form 11 July 2006; accepted 28 November 2006
Available online 16 December 2006

Communicated by Dinesh Manocha

Abstract

Point-based surface processing has developed into an attractive alternative to mesh-based processing tools for a number of geometric modeling applications. By working with point clouds directly, processing is based on the raw data and its underlying geometry rather than any arbitrary intermediate representations and generally artificial connectivity relations. We extend this principle into the area of subdivision surfaces by introducing the notion of meshless geometric subdivision. Our approach replaces the role of mesh connectivity with intrinsic point proximity thereby avoiding a number of limitations of mesh-based surface subdivision schemes. Apart from introducing this idea of meshless subdivision, we put forward a first intrinsic meshless subdivision scheme and present a new method for the computation of intrinsic means on Euclidean manifolds.

Keywords: Subdivision; Point-sampled geometry; Point-based surface processing; Fast Marching level set methods; Denoising; Smoothing; Intrinsic mean; Geodesic centroid

1. Introduction

Mesh subdivision has developed into a powerful and widely used tool for the free-form design, editing and representation of smooth surfaces. Subdivision schemes recursively apply a local subdivision operator to a coarse base mesh thereby producing a sequence of refined meshes which quickly converges to a smooth limit surface. The advantages of mesh subdivision include guaranteed global surface smoothness whilst supporting local feature control, the ability to handle surfaces of arbitrary topology and being efficiently and simple to apply once a base mesh is available [1,2].

Unfortunately, when dealing with point-sampled geometry, mesh subdivision requires frequently costly and generally non-geometric surface reconstruction, see, e.g. [3–8], often followed by mesh simplification ([9] and references therein),

* Corresponding author. Fax: +44 1223 334678.
E-mail addresses: cm230@cam.ac.uk, c_moenning@yahoo.com (C. Moenning).
[1] Carsten Moenning performed this work whilst visiting the University of Minnesota at Minneapolis.
[2] This work was partially supported by the Office of Naval Research and the National Science Foundation.

1524-0703/$ - see front matter © 2006 Elsevier Inc. All rights reserved.
doi:10.1016/j.gmod.2006.11.001

parameterisation [10] and remeshing ([11] and references therein), all pre-processing steps to obtain a base mesh. During the reconstruction step, any measurement noise, misalignment of scans, etc. may translate into topological artifacts in the form of erroneous connectivity and genus [12]. This hinders subsequent mesh simplification and remeshing and thus mesh subdivision processing. Also, in the case of extremely high-dimensional manifolds by samples [13], mesh pre-processing breaks down at the surface reconstruction step and it needs to be worked directly with the point cloud.

This paper advocates the use of meshless, or point cloud, geometric subdivision. We propose to avoid the generally costly, error-prone and non-geometric mesh pre-processing steps and instead to work with the point-sampled geometry of three or higher-dimensional manifold surfaces directly using intrinsic subdivision rules. In this paper, we show the conceptual viability of this notion of meshless geometric subdivision by introducing the first intrinsic meshless subdivision scheme using geodesic centroids of intrinsic point neighbourhoods. Apart from this contribution, we put forward a new method for the computation of these geodesic means, which by itself is of interest in other areas such as intrinsic statistical shape analysis [14] and variational theory [15,16].

Some of the related geometric operations may be approximated in an Euclidean context when working with large sampling densities, regular meshes and very local subdivision rules. Working with the point-sampled geometry directly and intrinsically, however, avoids the need for (non-geometric) mesh pre-processing steps and special rules dealing with irregular mesh connectivity. Unlike the non-geometric nature of Euclidean-based (pre-)processing, meshless intrinsic subdivision inherently captures the non-linear intrinsic structure of the object geometry and the principle applies equally well to three and higher-dimensional surfaces. Although it is possible to obtain a geodesic mesh representation of a point-sampled surface [17,18], to continue performing subdivision truly intrinsically, this mesh would have to be modified repeatedly during each iteration to re-enforce its intrinsic nature and more than one mesh would be required to be able to compute correct geodesic distances.

Our algorithm operates intrinsically throughout without the need for prior surface reconstruction with the help of the geodesic distance mapping algorithm for point clouds presented in Mémoli and Sapiro [19]. Following an overview of related work in Section 2 and some mathematical preliminaries in Section 3, we therefore summarise the main aspects of this technique in Section 4. Section 5 presents our intrinsic meshless subdivision algorithm. Section 6 gives experimental results and comments on implementational aspects. In the concluding Section 7, we briefly remark on our ideas and some preliminary results for the theoretical analysis of intrinsic meshless subdivision schemes.

2. Related work

We start with a summary of the ideas underpinning mesh-based subdivision of surfaces in $\mathbb{R}^3$. The overview is motivational in nature, for a thorough formal treatment of mesh subdivision, see Dyn and Levin [1]. The section is concluded with remarks on recent progress in meshless, point-based surface processing related to our work.

Following the notation of Dyn and Levin [1], surface subdivision schemes consist of a subdivision operator S recursively applied to control nets $N_l = N(V^l, E^l, F^l)$ of arbitrary topology, with $l \in \mathbb{Z}_0$ denoting the subdivision level, V representing a set of control vertices in $\mathbb{R}^3$ and E and F describing the topological relations in the form of edges and faces, respectively. The iterative application of this scheme generates a sequence $N_{l+1} = SN_l$. More specifically, starting with a coarse base net N_0, at each iteration, new control vertices are inserted and connected according to the scheme's refinement rule and re-positioned following the operator's geometric averaging rule. Both the refinement and the geometric averaging rule give the position of control vertices in N_{l+1} in the form of weighted averages of topologically neighbouring vertices in N_l. The careful choice of these rules in relation to the control vertex valency, i.e. the number of edges emanating from the vertex, guarantees the convergence of the scheme, in each component and in the uniform norm, to a limit surface of provable continuity.

Not every existing mesh subdivision operator allows for this simple distinction between a topological refinement and a geometric averaging rule. However, those that do allow for this kind of distinction, include the most widely used schemes. For example, the Loop [20] subdivision scheme for triangular control nets may be cast in this form. In the case of this scheme, the refinement rule adds points related to each edge in the triangulation using face splitting. The averaging rule of points, which

depends on the vertices and edges of the non-refined triangulation, then yields the final positions of the vertices in the refined triangulation.

Point-based surface processing advocates the use of points as editing and display primitive thereby avoiding the need for a mesh connectivity graph and the manifold reconstruction step. Recent progress in this field has been sufficiently substantial to realise powerful point-based editing, free-form and multiresolution modeling [21–25] and visualisation [26–32] alternatives to mesh-based processing methods with numerous applications in medical imaging, reverse engineering, cultural heritage preservation, natural science experimentation, geoscience, numerical simulations, etc.

In this paper, we propose to replace the role of mesh connectivity in subdivision with intrinsic point neighbourhood information and formulate a set of meshless refinement and geometric averaging rules in the form of weighted geodesic centroids of these local neighbourhoods.

Fleishman et al. [33] and Guennebaud et al. [34] touch upon the notion of point cloud subdivision. In [33], the authors generate progressive level-of-details of point clouds by transferring the mesh-based idea of subdivision displacement maps to the point cloud case. They devise a point cloud simplification method for the generation of a base point set and present both a projection and a local, uniform upsampling operator with the help of local surface reconstruction using Moving Least Squares [21]. The authors successfully mimic the principle of mesh-based subdivision displacement mapping for surfaces in point cloud form but do not consider the idea of meshless subdivision.

Similarly, Guennebaud et al. [34] are concerned with the upsampling of "surfel sets", i.e. input points equipped with normal, local sampling density (surfel radius) and texture information, for magnified point rendering by splatting. Their interpolatory method requires that the underlying point cloud is regularly uniformly distributed and noise-free and that features such as crease lines have been detected and adequately sampled in a pre-processing step. Their technique is restricted in applicability to surfaces in $\mathbb{R}^3$ and intended for the operation on top of a splatting algorithm such as Zwicker et al. [31,32] providing the surfel information. As is typically the case for independently determined neighbourhoods such as the authors' polygonal fan neighbourhoods, their (extrinsic) proximity and refinement operators are affected by overlapping neighbour relations and the need for refinement rules varying with the number of neighbours of the point under consideration. This interpolatory mesh subdivision-inspired method is used for the locally smooth upsampling of surfel sets as part of a dynamic splatting algorithm, the more general notion of meshless subdivision is not considered.

The meshless surface subdivision concept makes no assumptions on the availability of normal or local density information or the regularity of the input point cloud. An intrinsic neighbourhood concept is used which is based on a partitioning of the surface and avoids the shortcomings of point neighbourhoods determined independently from each other. The approach is not restricted to surfaces in $\mathbb{R}^3$ but extends to higher dimensions. Inspired by recent work on geodesic curve subdivision by Wallner and Dyn [35–37], it offers a theoretical basis for global convergence and smoothness analysis.

In the following, we present our intrinsic framework for meshless subdivision. We start with notation and the definition of some key notions used throughout the paper. This is followed by the discussion of the intrinsic distance mapping algorithm of Mémoli and Sapiro [19] and the presentation of the framework itself.

3. Preliminaries

Let M represent a differentiable (smooth), compact and connected Riemannian submanifold in $\mathbb{R}^m$, $m \geqslant 3$. By *intrinsic* processing, we mean processing directly on this submanifold rather than in its embedding space. M is represented by a finite point set, or (unstructured) point cloud, $P = \{p_1, p_2, \ldots, p_n\} \subset M$. The Riemannian metric on M at point $x \in M$ is a smoothly varying inner product $\langle .,. \rangle$ on the tangent space T_xM. The norm of a vector v in T_xM is given by $\|v\| = \langle v, v \rangle^{\frac{1}{2}}$. We endow M with the metric inherited from $\mathbb{R}^m$, hence $\langle v,w \rangle$ will coincide with the usual inner product for vectors v and w in $\mathbb{R}^m$. Consider a (sectionally) smooth curve $\gamma : [a, b] \in \mathbb{R} \to M$ parameterised by t. The length of $\gamma(t)$ follows from integrating the norm of its tangent vectors, $\dot{\gamma}(t)$, along the curve, i.e.

$$L(\gamma) = \int_a^b \|\dot{\gamma}(t)\| \, \mathrm{d}t. \tag{1}$$

The curve γ is called the (minimising) *geodesic* from a point x to a point y in M, if it represents the minimum length-curve among all the curves on M

joining x and y. Since we are assuming M to be compact, it is geodesically complete and there exists at least one such curve on M. This curve may not be unique. The length of the geodesic between x and y gives the *intrinsic*, or *geodesic*, *distance*, $d_M(x,y)$, between the points, i.e.

$$d_M(x,y) = \inf_{\gamma}\{L(\gamma)\}, \tag{2}$$

with $\gamma(a) = x$ and $\gamma(b) = y$. The function giving the intrinsic distance from a point $x \in M$ to every point in M, $d_M(x,\cdot)$, is called the *intrinsic distance function*, or *intrinsic distance map*, of x. Unique geodesics between two points x and y on M may thus be computed from $d_M(x,\cdot)$ by backtracking from y towards x in the direction of the (negative) gradient of $d_M(x,\cdot)$.

The *extrinsic distance* between points x and y on M, $d(x,y)$, is computed in the metric of the embedding space. Since we are concerned with manifold surfaces in $\mathbb{R}^m$, the extrinsic distance is Euclidean and given by the length of the Euclidean line segment between x and y in the ambient space. Apart from its end points, this line segment generally does not lie on the manifold. More detail on the above notions may be found in, for example, Chavel [38].

By *intrinisic*, or *geodesic*, *centroid*, we understand the mean of a local neighbourhood of points on M computed in terms of intrinsic distances between the points. This is to be distinguished from the *extrinsic centroid* of the subset, computed using Euclidean distances in the ambient space and subsequently projected onto the manifold surface.

Our meshless subdivision scheme makes extensive use of the geodesic Voronoi diagram concept: Define the bisector $\mathrm{BS}(p_i,p_j)$ 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. $BS(p_i,p_j) = \{q \in M | d_M(p_i,q) = d_M(p_j,q)\}$. Let the dominance region of p_i, $D(p_i,p_j)$, denote the region of M containing p_i bounded by $\mathrm{BS}(p_i,p_j)$. The Voronoi region of p_i with respect to point set P is given by $R(p_i,P) = \bigcap_{p_j \in P, p_j \neq p_i} D(p_i,p_j)$ and consists of all points for which the geodesic distance to p_i is smaller than the geodesic distance to any other $p_j \in P$. The boundary shared by a pair of Voronoi regions is called a Voronoi edge. Voronoi edges meet at Voronoi vertices. The *geodesic Voronoi diagram* of P is defined as

$$\mathrm{VD}(P) = \bigcup_{p_i \in P} \partial R(p_i,P), \tag{3}$$

where $\partial R(p_i,P)$ denotes the boundary of $R(p_i,P)$. Kunze et al. [39] and Leibon and Letscher [40] give further details on the notion of geodesic Voronoi diagrams.

4. Intrinsic distance mapping across point clouds

To compute intrinsic distance maps, and from them (minimal) geodesics, across point clouds, we use the extension of the original Fast Marching method [41–43] to surfaces in point cloud form introduced by Mémoli and Sapiro [19]. In the following, we summarise the main aspects of this technique. For full details, see Mémoli and Sapiro [19,44].

Fast Marching generally represents a very efficient technique for the solution of front propagation problems which can be formulated as boundary value partial differential equations. Take the simple case of a front propagating across a *3*D Euclidean domain with speed, or weight, function $F(v)$ away from a source (boundary) u, with u and v representing *3*-tuples in $\mathbb{R}^3$. We are interested here in the arrival time, or offset distance, $d(u,v)$, of the front at grid point v. The relationship between the magnitude of the distance map's gradient, $\nabla d(u,v)$, and the given weight $F(v)$ at v can be expressed as the following boundary value formulation

$$|\nabla d(u,v)| = F(v), \tag{4}$$

with boundary condition $d(u,u) = 0$. The problem of determining a Euclidean weighted distance map has therefore been transformed into the problem of solving a particular type of static Hamilton–Jacobi partial differential equation, the non-linear Eikonal equation. For $F(v) > 0$, this type of equation can be solved efficiently, in computationally optimal time, using conventional Cartesian numerics, see [41–43].

Mémoli and Sapiro [19] extend the applicability of this Fast Marching idea to the case of *general co-dimension* submanifolds in point cloud form in *three or higher dimensions*. For simplicity, consider the constant radius r-offset Ω_P^r, i.e. the union of Euclidean balls with radius r centred at points $p_i \in P$ (Fig. 1)

$$\Omega_P^r := \bigcup_{i=1}^{n} B(p_i,r) = \{x \in \mathbb{R}^m : d(p_i,x) \leqslant r\}. \tag{5}$$

To approximate the weighted intrinsic distance map, $d_M(q,\cdot)$, originating from a source point $q \in M$ on M, Mémoli and Sapiro [19] suggest

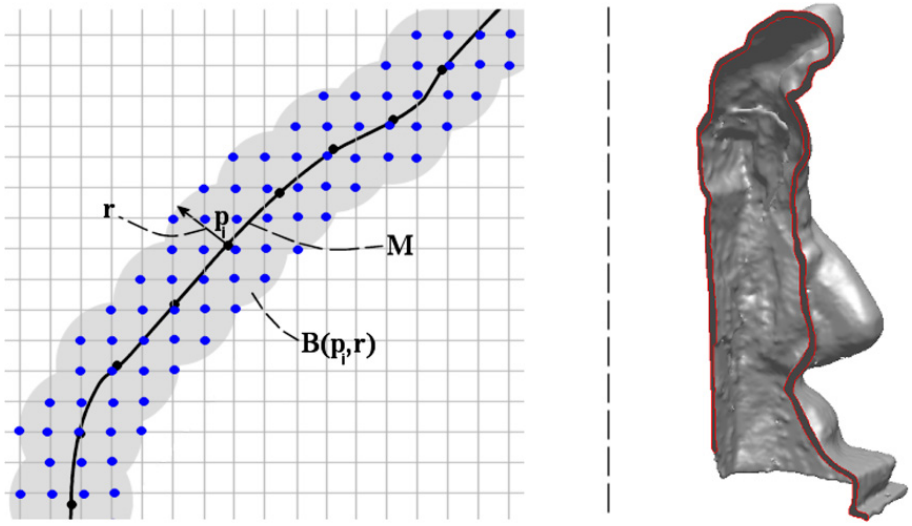

Fig. 1. Intrinsic distance mapping using Fast Marching for point clouds operates in an (not necessarily constant radius) offset band consisting of the union of balls $B(p_i,r)$ centred at (black) points p_i of the surface M (left). Only those (blue) grid points falling inside the offset band are considered during processing. A cross-sectional view of a constant radius offset band for the Michelangelo Youthful data set is shown on the right. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

computing the Euclidean distance map in Ω_P^r, denoted $d_{\Omega_P^r}$. That is

$$|\nabla_M d_M(p,\cdot)| = F(p), \tag{6}$$

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

$$|\nabla d_{\Omega_P^r}(p,\cdot)| = \tilde{F}(p), \tag{7}$$

for $p \in \Omega_P^r$ and boundary condition $d_{\Omega_P^r}(q,q) = 0$. $\tilde{F}$ represents the (smooth) extension of the propagation speed F on M into Ω_P^r; ∇_M denotes the intrinsic gradient operator. The problem of computing an intrinsic weighted distance map is therefore transformed into the problem of computing an Euclidean, or extrinsic, weighted distance map in the offset band Ω_P^r around the surface, i.e. in an Euclidean manifold with boundary.

Mémoli and Sapiro [19] prove uniform (probabilistic) convergence between these two distance maps, and geodesics computed from them, for *both noise-free and noisy* (provided noise is bounded from above by r), randomly sampled point clouds and thus show that the approximation error between the intrinsic and extrinsic distance maps is of the same theoretical order as that of the conventional Fast Marching algorithm. Fast Marching can therefore be used to approximate the solution to Eq. (7) in a *computationally optimal* manner and *without the need for any prior surface reconstruction* by only slightly modifying the conventional Cartesian Fast Marching technique to deal with bounded spaces. This is achieved by simply restricting the grid points visited by the conventional Fast Marching algorithm to those located in Ω_P^r. By performing the computations within this offset band, this method is relatively robust in the presence of noisy point samples, especially when compared to graph-based distance mapping algorithms such as [13,45] in which case the geodesics pass through the noisy samples rather than an union of Euclidean balls centred at the input points.

The complexity of this Fast Marching algorithm is $O(N \log N)$, where N represents the number of grid points located in the (narrow) offset band Ω_P^r. Memory efficiency is achieved by storing these grid points only. Note in this context that subject to the bounds given in Mémoli and Sapiro [19], r will generally be small and does not have to be constant but may vary with each p_i. Implementational details are given in Section 6.2.

A relevant question is how dense the point cloud must be as to guarantee that the error in the computation of intrinsic distances is conveniently bounded. From [19], we know that whenever $M \subset \Omega_P^r$, we have

$$\|d_M(p,\cdot) - d_{\Omega_P^r}(p,\cdot)\|_\infty \leqslant C_M r^{1/2}.$$

Now, if we choose points according to the Farthest Point Sampling strategy, i.e. the idea of repeatedly placing the next sample point in the middle of the

least-known area of the sampling domain [46], as explained in [47] Section 3.3, we also know that in order to have $M \subset \Omega_p^r$, we can enforce

$$n \geqslant \frac{\text{area}(M)}{f_M(r/2)}$$

where $f_M(s) = \min_{m \in M} \text{area}\ (B_M(m,s))$, where $B_M(m,s)$ represents the open geodesic ball on M centred at m with radius s. By combining these two bounds, we can answer the question above: Let δ be a desired upper bound for the approximation error in the distances, then, in order to have

$$\|d_M(p,\cdot) - d_{\Omega_p^r}(p,\cdot)\|_\infty \leqslant \delta$$

we can choose $r = (\frac{\delta}{C_M})^2$ and $n \geqslant \frac{\text{area}(M)}{f_M(r/2)}$. Finally, by using the Bishop–Günther theorem, one can relate $f_M(\cdot)$ to the sectional curvatures of M, see [47] Section 3.2 for more details.

This Fast Marching technique underpins our geodesic computation algorithm presented as part of our intrinsic meshless subdivision framework in the following section.

5. Intrinsic meshless subdivision

Subdivision schemes incorporate refinement and geometric averaging rules in the form of weighted averages of local neighbourhoods. Mesh-based subdivision schemes are derived in a parametric setting ignoring the geometric embedding of the points in space. As a result, they are formulated in terms of local mesh connectivity rather than object geometry. We determine local neighbourhood relations from intrinsic point proximity information instead.

We start this section with the discussion of a suitable intrinsic neighbourhood concept. This is followed by the presentation of our meshless subdivision scheme and a new method for the computation of geodesic centroids, which is at the heart of this scheme.

5.1. Intrinsic point proximity information

Depending on the acquisition technique used, point-sampled geometry may feature (locally) non-uniform point distributions. In this setting, naive neighbourhood concepts such as simple ball or k nearest neighbourhoods tend to be skewed, i.e. the neighbours $q_j \in P_l$ are frequently no longer distributed spherically, i.e. all around, the point $p \in P_l$ under consideration [22,48].

To allow for local sampling non-uniformities, Linsen's [22] enhanced k nearest neighbourhood, which enforces a maximum angle between successive neighbours around p, or Floater and Reimer's [48] local Delaunay neighbourhood may be used. Their algorithm projects a local ball neighbourhood of p and p itself into their least squares plane and computes the planar Delaunay triangulation of the projected points. The points' neighbour relations in this triangulation are then taken as the neighbour relations of p on the manifold. We suggest to collect local proximity information by considering the set of (Voronoi) neighbours, N_p, of p in the geodesic Voronoi diagram of P_l, $\text{VD}(P_l)$, instead, i.e. an *intrinsic* "natural neighbourhood" [49],

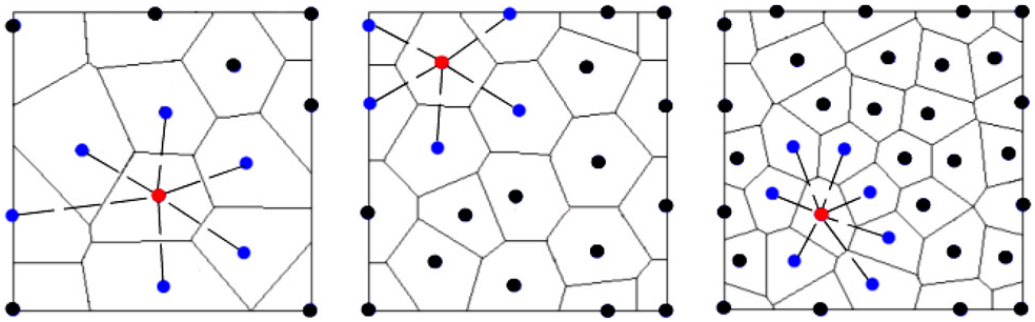

Fig. 2. As illustrated here for the planar case, the intrinsic Voronoi diagrams (solid lines) of various moderately irregularly distributed point sets provide point proximity relations in the form of *intrinsic* natural neighbourhoods (blue points) spherically distributed around the (red) point under consideration. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

$$N_p = \{q : p \text{ and } q \text{ are neighbours in } \mathrm{VD}(P^l)\},$$

for $p, q \in P_l$, $p \neq q$ (Fig. 2). Due to its intrinsic nature and unlike (Euclidean) ball neighbourhoods, points from disjoint parts of a surface are prevented from being assigned to the same natural neighbourhood. The need for determining local ball radii and the problem of locally ill-defined least squares fits in the case of k nearest neighbourhoods do not arise. Since intrinsic natural neighbourhoods are derived from a partitioning of the surface, the problem of overlapping neighbour relations is avoided as well. This problem is typically encountered when using point neighbourhoods computed independently from each other such as k nearest neighbours and its variations [22,34] and occurs irrespective of the regularity of the point set distribution. Intrinsic natural neighbourhoods are, however, less straightforward to compute. We discuss our approach in detail in Section 6.

We use intrinsic natural neighbourhood information for the computation of local geodesic centroids as part of our meshless subdivision scheme presented next.

5.2. An intrinsic meshless subdivision scheme

Within our meshless geometric subdivision framework, geodesic centroids of the intrinsic neighbourhoods discussed in the previous section are used to define meshless subdivision rules. More specifically, we propose the following set of rules to be applied at each iteration:

Refinement rule. For each neighbour $q_j \in N_p$, consider the union of intrinsic neighbours of p, $q_j \in P_l$, N_{pq_j}. Upsample P_l by inserting the weighted geodesic centroid, $c(N_{pq_j}) \in P_{l+1}$, of N_{pq_j}.

Geometric averaging rule. Replace $p \in P_l$ by the weighted geodesic centroid, $c(N_p) \in P_{l+1}$, of its intrinsic Voronoi neighbourhood N_p.

This use of weighted centroids in the refinement and geometric averaging rules is reminiscent of both classical subdivision schemes [2] (Section 2) and the "repeated averaging" approach towards the generation of subdivision surfaces ([50,51] and references therein). These subdivision rules are incorporated into our meshless subdivision algorithm as summarised in Algorithm 1.

Algorithm 1. One iteration of meshless subdivision in pseudocode.

Input: Point cloud $P_l \in \mathbb{R}^m$.
Output: Subdivided point cloud P_{l+1}.

```
0   *** INITIALISATION ***
1   Bucket the base point cloud P_l in a m-
    dimensional Cartesian grid;
2   Compute the discrete geodesic Voronoi
    diagram, VD(P_l), of P_l.
3
4   *** MAIN PROCESSING ***
5   FOR each point p_i ∈ P_l;
6     Determine the Voronoi neighbourhood N_{p_i}
      from VD(P_l);
7     FOR each neighbour q_j ∈ N_{p_i} ;
8       Determine the joint Voronoi
        neighbourhood N_{p_i q_j} from VD(P_l);
9       Compute the weighted geodesic centroid
        c(N_{p_i q_j});
10      (Refinement rule) Upsample P_l to P_{l+1} by
        inserting c(N_{p_i q_j}) ;
11    ENDFOR
12  ENDFOR
13  Compute VD(P_{l+1}) ;
14  FOR each point p_i ∈ P_{l+1} ;
15    Determine the Voronoi neighbourhood N_{p_i}
      from VD(P_{l+1});
16    Compute the weighted geodesic centroid
      c(N_{p_i}) ;
17    (Geometric averaging rule) Replace p_i in P_{l+1}
      with c(N_{p_i});
18  ENDFOR
```

In its initialisation phase, the algorithm buckets the input point set P_l in a Cartesian grid, subsequently used to support weighted distance mapping and geodesic centroid computation. Initialisation is completed with the computation of the (discrete) geodesic Voronoi diagram of P_l, $\mathrm{VD}(P_l)$. Main processing loops over all points in P_l and proceeds with the upsampling of P_l to P_{l+1} using joint neighbourhood information readily available from $\mathrm{VD}(P_l)$ and our refinement rule. Following this refinement step, $\mathrm{VD}(P_{l+1})$ is computed. Meshless subdivision is concluded with the application of our geometric averaging rule to each point in the refined point set P_{l+1} yielding the final, subdivided point set.

The applicability of this subdivision algorithm does not depend on a preceding simplification step.

Potential algorithm applications, however, include the case in which it may be desirable to simplify an excessively dense point cloud P to a suitable base point set P_0 in the expectation that recursive subdivision of P_0 results in a smoother, more regular and more compact approximation of the underlying surface than given by P. Since our subdivision algorithm requires the computation of geodesic centroids across the base point set P_0, for these centroids to be well defined, any simplification of P needs to be performed subject to a minimum point density in P_0. Due to the method's simple control of a guaranteed point density, its purely intrinsic operation, its close relationship with the natural neighbourhood concept of Section 5.1 and its efficient implementability using the geodesic distance mapping method of Mémoli and Sapiro [19] (Section 4), we use the algorithm of Moenning and Dodgson [46] for the simplification of an input point cloud P to a base point set P_0 still sufficiently dense to support the computation of geodesic centroids. As another result of this pre-processing step, the (discrete) geodesic Voronoi diagram of P_0 becomes available [46] so that it does not need to be computed in our subdivision algorithm's initialisation phase and the natural neighbours of a point $p_i \in P_0$ are readily known.

By performing the averaging intrinsically, our meshless subdivision rules raise the questions of how to compute geodesic centroids on manifolds and how to determine a suitable neighbour weighting scheme. In this paper, we guide the choice of weights by experimental results (Section 6) rather than theoretical evidence for the scheme's convergence towards a smooth limit surface. Future work will consider formal proofs of the scheme's convergence to a limit surface and, consequently, any light such proofs may throw on the optimal choice of weights.

As regards the computation of geodesic centroids, Buss and Fillmore [52] present an algorithm for the computation of geodesic averages on spheres. We generalise the underlying, earlier idea ([53] and references therein) of minimising a least squares expression in geodesic distances in the following section.

5.3. Geodesic centroid computation

The benefit of performing the averaging intrinsically is that it ensures that subdivision generates smoother, denser representations which remain geometrically close to the surface. This is not guaranteed to be the case when considering Euclidean instead of geodesic centroids. For the simple example illustrated in Fig. 3, Euclidean averaging ignores the non-linear, intrinsic geometry of the object and moves the centroid away from the surface. By contrast, since the computation of the geodesic centroid is based on intrinsic rather than Euclidean distances, it is inherently geometry-sensitive and falls onto the surface in each case.

The weighted geodesic centroid of a set of n points is defined as the point $g \in M$ which minimises the weighted sum of squared intrinsic distances to each point

$$J(g) := \frac{1}{2}\sum_{k=1}^{n} w_k d_M^2(g, p_k), \tag{8}$$

where $w_1, \ldots, w_n$ represent the point weights, with $0 \leqslant w_i \leqslant 1$, $\sum_{i=1}^{n} w_i = 1$. In general, $\operatorname{argmin}_g J(\cdot)$

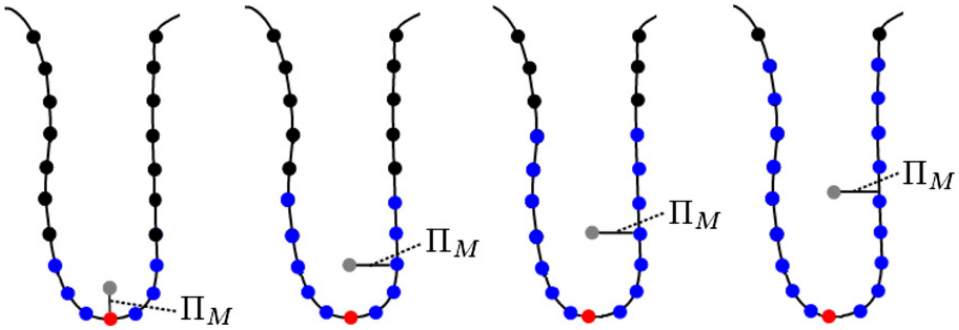

Fig. 3. The unweighted centroid of a (blue) subset of this set of points is expected to be located on or near the underlying surface. Due to the use of intrinsic distances, this is the case when computing the geodesic centroid (red). By contrast, in the case of the Euclidean averaging of the (blue) points, the resulting centroid (grey) is located away from the underlying surface. This effect gets more pronounced when increasing the size of the subset (from left to right). Note that for *geodesically close* neighbourhoods and those kinds of neighbourhoods only, the *orthogonally projected* ($\Pi_M : \Omega_M^r \to M$) Euclidean average, i.e. the extrinsic mean, generally provides a good (first) estimate of the geodesic centroid (leftmost). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

may not exist or may not be a single point. However, if $p_1,\ldots,p_n$ are all contained in a sufficiently small open geodesic ball B_M, a unique solution, g_{B_M} of $J(\cdot)$, which happens to lie in B_M [53], is guaranteed. The property we are alluding to here is (geodesic) convexity, i.e. for any $p_i, p_j \in B_M$, the minimal geodesic from p_i to p_j is unique in M and contained in B_M.

In the Euclidean case, direct differentiation of $J(\cdot)$ yields the minimiser $g = \sum_{k=1}^{n} w_k p_k$. This simple result does not extend to the general case considered here but we can prove that any minimiser must satisfy:

$$V(g) := \sum_{k=1}^{n} w_k \nabla_M \frac{1}{2} d_M^2(g, p_k) = 0. \tag{9}$$

Then, starting from a good initial guess g_0, we can track the minimiser g using back propagation with velocity field $V(\cdot)$. This is due to the fact that if $g_0 \in B_M$ and B_M as above, then $-V(x)$ points towards g_{B_M}, for $x \in B_M$ [53].

In practise, we set $g_0 = \Pi_M(\sum_{k=1}^{n} w_k p_k)$, where $\Pi_M : \Omega_M^r \to M$ is the orthogonal projection operator. To start from any of the points p_k represents another simple choice. We now show that in the light of the considerations presented above, this extrinsic mean represents either a reasonable initial condition for the back propagation or a first approximation to the intrinsic centroid. By a simple application of Lemma 17 of [35], we have that

$$\left\| \sum_{k=1}^{n} w_k p_k - \Pi_M\left(\sum_{k=1}^{n} w_k p_k\right) \right\| \leqslant C(\mathrm{diam}(B))^2, \tag{10}$$

where C is a global constant which depends on the curvatures of M. Then, let $B = B_M(x, \epsilon)$, for some $x \in M$ and $\epsilon > 0$. Since $\|p_i - x\| \leqslant d_M(p_i, x) \leqslant \epsilon$, we also have $\|\sum_{k=1}^{n} w_k p_k - x\| \leqslant \epsilon$. Therefore, since $\|g_0 - x\| \leqslant \|g_0 - \sum_{k=1}^{n} w_k p_k\| + \|\sum_{k=1}^{n} w_k p_k - x\|$, we obtain

$$\|g_0 - x\| \leqslant C\epsilon^2 + \epsilon, \tag{11}$$

which implies $d_M(g_0, x) \leqslant \epsilon(1 + Q\epsilon)(1 + C\epsilon)$, for another constant Q depending on global metric properties of M [19]. We only care for a simplified bound

$$d_M(g_0, x) \leqslant E\epsilon. \tag{12}$$

Finally, let $\overline{\delta} > 0$ be the maximal $\delta > 0$ such that $B_M(x, \delta)$ is (geodesically) convex. Note that it is a fact that if $\delta \leqslant \frac{1}{2}\min(\mathrm{inj}(M), \frac{\pi}{\sqrt{K}})$, where $\mathrm{inj}(M)$ is the injectivity radius of M and K bounds all sectional curvatures in M from above, then $B_M(x, \delta)$ is convex for any $x \in M$. See Sections 7.6 and 7.7 in Chavel [38]. For such a $\overline{\delta} > 0$ and provided $\epsilon \leqslant \overline{\delta}/E$, and $\{p_1,\ldots,p_n\} \subset B_M(x, \epsilon)$ for some $x \in M$, $g_0 \in B_M(x, \overline{\delta})$ and $-V(g_0)$ will be pointing towards g_{B_M}. Also, in case we want to use g_0 as an approximation to g_{B_M}, we have the (weak) bound $d_M(g_{B_M}, g_0) \leqslant (E + 1)\epsilon$. Therefore, g_0, as defined above, represents a sensible choice as the initial condition of an eventual back propagation step, or, in any case, a rough approximation to g_{B_M} with known error bound. See also Fig. 3 (left). Note in particular that it is also a useful choice from the point of view of computational ease. The algorithm is summarised in Algorithm 2.

Algorithm 2. Procedure for computation of a weighted geodesic centroid in pseudocode.

Input: Intrinsic Voronoi neighbourhood N_p of point $p \in P_l$. Weights w_i at points $q_i \in N_p$.
Output Weighted geodesic centroid g.

```
0  *** Computation of extrinsic centroid g_0 ***
1     Compute the Euclidean weighted centroid of N_p;
2     Compute g_0 by orthogonally projecting the
      weighted Euclidean centroid;
3
4  *** Computation of intrinsic centroid g ***
5     Compute local weighted distance maps
      d_{Ω_p^r}(q_i, ·) from each neighbour
      q_i ∈ N_p outwards and accumulate their
      squared values at the grid vertices;
6     Approximate the gradient of the accumulated
      distance maps using finite
      difference approximation;
7     Back propagate from g_0 towards g by
      following the negative gradient;
```

To demonstrate the applicability of this approach in the context of meshless subdivision, we first consider the case of M representing the unit sphere in the following section.

6. Experimental results and implementational details

We begin with the intrinsic meshless subdivision of a set of points sampled relatively regularly uniformly from the surface of the unit sphere in $\mathbb{R}^3$. This initial restriction to spherical geometry allows for the computation of precise geodesic distances

without the need for numerical techniques. This way qualitative and quantitative aspects of our operator can be presented without the influence of the particular projection and gradient descent techniques utilised when processing more complex geometry. This presentation is followed by applications of our subdivision operator to more complex geometry. The section concludes with comments on some implementational aspects.

6.1. Results and discussion

To implement the intrinsic centroid computation method, techniques for the computation of intrinsic distances between points on the surface, the projection of the starting point for the back propagation onto the surface and the computation of the back propagation itself are required. In the case of the unit sphere, these techniques are readily available and no numerical techniques are required. Intrinsic distances between points follow trigonometrically and orthogonal projection is trivial. Similarly, the exponential map and its inverse are directly available and may be used to implement the back propagation procedure. As a result, for the case of spherical geometry, our approach for geodesic centroid computation narrows down to the technique of Buss and Fillmore [52].

We apply our intrinsic meshless subdivision operator to a base point set P_0 of 2144 points sampled relatively regularly uniformly from the unit sphere (Fig. 5). The application of our subdivision operator to P_0 using the initial intrinsic proximity information from $\mathrm{VD}(P_0)$ yields the subdivided point set P_1 of Fig. 6. The result, P_2, obtained from the application of the operator to P_1 using natural neighbourhood information from $\mathrm{VD}(P_1)$ is shown in Fig. 7. Given the relatively strong regularity of the input data, uniform weighting was used for both the refinement and the geometric averaging rule in both iterations. The results produced by our meshless subdivision operator are presented alongside the point sets produced by the application of Loop subdivision to a triangular mesh representation of

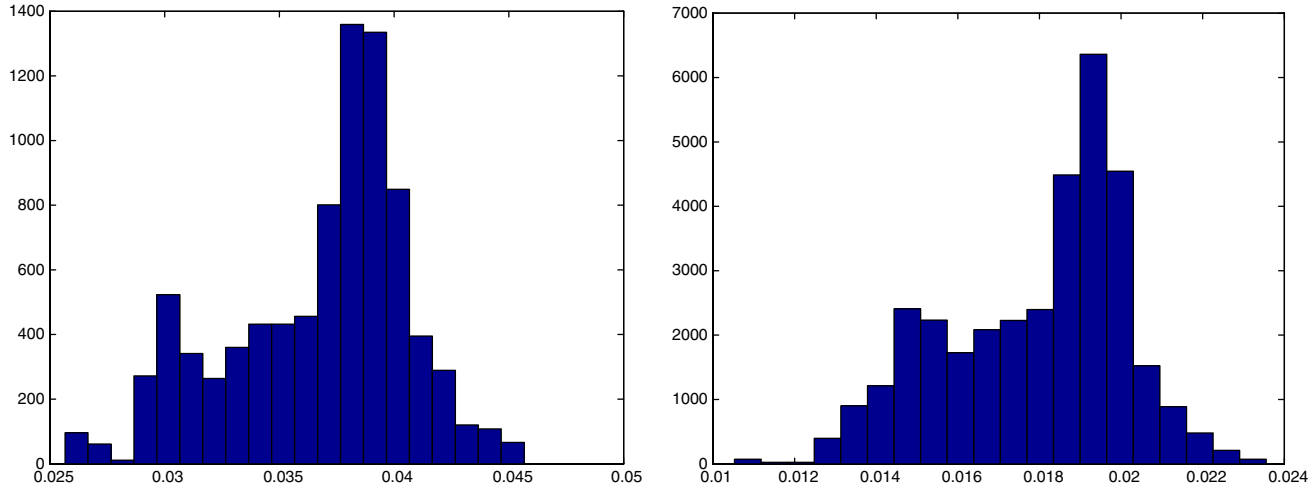

Fig. 4. Histograms of the (spherical) distance from each point in P_1 (left) and P_2 (right) to its closest neighbour in the respective set.

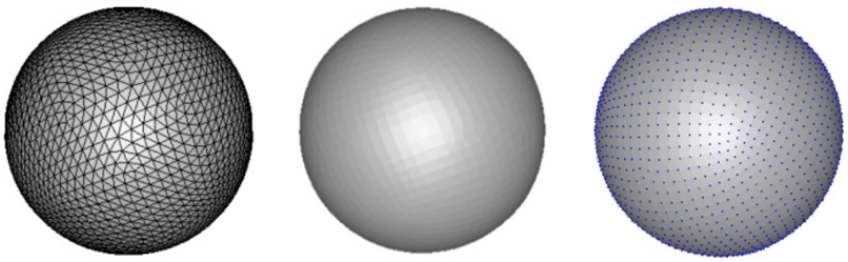

Fig. 5. Relatively regularly uniformly distributed base point set, P_0, of 2144 points acquired from the unit sphere (right). The triangular base mesh generated from P_0 for the support of Loop subdivision is shown on the left. A flatly shaded rendering of the reconstructed surface is shown in the centre.

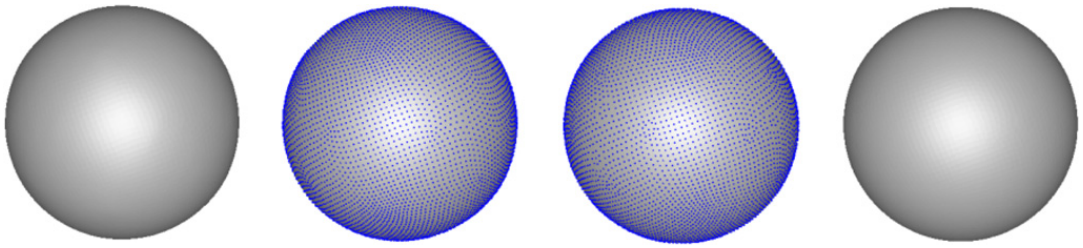

Fig. 6. Results after one iteration of Loop (left) and meshless subdivision (right); $|P_1| = 8570$ points in both cases. Flatly shaded renderings of the reconstructed surfaces are shown next to each point set.

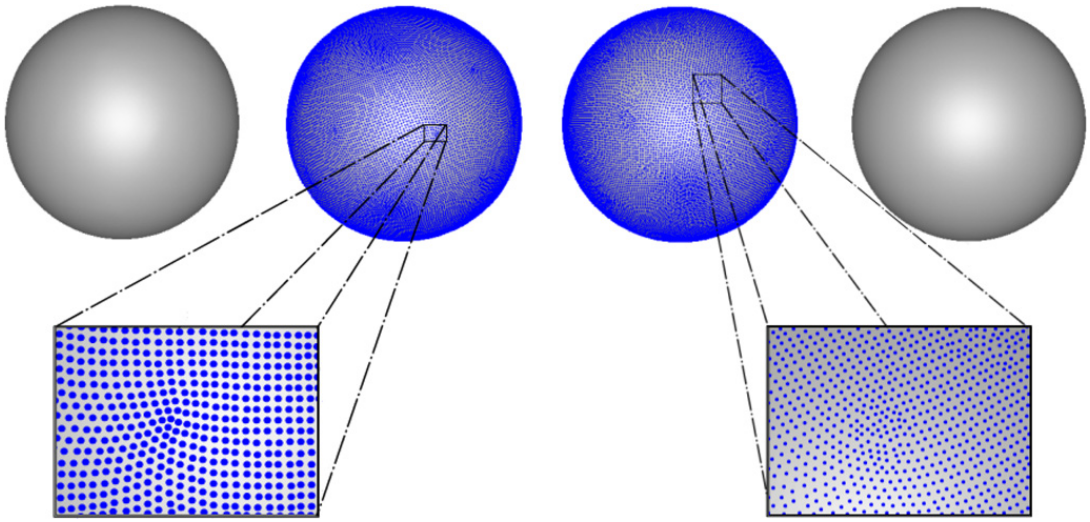

Fig. 7. Results after the second iteration of Loop (left; 34,275 points) and meshless subdivision (right; 36,442 points). The corresponding reconstructed surfaces are given next to each point set. The detail views indicate how the slight irregularities in the distribution of P_0 lead to slightly more pronounced local irregularities in the case of meshless subdivision due to the use of uniform weighting. These slightly more pronounced irregularities do not have any noticeable effect on the smoothness of the surface.

P_0. As indicated by the detail views of Fig. 7, the point distributions obtained from the two operators after two iterations are qualitatively similar with the slight irregularities in the distribution of P_2 being slightly more pronounced in the case of meshless subdivision due to the use of uniform weights. There are, however, no noticeable differences in the smoothing effect of these two operators.

In order to analyse the point set distributions generated by our subdivision operator quantitatively, we compute the mean and the standard deviation of the distance from each point in the set to its closest neighbour(s) for the subdivided point sets P_1 and P_2. For each p in the point set P_l, let $\mathrm{sd}_k(p)$ denote the (spherical) distance from p to its kth closest neighbour. As an indicator for the uniformity of the density of point set P_l, consider $\rho(k) = \frac{\min_P \mathrm{sd}_k(p)}{\max_P \mathrm{sd}_k(p)}$. Since $\rho(k)$ represents an absolute measure, it may be too sensitive, therefore we compute instead $\widehat{\rho(k)} = \frac{\mathrm{mean}(\mathrm{sd}_k) - \mathrm{std}(\mathrm{sd}_k)}{\mathrm{mean}(\mathrm{sd}_k) + \mathrm{std}(\mathrm{sd}_k)}$, where mean and std stand for the mean and standard deviation of the spherical distances over the point set, respectively.

The histograms of $\mathrm{sd}_1(x)$ corresponding to the two sets of points are given in Fig. 4. Note in particular in Table 1 that the values of $\widehat{\rho(k)}$, for $1 \leqslant k \leqslant 10$, are quite close to *1.0* therefore indicating small dispersion up to the *10*th closest neighbour.

Table 1
Values of the density uniformity measure $\widehat{\rho(k)}$ for P_1 and P_2 and with $k \in \{1, 2, \ldots, 10\}$

Model\k	1	2	3	4	5	6	7	8	9	10
P_1	0.807	0.832	0.832	0.821	0.788	0.809	0.8132	0.786	0.818	0.828
P_2	0.781	0.815	0.823	0.804	0.785	0.799	0.7980	0.784	0.810	0.822

The values underline the regular uniformity of the subdivided point sets generated by our algorithm.

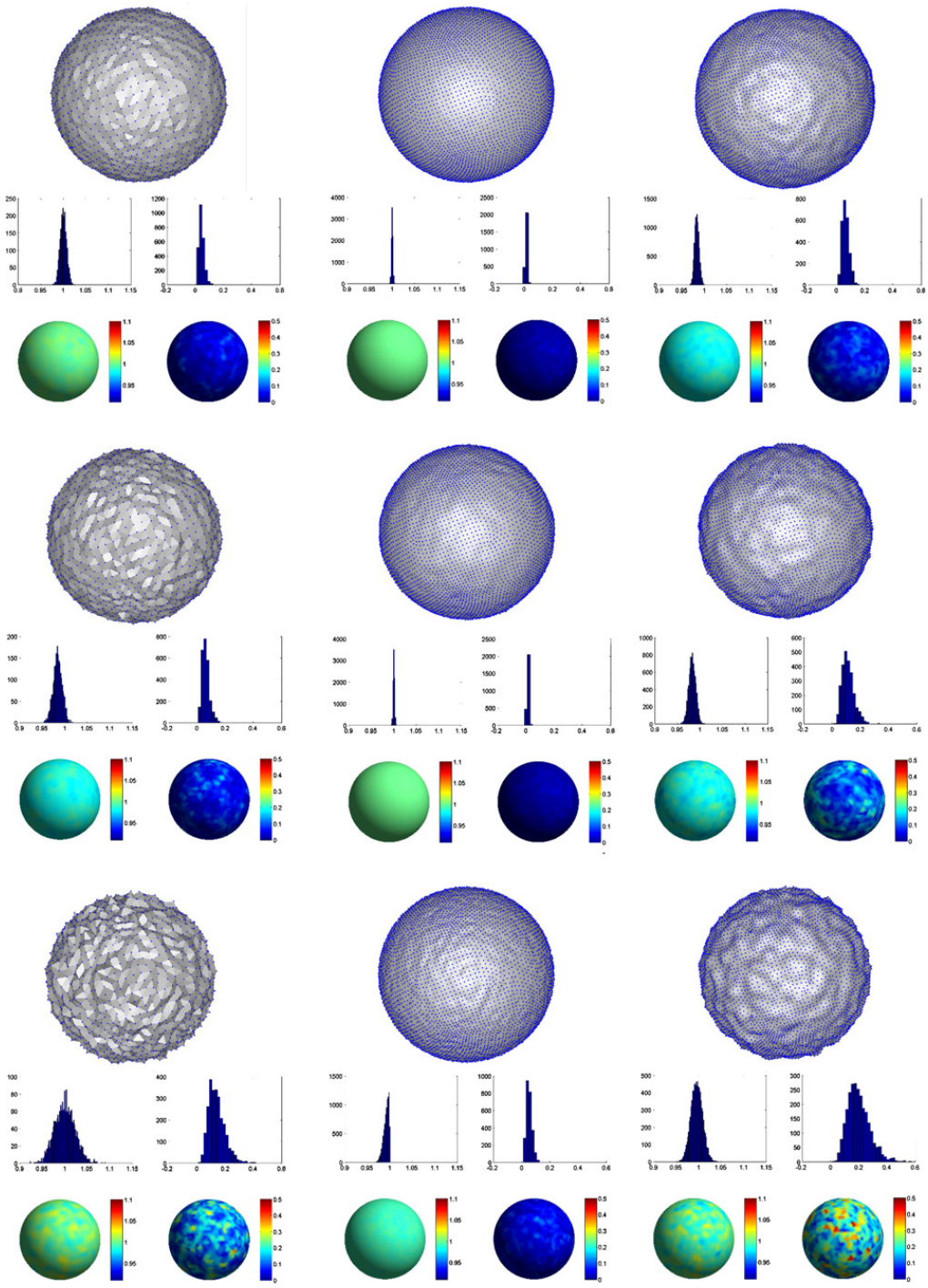

Fig. 8. One iteration of meshless (centre) and Loop subdivision (right) respectively of input point sets (left; $|P_0| = 2144$) exhibiting (from top to bottom) increasing levels of normally distributed noise along the points' normals. Meshless subdivision produces visibly smoother results than mesh-based Loop subdivision ($|P_1| = 8570$ in each case). This visual impression is confirmed by the analytical results presented beneath each graphical representation: a histogram of the values of the height function h in the top-left corner, a histogram of the values of the discretely approximated norm of h, $\|\mathrm{Dh}\|$, (for $k = 10$) in the top-right corner, the sphere colored with the values of h in the bottom-left corner and, finally, the sphere colored with values of $\|\mathrm{Dh}\|$ in the bottom-right corner. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

Fig. 8 illustrates experimental results from both the meshless and the Loop subdivision of input point sets exhibiting increasing levels of normally distributed noise in the points' normals. More specifically, the unit normals were exposed to zero-mean noise with variances of 0.005 (L, low), 0.015 (M, medium) and 0.05 (H, high), respectively. Due to its meshless nature, our algorithm proves less sensitive to these

distortions and generates noticeably smoother results than those produced by Loop subdivision.

Apart from this visual inspection, the subdivided point sets were analysed more thoroughly as follows. For each of the *9* models X_i, $i = 1,\ldots,9$ (*3* noisy input spheres, *3* meshless subdivision outputs and *3* Loop subdivision outputs), we computed the *height function* h: $X_i \to R$ measuring the distance of each point in the set to the origin. The values of this function can be expected to be distributed around *1* for the point sets resulting from the application of a *reasonable* subdivision algorithm. Also, in order to quantify the spatial variation of h due to a particular algorithm, we computed a discrete approximation to the (norm of the) gradient of h. More specifically, let k be a positive integer and $x \in X_i$, then we compute

$$\|\mathrm{Dh}(x)\| \simeq \max_{j=1,\ldots,k} \frac{|h(x) - h(x_j)|}{\|x - x_j\|},$$

where the x_j represent the k nearest neighbours of point x (among points in X_i). Fig. 8 provides for each data set, a rendering of the point set across the reconstructed surface plus a panel with *4* additional figures: A histogram of the values of h in the top-left corner, a histogram of the values of $\|\mathrm{Dh}\|$ (for $k = 10$) in the top-right corner, the sphere colored with the values of h in the bottom-left corner and finally, the sphere colored with values of $\|\mathrm{Dh}\|$ in the bottom-right corner. In addition, Tables 2 and 3 show the mean and standard deviation of h and $\|\mathrm{Dh}\|$ for each level of noise and the results of both Loop and meshless subdivision of the noisy input point sets. All of these analytical results confirm the conclusion obtained from the visual inspection of the subdivided point sets.

Fig. 9 presents an application example dealing with more complex geometry. We apply our meshless subdivision operator to a base point set of 10088 points generated from the Michelangelo Youthful data set with the help of Moenning and Dodgson [46]. Experimentation revealed the base point set to be regular enough to allow for simple reciprocal distance weighting in the computations of the weighted geodesic centroids. The flatly shaded renderings of the surfaces reconstructed from the subdivided point sets P_1 and P_2 illustrate the smoothing effect of the meshless subdivision. As indicated by the comparative illustrations in Fig. 10, this approach may be used to obtain a smoother, more regular and more compact representation of an highly dense point cloud. A similar effect is shown in Fig. 11. The 50% decimated versions of the rocker arm and screwdriver CAD data sets available from the Cyberware website were meshlessly subdivided twice, with the legal cavity of the rocker arm data set being maintained correctly by the algorithm. The smoothing effect of the iterations is again clearly apparent when comparing the surfaces reconstructed from the subdivided point sets to those reconstructed from P_0 and the non-simplified, non-subdivided data sets respectively.

Table 2
Mean and standard deviation of the values of h

Noise level	Noisy dataset		Loop		Meshless	
L	1.001	0.007	0.983	0.005	1	0.002
M	0.985	0.011	0.983	0.007	1	0.002
H	1.002	0.022	0.996	0.013	0.99	0.006

Table 3
Mean and standard deviation of the values of $\|\mathrm{Dh}\|$

Noise level	Noisy dataset		Loop		Meshless	
L	0.044	0.018	0.065	0.024	0.014	0.006
M	0.063	0.026	0.113	0.043	0.015	0.006
H	0.14	0.058	0.199	0.079	0.051	0.029

When dealing with non-uniformly distributed point sets including illegal cavities, meshless geometric subdivision remains unchanged provided an adaptive offset band $\Omega_P^{r_i}$ can be fitted during preprocessing. This is prone to failure when dealing with particularly pronounced non-uniformities. Similarly, when processing point sets affected by noise levels exceeding the offset band radius r, the algorithm will no longer converge and fail (Section 4). Note in this context that geodesic distance computation cannot be meaningfully performed in the case of excessive non-uniformities irrespective of the algorithm used. To overcome both this problem and the problem of excessively high levels of noise, the input point set could be resampled in a pre-processing step using the resampling or hole-filling techniques of, for example, Weyrich et al. [54].

The detail view of Fig. 9 highlights the local clustering effect caused by overlapping neighbour relations as discussed in Section 5.1, an effect typically only encountered with point neighbourhoods which determine the neighbour relations independently of each other. Our discrete approximation of the intrinsic Voronoi diagram, however, implies discretisation error and thus individual instances of a point being assigned to the wrong Voronoi region. The generally limited number of these instances is considered preferable to the complications associated

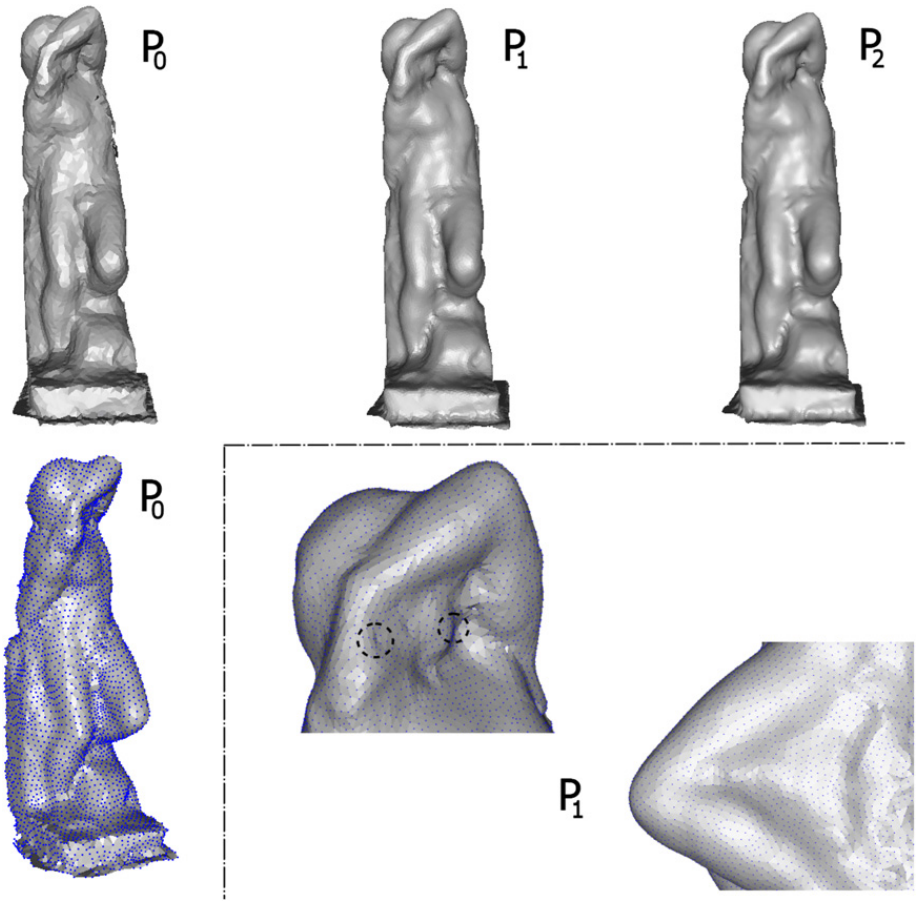

Fig. 9. Flatly shaded renderings of the surfaces reconstructed from point sets $|P_0| = 10088$ (top and bottom left), $|P_1| = 39888$ (centre and bottom right) and $|P_2| = 208010$. The smoothing effect of meshless subdivision is clearly visible. As illustrated by the detail front and side views, the subdivided point sets are distributed relatively regularly uniformly. Instances of local irregularities (encircled in black) are caused by discretisation errors during discrete intrinsic Voronoi diagram computation.

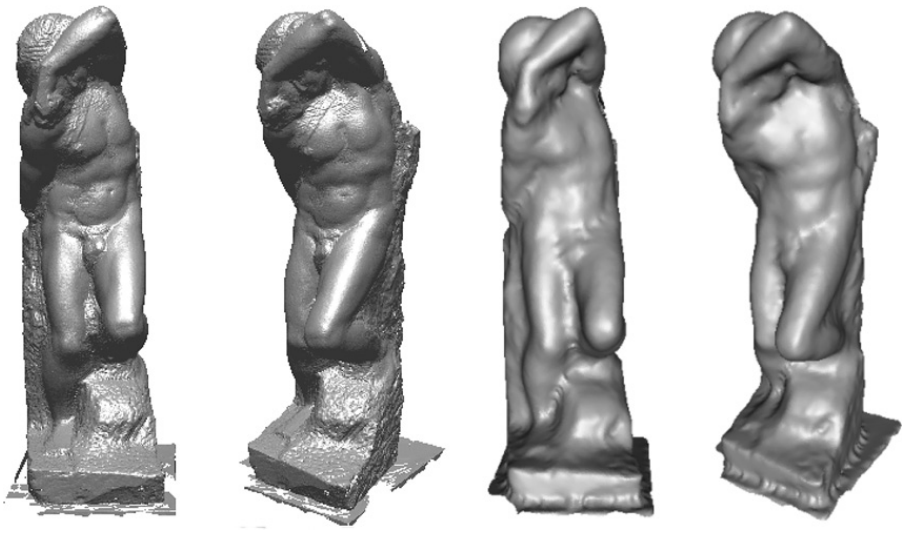

Fig. 10. Flatly shaded renderings of the surfaces reconstructed from the non-simplified Youthful point set (left) and P_2 of Figure 9 illustrating how meshless subdivision of a base point set resulting from point cloud simplification may be used to obtain a smoother, more regular and more compact representation of the original data set of 1,728,305 points.

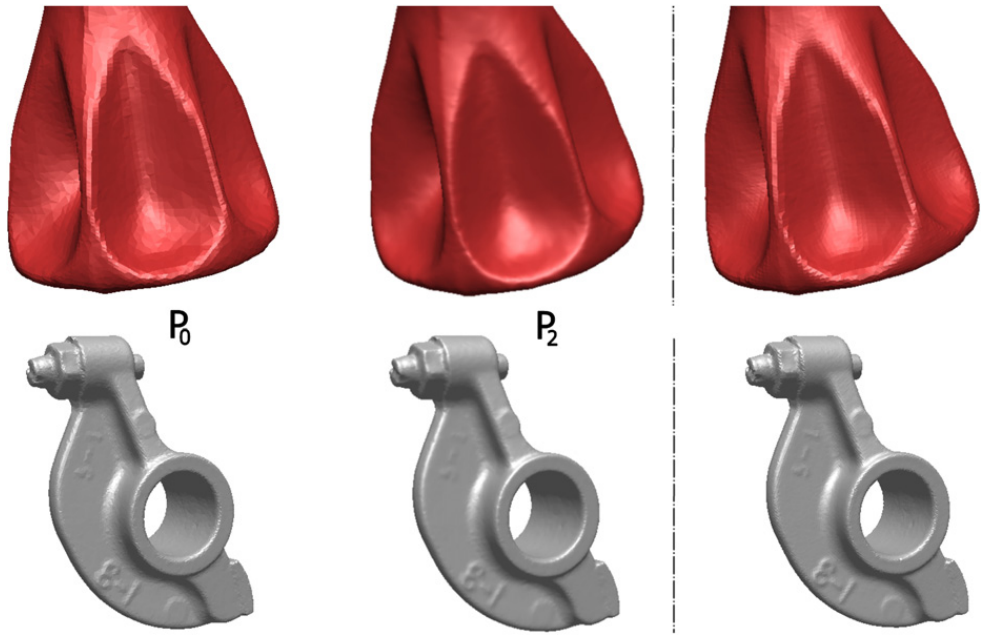

Fig. 11. On the left, the flatly shaded renderings of surfaces reconstructed from the 50% decimated screwdriver (top) and rocker arm (bottom) CAD data sets (P_0) are shown. The reconstructed surfaces obtained from two iterations of meshless subdivision of these point sets are presented in the centre (P_2). The surfaces reconstructed from the non-simplified, non-subdivided data sets are given on the right. The smoothing effect of meshless subdivision is again clearly visible.

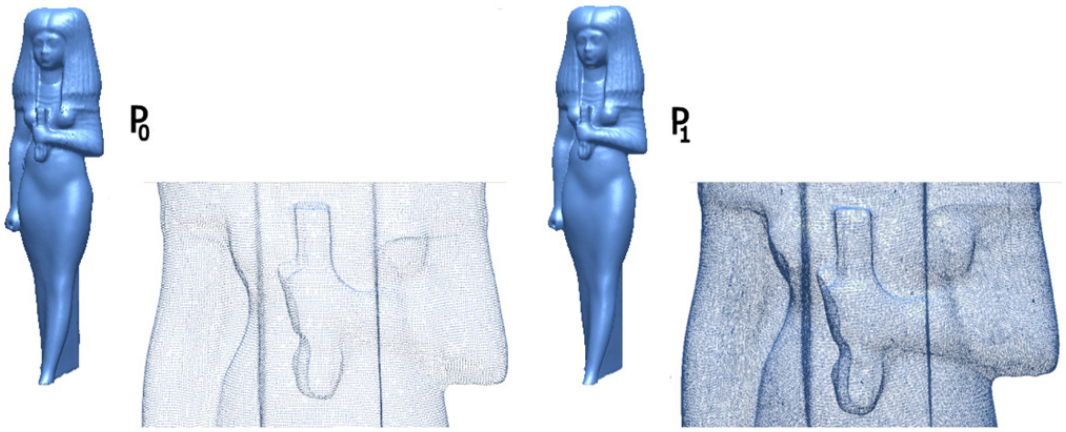

Fig. 12. Point set detail views and smoothly shaded renderings of the Isis data set (left), P_0, $|P_0| = 187{,}644$, and the subdivided point set P_1 (right), $|P_1| = 760{,}162$, resulting from one iteration of meshless subdivision. Due to the high density of P_0, the difference in location between the (uniformly weighted) extrinsic and the corresponding intrinsic centroid was found to be negligible and extrinsic means were used throughout.

with addressing the problem of overlapping neighbour relations (see, e.g. Guennebaud et al. [34]) when using alternative neighbourhood concepts.

The limited impact of these assignment errors is illustrated by the detail view of Fig. 12. The Isis data set was subdivided once with the regularity of the subdivided point set being only mildly affected by erroneous neighbourhood assignments. In contrast to the processing of the Youthful base point set, meshless subdivision of the Isis point cloud using geodesic centroids was found to yield results not significantly different from the more efficient subdivision by extrinsic centroids. We exploit this observation, due to the high density of the initial point set, by presenting the results from orthogonal projection of the uniformly weighted Euclidean centroids, i.e. without subsequent gradient descent towards their geodesic counterparts.

Table 4 summarises the parameter settings and point set sizes for the various application examples. Using our non-optimised implementation, the offset band computation pre-processing step and a mesh-

Table 4
Parameter settings and point set sizes for the application examples; Δx, Δy, Δz refer to the grid spacing in the three principal Cartesian grid directions; r represents the constant offset band radius

	$\Delta x(=\Delta y=\Delta z)$	r	$\lvert P_0\rvert$	$\lvert P_1\rvert$
Iteration 1				
Sphere	0.25	0.8	2144	8570
Youthful	1.0	2.2	10080	39888
Screwdriver	0.5	1.0	13577	56220
Rocker arm	1.0	1.9	20088	81442
Isis	0.1	0.2	187644	760162
	$\Delta x(=\Delta y=\Delta z)$	r	$\lvert P_1\rvert$	$\lvert P_2\rvert$
Iteration 2				
Sphere	0.25	0.75	8570	34275
Youthful	0.25	0.75	39888	208010
Screwdriver	0.25	0.6	56220	295110
Rocker arm	0.25	0.45	81442	488212

less subdivision iteration took a maximum of a few hundred seconds each. Both offset band and intrinsic Voronoi diagram computation efficiency generally depend strongly on the offset band radius r and the grid spacing, our settings of which are presented in the table. Note in this context that the numerical error inherent in the intrinsic distance computations increases with r, the admissible minimum value of which necessarily increases with the grid spacing [44]. Details on various aspects of the implementation of our algorithm are provided in the following section.

6.2. Implementational details

The algorithm was implemented in C++ (Microsoft Visual C++ 7.1) with the help of the "Blitz++ 0.7 Numerical Library" [55] on a Pentium 4 2.8 GHz, 512MB Windows XP machine. In the following, we summarise relevant implementational details.

6.2.1. Offset band computation and intrinsic distance mapping

Intrinsic distance mapping requires, first, the computation of the offset band Ω_P^r and, second, weighted Euclidean distance mapping within the offset band. We meet both of these requirements with the help of conventional Fast Marching. For a given r, offset band computation amounts to the simultaneous propagation of (circular) fronts from each input point outwards until the front's extent equals r. The offset band consists of those grid vertices visited during the propagation. All other grid vertices of the point set's bounding volume are discarded to minimise memory usage. When dealing with relatively large point sets, we perform this offset band computation as a pre-processing step which makes the set of valid grid vertices available for subsequent weighted distance mapping. Irregularly distributed point sets may require the offset band to be adaptive. This can be achieved by computing radii r_i using a minimum spanning tree or local principal component analysis as discussed in [19,44] or by using the point neighbourhood-based technique put forward in Moenning and Dodgson [46]. Weighted distance mapping within Ω_P^r (or $\Omega_P^{r_i}$) is achieved by another application of conventional Fast Marching restricted to operate within the offset band only, i.e. the set of grid vertices returned by the preceding band fitting step.

A single min-heap as typically used for the efficient implementation of the conventional Fast Marching method [56] and a Cartesian grid represent the main data structures required in this context. The grid data structure is implemented in the form of a lookup table with each grid vertex mapped to, amongst others, its distance map value and min-heap index and offset band membership status. By using a lookup table, invalid grid vertices can be quickly discarded and different grid spacing in each direction is supported.

6.2.2. Geodesic Voronoi diagrams and intrinsic natural neighbourhoods

The natural neighbourhood information of a point $p_i \in P_l$ is available from $\mathrm{VD}(P_l)$. We approxi-

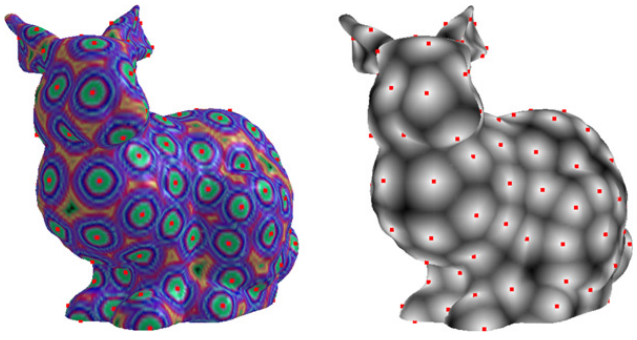

Fig. 13. Wave propagation for the computation of a discrete geodesic Voronoi diagram. By simultaneously propagating waves for geodesic distance mapping purposes from the (red) generator points outwards (left), an intrinsic Voronoi partitioning of the triangulated surface is obtained (right). (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this paper.)

mate this diagram using weighted geodesic distance maps, i.e. in analogy to the dropping of pebbles in still water, circular fronts move across the surface from the points of impact. The locations where wave fronts meet define the geodesic Voronoi diagram of the points of impact. Fig. 13 gives a triangular mesh-based example produced using public domain software [17]. The wave propagation is discretised and simulated accurately by solving Eq. (7) using our implementation of the intrinsic distance mapping for surfaces in point cloud form discussed in Section 6.2.1. This way each point's Voronoi neighbours are obtained *during front propagation* as loci of intersection between geodesic offset curves [57]. That is, by propagating fronts from all $p \in P$ outwards "simultaneously" and inside the offset band Ω_P^r, the boundary of the (discrete) geodesic Voronoi region of p_i is detected at the grid vertices in Ω_P^r at which the front emanating from p intersects with fronts originating from points $p_j \in P$, $j \neq i$. Since each front carries the information of the point it emanated from, the detection of a front intersection implies the detection of an intrinsic natural neighbourhood relation between these points. Thus, in contrast to mesh-based, i.e. extrinsic graph-based, methods such as those presented in [40,58] which could be used for the computation of intrinsic natural neighbourhoods, our technique determines these neighbourhoods truly geodesically and without the need for determining and maintaining auxiliary data structures such as Voronoi edges.

The initial geodesic Voronoi diagram, $\mathrm{VD}(P_0)$, can either be computed during the algorithm's initialisation phase by propagating fronts simultaneously from all $p_i \in P_0$ outwards (Fig. 13) or is already given due to prior intrinsic point cloud simplification as indicated in Section 5.2.

The neighbourhood information is held in the form of a lookup table mapping each p to its set of natural neighbours N_p represented by a set of indices referencing the corresponding input points. The lookup table of (valid) grid vertices, already discussed above, represents the only other main data structure used in this context and is required here, as above, for distance mapping support.

6.2.3. Geodesic centroid computation

Using the natural neighbourhood information N_p of point p, its weighted Euclidean centroid is readily available. We use the "almost" orthogonal projection operator of Alexa and Adamson [59] described in the following section to project this centroid onto M thereby obtaining g_0 (Section 5.3).

Fast Marching for point clouds [19] is then again used to compute distance maps within the offset band from each point in N_p outwards with the squared distance map values being accumulated at the grid vertices. To avoid unnecessary propagation, the extent of the distance mapping is limited to the radius of the sphere enclosing the points in N_p. A standard Runge–Kutta gradient descent procedure with multilinear interpolation is finally employed to back propagate from g_0 towards the weighted geodesic centroid g by following the (negative) gradient of the accumulated distance maps estimated with the help of the normalised central difference operator provided by Blitz++ 0.7 [55].

6.2.4. Orthogonal projection

As pointed out by Alexa and Adamson [59], the first weighted least squares regression of Moving Least Squares [21], which is frequently used to obtain a surface normal estimate (see, e.g. Pauly et al. [24]), fits a support plane to a point which will frequently be located close to but not on the underlying surface. In these cases, the support plane is therefore not tangent to but near the surface and its normal will generally not coincide with the surface normal.

Although in the case of Alexa and Adamson's [59] projection procedure, the normal is again estimated using weighted least squares, the normal estimate and projection are iteratively improved subject to an approximation error threshold ϵ. More specifically, their projection method repeatedly projects a point p_i onto local support planes defined by the normal estimate n at p_i and the weighted average $a(p_i) = \frac{\sum_{j=1}^{\|P\|} \theta_j(\|p_i - p_j\|)p_j}{\sum_{j=1}^{\|P\|} \theta_j(\|p_i - p_j\|)}$. The weight function θ_j determines the influence of point p_j and is assumed to be smooth, positive and monotonically decreasing in distance. Thus, similar to Moving Least Squares [21], Alexa and Adamson's [59] method is based on an implicit surface definition and operates meshlessly. The implicit surface approximation is given by those points p_i for which $f(p_i) = n^T(a(p_i) - p_i)$ equals zero, i.e. $f(p_i)$ describes the distance of the weighted point average $a(p_i)$ in normal direction from the implicit surface described by $f(p_i) = 0$. The orthogonal projection procedure then consists of the steps summarised in Algorithm 3.

Algorithm 3. Approximately orthogonal projection algorithm in pseudocode

Input: Point $p_i \in P$. Approximation error threshold ϵ.
Output: Approximately orthogonal projection p_i' of p_i.

```
0        *** Initialisation ***
1          Set p_i' = a(p_i);
2
3        *** Projection ***
4          REPEAT
5            Compute n at p_i' and a(p_i');
6            Set p_i' = p_i − (n^T(a(p_i') − p_i))n;
7          UNTIL ||n^T(a(p_i') − p_i')|| < ε;
```

Thus, at each iteration, the current projection p_i' is used to update n and $a(p_i')$, i.e. the position of the support plane (line 5); p_i is then projected onto this support plane to obtain an updated projection approximation p_i' (line 6). Upon convergence, this process continues until p_i' is within a given ϵ of the implicit surface, i.e. $\|n^T(a(p_i') - p_i')\| \approx 0$ (line 7). Thus, this technique guarantees (almost) orthogonal projection onto the surface since $p_i' - p_i$ is enforced to be in the direction of normal vector n at p_i'. We use our implementation of this method with a Gaussian as weight function θ_j to obtain a surface normal estimate for the projection of the extrinsic mean onto the surface.

7. Conclusion

The main contributions of this paper are the introduction of meshless subdivision based on the computation of weighted geodesic centroids on manifolds represented by noise-free or noisy point clouds and a new method for the computation of such centroids.

The consideration of local intrinsic point proximity instead of mesh connectivity helps to overcome some of the limitations of existing mesh subdivision schemes, in particular:

- Costly and generally non-geometric mesh preprocessing steps are avoided.
- Any non-linear intrinsic structure of the geometry is captured inherently.
- Special rules for dealing with irregular mesh connectivity are not required.
- The principle applies equally well to three and higher-dimensional surfaces.

Without any assumptions on the availability of normal or local density information, we have experimentally shown the applicability of our meshless subdivision operator for the smoothing of a number of different point sets. Further applications include surface reconstruction and modeling and are the subject of on-going work.

For extremely high-dimensional data, meshless subdivision operators need to be devised which upsample the point-sampled geometry more conservatively than the operator suggested in this paper. In this direction, we are considering introducing adaptive neighbourhoods based on curvature estimators such as those reported in Cazals and Pouget [60] and Mitra and Nguyen [61]. We leave this to future work.

The execution efficiency of our research quality implementation of meshless geometric subdivision does not compare well with that of well-established mesh-based subdivision schemes. As stressed in the introduction and similar to early work on mesh-based subdivision, however, the intention of this paper is to introduce the notion of meshless geometric subdivision and to show its conceptual viability. Future work will be concerned with improving the practical usefulness of the algorithm by optimising the execution and memory efficiency of the implementation.

As first proposed in the context of univariate spline subdivision schemes [50], we further plan to investigate the practical and theoretical aspects of repeating the geometric averaging step several times after each refinement step. We expect to get higher smoothness with a higher number of averaging steps in each iteration of the meshless subdivision process.

Depending on the extent of any non-uniformities of the input point cloud, the experimental selection of point weights can be elaborate. We are therefore currently working on the more systematic choice of weights.

This issue is closely related to the theoretical analysis of our meshless intrinsic subdivision scheme. In this context, we are building upon the ideas of Wallner and Dyn [35–37], which this research was inspired by, in particular their convergence and smoothness analysis of non-linear geodesic curve subdivision by proximity to a corresponding linear extrinsic subdivision scheme.

Acknowledgments

We gratefully acknowledge the permission to use the Michelangelo point sets granted by the Stanford

Computer Graphics group. The Isis, CAD and Bunny point set were taken from the Cyberware and Stanford 3D Scanning Repository websites respectively. The public domain (extended) Loop subdivision implementation was obtained from Henning Biermann's website. Surface reconstructions were performed using Paraform's public domain Points2Polys software.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.gmod.2006.11.001.

References

[1] N. Dyn, D. Levin, Subdivision schemes in geometric modelling, Acta Numerica 12 (2002) 73–144.

[2] D. Zorin, P. Schroder, Subdivision for modeling and animation, SIGGRAPH '00 course notes.

[3] N. Amenta, M. Bern, M. Kamvysselis, A new voronoi-based surface reconstruction algorithm, in: Proceedings of SIGGRAPH '98, 1998, pp. 415–422.

[4] N. Amenta, S. Choi, R. Kolluri, The power crust, in: Proceedings of the 6th ACM Symposium on Solid Modeling, 2001, pp. 249–260.

[5] F. Bernadini, J. Mittleman, H. Rushmeier, C. Silva, G. Taubin, The ball-pivoting algorithm for surface reconstruction, IEEE Transactions on Visualization and Computer Graphics 5 (1999) 349–359.

[6] J.-D. Boissonnat, F. Cazals, Smooth surface reconstruction via natural neighbour interpolation of distance functions, in: Proceedings of the 16th ACM Symposium on Computational Geometry, 2000, pp. 223–232.

[7] B. Curless, M. Levoy, A volumetric method for building complex models from range images, in: Proceedings of SIGGRAPH '96, 1996, pp. 303–312.

[8] H. Edelsbrunner, E.P. Mucke, Three-dimensional alpha shapes, ACM Transactions on Graphics 13 (1994) 43–72.

[9] C. Gotsman, S. Gumhold, L. Kobbelt, Simplification and compression of 3d meshes, in: A. Iske, E. Quak, M.S. Floater (Eds.), Tutorials on Multiresolution in Geometric Modelling, 2002, pp. 319–362.

[10] M.S. Floater, K. Hormann, Surface parameterization: a tutorial and survey, in: Advances in Multiresolution for Geometric Modelling, 2004, pp. 157–186.

[11] P. Alliez, D. Cohen-Steiner, O. Devillers, B. Lévy, M. Desbrun, Anisotropic polygonal remeshing, in: Proceedings of SIGGRAPH '03, 2003, pp. 485–493.

[12] Z. Wood, H. Hoppe, M. Desbrun, P. Schroder, Removing excess topology from isosurfaces, ACM Transactions on Graphics 23 (2) (2004) 190–208.

[13] J.B. Tenenbaum, V. de Silva, J.C. Langford, A global geometric framework for nonlinear dimensionality reduction, Science (2000) 2319–2323.

[14] P.T. Fletcher, C. Lu, S. Joshi, Statistics of shape via principal geodesic analysis on lie groups, in: Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, 2003, pp. 95–101.

[15] J. Jost, Equilibrium maps between metric spaces, Calc. Var. 2 (1994) 173–204.

[16] W.S. Kendall, Probability, convexity and harmonic maps with small image i: uniqueness and fine existence, Proceedings of the London Mathematical Society 61 (2) (1990) 371–406.

[17] G. Peyre, L.D. Cohen, Geodesic re-meshing using front propagation, in: Proceedings of the 2nd IEEE VLSM Workshop, 2003, pp. 33–40.

[18] O. Sifri, A. Sheffer, C. Gotsman, Geodesic-based surface remeshing, in: Proceedings of the 12th International Meshing Roundtable, 2003, pp. 189–199.

[19] F. Mémoli, G. Sapiro, Distance functions and geodesies on submanifolds of $r^d$and point clouds, SIAM Journal of Applied Mathematics 65 (4) (2005) 1227–1260.

[20] C. Loop, Smooth surface subdivision based on triangles, Master's thesis, University of Utah, Department of Mathematics (1987).

[21] M. Alexa, J. Behr, D. Cohen-Or, S. Fleishman, D. Levin, C.T. Silva, Computing and rendering point set surfaces, IEEE Transactions on Visualization and Computer Graphics 9 (1) (2003) 3–15.

[22] L. Linsen, Point cloud representation, Tech. Rep. 2001-3, Computer Science Department, Universitaet Karlsruhe, Germany (2001).

[23] M. Pauly, L.P. Kobbelt, M. Gross, Multiresolution modeling of point-sampled geometry, Tech. Rep. 378, Computer Graphics Lab, ETH Zurich, Switzerland (2002).

[24] M. Pauly, R. Keiser, L.P. Kobbelt, M. Gross, Shape modeling with point-sampled geometry, Proceedings of SIGGRAPH '03 (2003) 641–650.

[25] M. Zwicker, M. Pauly, M. Knoll, M. Gross, Pointshop3d: An interactive system for point-based surface editing, Proceedings of SIGGRAPH '02 (2002) 322–329.

[26] J.P. Grossman, W.J. Dally, Point sample rendering, Proceedings of the 9th Eurographics Workshop on Rendering, 1998, pp. 181–192.

[27] A. Kalaiah, A. Varshney, Differential point rendering, Proceedings of the 12th Eurographics Workshop on Rendering, 2001, pp. 139–150.

[28] M. Levoy, T. Whitted, The use of points as display primitive, Tech. Rep. 85022, UNC-Chapel Hill Computer Science Department, Chapel Hill, North Carolina, USA (1985).

[29] H. Pfister, M. Zwicker, J. van Baar, M. Gross, Surfels: Surface elements as rendering primitives, in: Proceedings of SIGGRAPH '00, 2000, pp. 335–342.

[30] S. Rusinkiewicz, M. Levoy, Qsplat: a multiresolution point rendering system for large meshes, in: Proceedings of SIGGRAPH '00, 2000, pp. 343–352.

[31] M. Zwicker, H. Pfister, J. van Baar, M. Gross, Surface splatting, Proceedings of SIGGRAPH '01 (2001) 371–378.

[32] M. Zwicker, H. Pfister, J. van Baar, M. Gross, Ewa splatting, IEEE Transactions on Visualization and Computer Graphics 8 (3) (2002) 223–238.

[33] S. Fleishman, D. Cohen-Or, M. Alexa, C.T. Silva, Progressive point set surfaces, ACM Transactions on Graphics 22 (4) (2003) 997–1011.

[34] G. Guennebaud, L. Barthe, M. Paulin, Real-time point cloud refinement, in: Proceedings of the Eurographics Symposium on Point-Based Graphics, 2004, pp. 41–49.

[35] J. Wallner, N. Dyn, Smoothness of subdivision schemes by proximity, Tech. Rep.112, Tech. Univ. Wien, Austria, 2003.
[36] J. Wallner, N. Dyn, Smoothness of subdivision schemes in manifolds, Tech. Rep. 125, Tech. Univ. Wien, Austria, 2004.
[37] J. Wallner, N. Dyn, Convergence and cl analysis of subdivision schemes on manifolds by proximity, Computer Aided Geometric Design 22 (2005).
[38] I. Chavel, Riemannian Geometry: a modern introduction, Cambridge University Press, Cambridge, UK, 1997.
[39] R. Kunze, F.-E. Wolter, T. Rausch, Geodesic voronoi diagrams on parametric surfaces, Proceedings of Computer Graphics International '97, 1997, pp. 230–238.
[40] G. Leibon, D. Letscher, Delaunay triangulations and voronoi diagrams for riemannian manifolds, in: Proceedings of the 16th ACM Symposium on Computational Geometry, 2000, pp. 341–349.
[41] J. Helmsen, E.G. Puckett, P. Collela, M. Dorr, Two new methods for simulating photolithography development in 3d, Proceedings of SPIE Microlithography IX, 1996, pp. 253–261.
[42] J.A. Sethian, Theory, algorithms, and applications of level set methods for propagating interfaces, Acta Numerica 5 (1996) 309–395.
[43] J.N. Tsitsiklis, Efficient algorithms for globally optimal trajectories, IEEE Transactions on Automatic Control 40 (1995) 1528–1538.
[44] F. Mémoli, G. Sapiro, Fast computation of weighted distance functions and geodesies on implicit hyper-surfaces, Journal of Computational Physics 173 (1) (2001) 764–795.
[45] J. Giesen, U. Wagner, Shape dimension and intrinsic metric from samples of manifolds with high co-dimension, Proceedings of the 19th Symposium on Computational Geometry, 2003, pp. 329-337.
[46] C. Moenning, N.A. Dodgson, Intrinsic point cloud simplification, in: Proceedings of the 14th GraphiCon '04, 2004.
[47] F. Mémoli, G. Sapiro, A theoretical and computational framework for isometry invariant recognition of point cloud data, Foundations of Computational Mathematics 5 (3) (2005) 313–347.
[48] M.S. Floater, M. Reimers, Meshless parameterization and surface reconstruction, Computer Aided Geometric Design 18 (2) (2001) 77–92.
[49] R. Sibson, A vector identity for the dirichlet tessellation, in: Math. Proceedings of the Cambridge Philos. Soc, 1980, pp. 151–155.
[50] M. Lane, R.F. Riesenfeld, A theoretical development for the computer generation of piecewise polynomial surfaces, IEEE Transactions on Pattern Analysis and Machine Intelligence 2 (1980) 35–46.
[51] P. Oswald, P. Schroder, Composite primal/dual sqrt(3)-subdivision schemes, Computer Aided Geometric Design 20 (3) (2003) 135–164.
[52] S.R. Buss, J.P. Fillmore, Spherical averages and applications to spherical splines and interpolation, ACM Transactions on Graphics 20 (2) (2001) 95–126.
[53] H. Karcher, Riemannian center of mass and mollifier smoothing, Communications on Pure and Applied Mathematics 30 (5) (1977) 509–541.
[54] T. Weyrich, M. Pauly, R. Keiser, S. Heinzle, S. Scandella, M. Gross, Postprocessing of scanned 3d surface data, Proceedings of the Eurographics Symposium on Point-Based Graphics, 2004, pp. 85–94.
[55] Blitz++ Numerical Library (2003).
[56] J.A. Sethian, Level Set Methods and Fast Marching Methods, 2nd ed., Cambridge University Press, Cambridge, UK, 1999.
[57] L.D. Cohen, Multiple contour finding and perceptual grouping using minimal paths, Journal of Mathematical Imaging and Vision 14 (3) (2001) 225–236.
[58] A. Okabe, B. Boots, K. Sugihara, Spatial Tessellations—Concepts and Applications of Voronoi Diagrams, 2nd ed., John Wiley, Chicester, UK, 2000.
[59] M. Alexa, A. Adamson, On normals and projection operators for surfaces defined by point sets, Proceedings of the Eurographics Symposium on Point-Based Graphics, 2004, pp. 149–156.
[60] F. Cazals, M. Pouget, Estimating differential quantities using polynomial fitting of osculating jets, Proceedings of the Eurographics Symposium on Computational Geometry, 2003, pp. 177–187.
[61] N.J. Mitra, A. Nguyen, Estimating surface normals in noisy point cloud data, Proceedings of the 19th Conference on Computational Geometry, 2003, pp. 322–328.