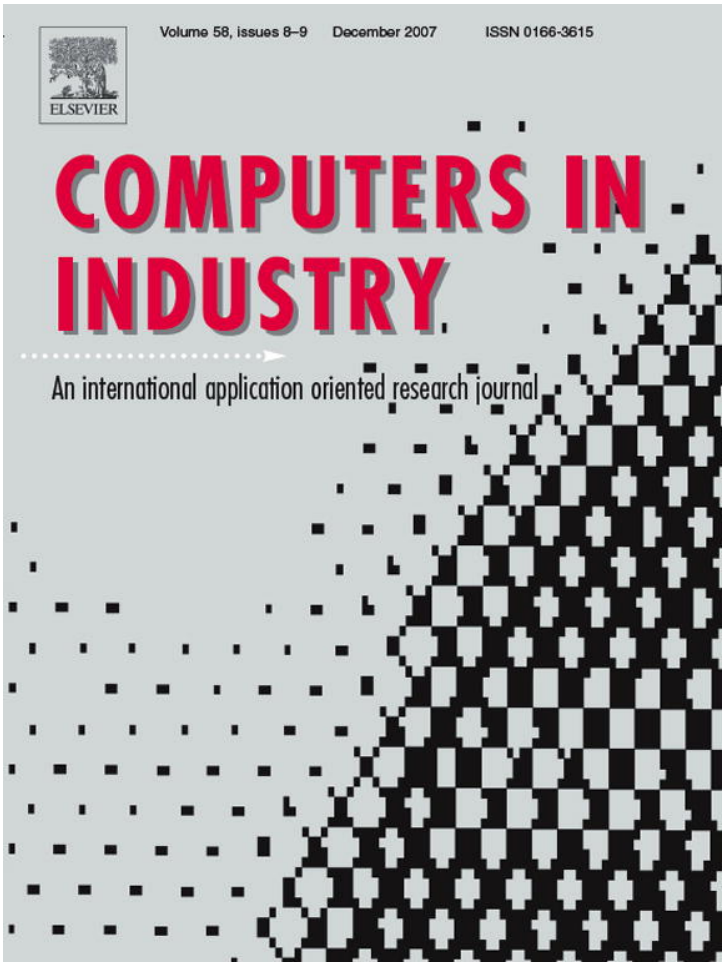

Available online at www.sciencedirect.com
ScienceDirect
Computers in Industry 58 (2007) 832–843
COMPUTERS IN INDUSTRY
www.elsevier.com/locate/compind

Product design using point-cloud surfaces: A recursive subdivision technique for point parameterization

Philip Azariadis *, Nickolas Sapidis

University of the Aegean, Department of Product & Systems Design Engineering, Greece

Received 17 April 2006; received in revised form 20 September 2006; accepted 13 March 2007
Available online 25 April 2007

Abstract

This paper presents a method for parameterizing three-dimensional (3D) point-clouds using a recursive subdivision approach. The proposed solution adapts ideas from emerging point-based geometric modelling and extends the dynamic base surfaces (DBS) concept in order to improve the accuracy of the produced parameterizations. Using the new approach it is possible to compute parameterizations for point-clouds which may be ''thick'' or with a varying density. Indicative examples are presented to illustrate the benefits of the proposed method.

Keywords: Point-cloud; Point-set surfaces; Parameterization; Dynamic base surfaces; Point-based modelling

1. Introduction

Point-based modelling has received a considerable attention during the last years mainly due to the availability of fast and accurate measuring devices and the increased processing capacity of modern personal computers. The basic idea is to use the point as the primary geometric structure for representing object surfaces. Under this way a surface comprises of a set of points known as a *point-cloud*. Usually, no additional geometric or topological information is stored in these point-based structures which are also called as *point-set surfaces* [1].

Nowadays, a considerable amount of research work focuses on performing various geometric modelling and graphics tasks using a point-set surface; indicative examples include:

- Rendering [1–3].
- Object manipulation and editing [4,5].
- CNC machining and rapid prototyping (RP) [6,7].
- Interactive surface painting [8] and curve drawing onto point-clouds for industrial design [9,10].

On the other hand, existing CAD/CAE/CAM systems for developing free-form products use as *primary CAD-model* a representation based on continuous, analytic curves and surfaces (e.g., NURBS). Thus, when the input is a point-cloud, a reverse engineering (RE) method is used to ''translate'' the given point-model into a curve/surface model [11]. A major processing phase in RE is the assignment of adequate parametric values that best represent the points' position within the rectangular domain in the plane. This problem becomes complicated with point-clouds with (a) varying density, (b) holes or sharp edges, (c) boundaries. Furthermore, modern data-acquisition devices produce very dense or ''thick'' (otherwise noisy) point-clouds rendering a lot of existing parameterization methods inappropriate.

The above issues hinder modern design-technologies like virtual engineering [12] and rapid prototyping [7] since these are using, almost exclusively, polygon meshes. For instance, stereolithography (STL) format has become an industry standard with respect to RP or 3D printing [13]. However, producing a robust surface triangulation from a point-cloud is still a hard procedure which involves user's intervention and a good point parameterization.

This paper presents a new method for parameterizing unstructured point-clouds based primarily on ideas from point-based geometric modelling. The new method is built upon the dynamic base surfaces (DBS) method introduced in [14] and improved shortly after in [9,10,15]. The new method is able to process point-sets with some or all of the characteristics (a)–(c).

* Corresponding author.
E-mail addresses: azar@aegean.gr (P. Azariadis), sapidis@aegean.gr (N. Sapidis).

0166-3615/$ – see front matter
doi:10.1016/j.compind.2007.03.001

Although the proposed method is suitable for point-clouds with disk-like topology, which is considered in [16] as ''the most-important parameterization task'', the new method is able to handle trimmed surfaces with holes or with undersampled areas.

It must be clarified that, it is meaningless in the context of the present research to talk about ''estimating measurement error in a point-cloud'' or about ''denoising a point-cloud''. Indeed, this research considers the point-cloud as a ''*primary geometric model*'', at the same level with NURBS, solid or triangulation models, describing three-dimensional objects in a CAD/CAE/CAM or computer-graphics system. Thus, a given point-cloud is treated as the only available (''exact'') model for a geometric object that must be further processed, so that specific design/analysis or visualization tasks are achieved.

The paper is structured as follows: Section 2 reviews existing parameterization methods focusing on the original DBS method. Sections 3 and 4 present the components of the new recursive DBS method. The main technical details of the new method are described in Sections 5 and 6. Finally, Section 7 presents an in-depth discussion of selected examples.

2. Previous work and motivation

Since the introduction of texture mapping by Catmull in 1974 [17], one of the most intensive areas of computer graphics research involved the determination of adequate surface parameterizations for performing non-distorted texture mapping. Indeed, the early method of Ma and Lin [18] considered a three-dimensional (3D) triangle mesh which is flattened onto the plane preserving (optimally) the lengths of the 3D triangle edges. Later, Maillot et al. [19] improved the proposed surface parameterization method by taking into account the signed areas of the 3D triangles.

Azariadis and Aspragathos [20] utilized a constrained energy model, which is an extension of [19], in order to derive surface parameterizations with local control. That paper established also a method for measuring the metric distortion between a 3D triangle mesh and the corresponding 2D parameterization by examining a piecewise-linear mapping between those two domains in a triangle-by-triangle basis. The result of every such comparison is two values named as d_p and d_q which express the mapping distortion between the corresponding triangles. This analytically computed distortion is then used for the development of a space-variant filtering technique for antialising texture mapped images. A similar approach is followed for performing a virtual quality control in the 2D parameterization of a given 3D triangulated surface [21].

During that period Hormann and Greiner [22] studied independently the distortion between a triangle in a 3D mesh and its corresponding in the 2D parameterization and concluded also to the same distortion indices denoted therein by σ_1 and σ_2. These indices are utilized to formulate and minimize a non-linear energy function in order to derive a most isometric parameterization of a 3D triangle mesh. This method has been successfully applied to remeshing triangulated surfaces [23].

Many parameterization methods employ an underlying 3D triangle mesh and, generally, using an iterative procedure they produce a topologically identical 2D triangulation (or parameterization). An extensive review of parameterization methods is given by Floater and Hormann [16] and it is beyond the scope of this paper. We present, however, a set of indicative methods for parameterizing a 3D triangle mesh.

Harmonic parameterizations minimize the squared distances of the triangle edge lengths [24]. The proposed method requires a fixed and convex planar boundary to produce the underlying conformal maps. Floater's [25] shape preserving approach locates the new vertices of the interior triangles using barycentric mappings. Horman and Greiner [26] utilized hierarchical representations of triangulated surfaces in order to speed up the parameterization process. Through this process the global problem is represented by a hierarchical parameterization of different levels of detail.

In [27] a special multi-dimensional scaling (MDS) method is proposed to parameterize a curved surface utilizing geodesic distances. Zigelman et al. [28] improved this method by introducing a new mapping technique that preserves both the local and the global structure of the parameterization with minimal shearing effects. Sander et al. [29,30] used a ''texture stretch'' metric based on the singular values of the Jacobian of the affine 2D-to-3D mapping to compute a good parameterization.

Levy et al. [31] proposed the so called least squares conformal maps which are built upon a criterion that minimizes angle deformations and non-uniform scalings. Desbrun et al. [32] developed a set of intrinsic parameterizations which preserve either angles (discrete conformal mapping) or areas (discrete authalic mapping) which are combined to form a general discrete parameterization method. They apply their theoretical results to parameterize 3D triangle meshes with a fixed 2D boundary in the parametric space. They also show how to interactively optimize the boundary of the parameterization with respect to an appropriate energy.

Yu et al. [33] presented an algorithm using the in-plane strain related to the transformation of the curved surface to its planar development. Another approach [34] formulated the planar development problem using a spring–mass system and calculates the strain energy released during flattening.

Another set of parameterization methods considers that the point-cloud is ''enclosed'' within a 3D boundary topologically identical to a disk which is mapped onto a convex 2D border with equivalent topology. This approach obtains a parameterization by solving a linear system to determine the 2D embedded positions of the interior vertices [35]. Various methods are developed to allow non-convex 2D boundaries; see e.g., [36].

Other approaches in the above category consider a *base surface* interpolating a given 3D boundary. The orthogonal projection of the point-cloud onto the base surface produces the required parameterization [37]. *Dynamic base surfaces* [14] extend the ''base surface concept'' by approximating the surface of a point-cloud iteratively. Under a similar ''in-spirit'' approach *active surfaces* [38] approximate the shape of a point-set surface through an iterative scheme by minimizing a

quadratic approximation of a squared distance function expressing the distance from the active surface to the point-cloud.

2.1. Point-cloud parameterization based on dynamic base surfaces

Given a cloud of points $\mathcal{C} = \{\mathbf{p}_\mu = (x_\mu, y_\mu, z_\mu) | \mu = 0, \ldots, N-1\}$ and a set of four boundary curves, topologically identical to a rectangle, the original DBS method [14] produces a continuous surface which interpolates the given boundary and approximates the cloud points. This method is based on a grid-projection scheme where an initial grid of points and a set of corresponding projection directions are obtained from the approximating surface. The grid of points is projected onto the point-cloud by linearly minimizing an energy function which is a convex combination of a distance metric and a first-order smoothing functional. Then a continuous DBS is constructed by interpolating this (projected) grid. If the approximation error (defined to be the mean distance between every cloud point and its projection point onto the DBS) is large, then a new grid is obtained with increased size and the overall procedure is repeated as long as the approximation accuracy is not achieved and the grid size is under a predefined threshold.

One of the main advantages of DBS parameterization, compared against existing methods, is that DBS produces satisfactory results also for "thick" clouds with many points, and for point-clouds with a varying density. Indeed, parameterizing methods based on convex combinations (see e.g., [35]) impose a sparse linear system with millions of coefficients which is quite difficult to store in computer memory and solve. Nevertheless, variations in the point-cloud density can introduce either wrong neighbourhood estimations or erroneous calculations of weighting coefficients. On the other hand, an attempt to triangulate a "thick" point-cloud can fail due to the lack of correct topological data. Other existing methods for surface reconstruction using triangle meshes contradict all the above characteristics as they need a constant point-cloud density [39] or a "good sample from a smooth surface" [40,41].

Although the original DBS parameterization method attempts to approximate the entire point-cloud with one continuous surface minimizing, therefore, the number of control points, it is not possible to guarantee an acceptable surface approximation for the entire point-cloud. Indeed, quite often the approximation error is acceptable for most of the cloud and still it is quite large for some very small regions of that. This results to poor point parameterization in these regions.

Fig. 1 illustrates a failure of the original DBS method to approximate the front part of the point-cloud of a shoe last. Although the mean approximation error is 0.341 mm (surface bounding box is 246 mm × 38 mm × 102 mm) there is still a large error in the area indicated in Fig. 1b which will result to poor local parameterization.

The proposed method overcomes the local accuracy problem by following a "divide and conquer" approach. When a large local error of approximation is identified the current "base surface modelling problem" is subdivided in four such problems which are treated independently. The overall structure is represented by a quadtree scheme, while neighbouring DBS patches are seamed under a C^0 rule.

Fig. 2 shows the result of applying the new recursive DBS method to the shoe last point-cloud. The previous accuracy problems are overcome and the new base surface approximates the point-cloud with a mean accuracy of 0.1 mm. The new method is also based on a new grid-projection technique which produces smoother DBS patches.

3. Approximating a point-cloud with a C^0 composite surface based on recursive DBS subdivision

The proposed point parameterization methodology refers to a given cloud of points $\mathcal{C}$, equipped with (i) a four-sided boundary $\mathcal{B} = \{b_l, b_r, b_t, b_b\}$ (topologically identical to a rectangle), and (ii) a "cloud surface accuracy criterion" expressing the average

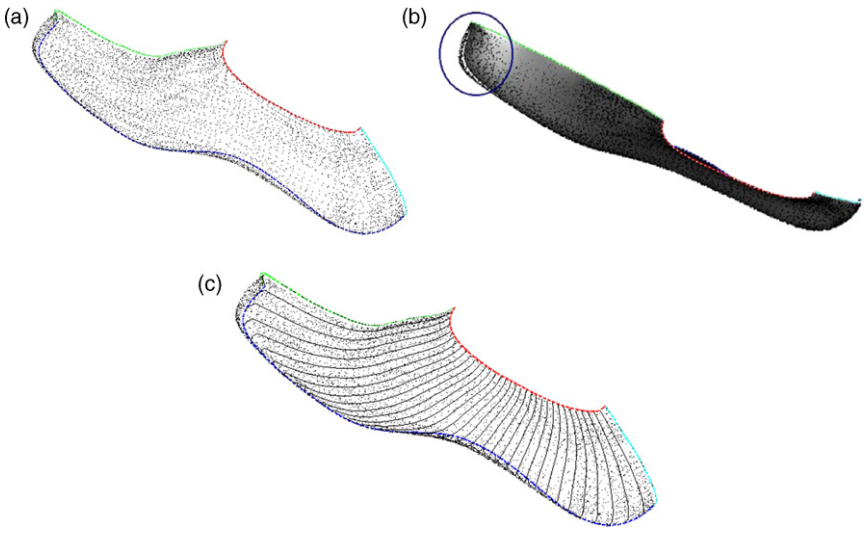

Fig. 1. (a) A point-cloud together with a set of boundary curves (denoted by the red, blue, green and cyan curves). (b) The resulted dynamic base surface is not able to approximate accurately the surface forepart. (c) The corresponding grid of points of the final DBS.

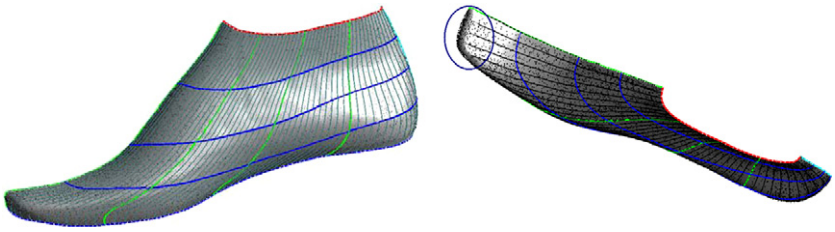

Fig. 2. Different views of the result of the new recursive DBS method.

distance between the cloud points and their projection on the approximating surface $\mathcal{A}$. The present parameterization methodology requires that $\mathcal{C}$ is approximated by a surface $\mathcal{A}$ that satisfies the accuracy criterion. However, when this criterion is not satisfied then $\mathcal{A}$ is subdivided into smaller parts $\mathcal{A}_i$ and cloud $\mathcal{C}$ is approximated by a C^0 assembly $\{\mathcal{A}_i\}$ of surfaces. The overall problem is summarized in the following:

Problem R Subdivide DBS $\mathcal{A}$ and cloud $\mathcal{C}$ into sub-surfaces $\{\mathcal{A}_i\}$ and subclouds $\{\mathcal{C}_i\}$, where $\mathcal{C} \equiv \bigcup \mathcal{C}_i$, so that (a) each $\mathcal{C}_i$ is approximated by a surface $\mathcal{A}_i$ in accordance with the given accuracy criterion, and (b) the surfaces $\mathcal{A}_i$ form a C^0 assembly $\{\mathcal{A}_i\}$ of surfaces whose boundary is $\mathcal{B} = \{b_l, b_r, b_t, b_b\}$.

The proposed algorithm to solve *Problem R* is an iterative process, whose initial stage ($k = 0$) defines the corresponding surface $\mathcal{A}(0)$ as a Coons-patch interpolating $\mathcal{B}(0) \equiv \mathcal{B}$. When $\mathcal{A}(0)$ is not approximating $\mathcal{C}(0)$ with the required accuracy, the following iterative surface-subdivision process is applied (here described for the kth level of subdivision).

Let $\mathcal{A}(k)$ be the dynamic base surface that approximates a cloud $\mathcal{C}(k)$ after k levels of subdivisions, and let $\mathcal{B}(k) = \{b_l(k), b_r(k), b_t(k), b_b(k)\}$ denote the closed boundary of $\mathcal{A}(k)$. $\mathcal{A}(k)$ is subdivided into four sub-surfaces according to the following Algorithm A1 (see also Fig. 3). Briefly, this subdivision requires (a) four new sets of boundaries (one set for each sub-surface) and (b) four new subsets of point-clouds, for associating every subset with the corresponding sub-surface. The subdivision procedure is driven by certain terminating criteria which assert whether a sub-surface approximates the corresponding subset of cloud points with the required accuracy. The algorithm details are given below.

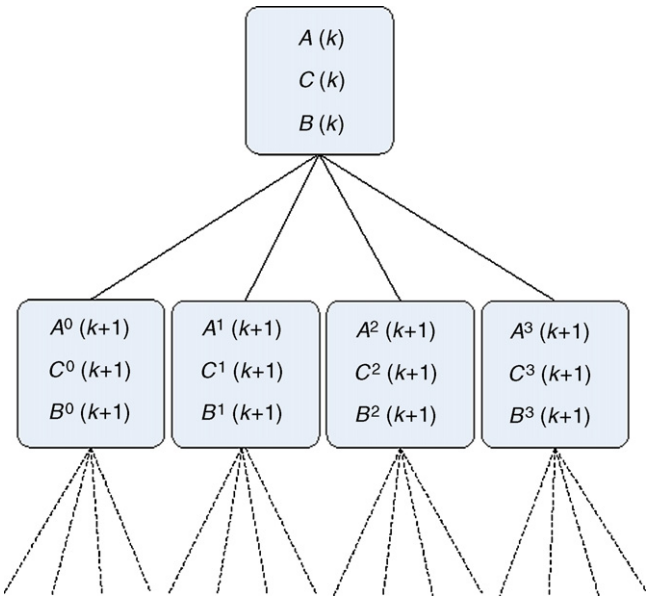

Fig. 3. The quadtree structure of the recursive dynamic base surfaces.

Algorithm A1 (*Recursive DBS subdivision*).

A1.1 The two "middle" parametric curves of $\mathcal{A}(k)$ are calculated for $u = 1/2$, $v = 1/2$, respectively, and are projected, in a "discrete manner", onto the point-cloud $\mathcal{C}(k)$. More specifically, on each curve, a finite set of points is selected and projected onto $\mathcal{C}(k)$ along certain projection directions which are normal to $\mathcal{A}(k)$ at these points (see Section 5). In this way two new "digital curves" are obtained, namely $b_{l,r}(k)$ and $b_{l,b}(k)$ (examples are the "middle" curves in Fig. 4b).

A1.2 Using the two digital curves defined in A1.1, four four-sided boundaries are derived after splitting their corresponding parametric intervals in half. Thus one derives four sets of boundary curves $\mathcal{B}^i(k+1) = \{b_l^i(k+1), b_r^i(k+1), b_t^i(k+1), b_b^i(k+1)\}$, for $i = 1, \ldots, 4$.

A1.3 Using $\mathcal{B}^i(k+1)$ and $\mathcal{A}(k)$ four new sets of grid points $\mathcal{G}^i(k+1) = \{\mathbf{p}_{m,n}^i(k+1)\}$ are derived for $i = 1, \ldots, 4$, where m and n denote the grid size in the two parametric directions, respectively. The obtained grids are associated with projection directions $\mathcal{N}^i(k+1) = (\mathbf{n}_{m,n}^i(k+1))$ which are normal to $\mathcal{A}(k)$ at $\mathbf{p}_{m,n}^i(k+1)$.

A1.4 By interpolating $\mathcal{G}^i(k+1)$ four dynamic base surfaces $\mathcal{A}^i(k+1)$ are obtained. All derived surfaces are seamed along the boundaries $b_{l,r}(k)$ and $b_{l,b}(k)$ with C^0 continuity.

A1.5 Using $\mathcal{B}^i(k+1)$ and $\mathcal{A}^i(k+1)$, $\mathcal{C}(k)$ is subdivided into four subclouds $\mathcal{C}^i(k+1)$ ($\mathcal{C}(k) = \bigcup_i \mathcal{C}^i(k+1)$). Every cloud $\mathcal{C}^i(k+1)$ is associated to one surface $\mathcal{A}^i(k+1)$, for $i = 1, \ldots, 4$ (see subclouds in Fig. 4c).

A1.6 Every surface $\mathcal{A}^i(k+1)$ is projected towards $\mathcal{C}^i(k+1)$ (see Section 5 for the details of this operation). This step involves a terminating criterion which is applied to every $\mathcal{A}^i(k+1)$ surface as follows:

A1.6.1 If $\mathcal{A}^i(k+1)$ is not approximating $\mathcal{C}^i(k+1)$ with the required accuracy then,

A1.6.1.a if the grid size is under a predefined threshold, then a new row and column is added to the grid along the two parametric directions, respectively, and

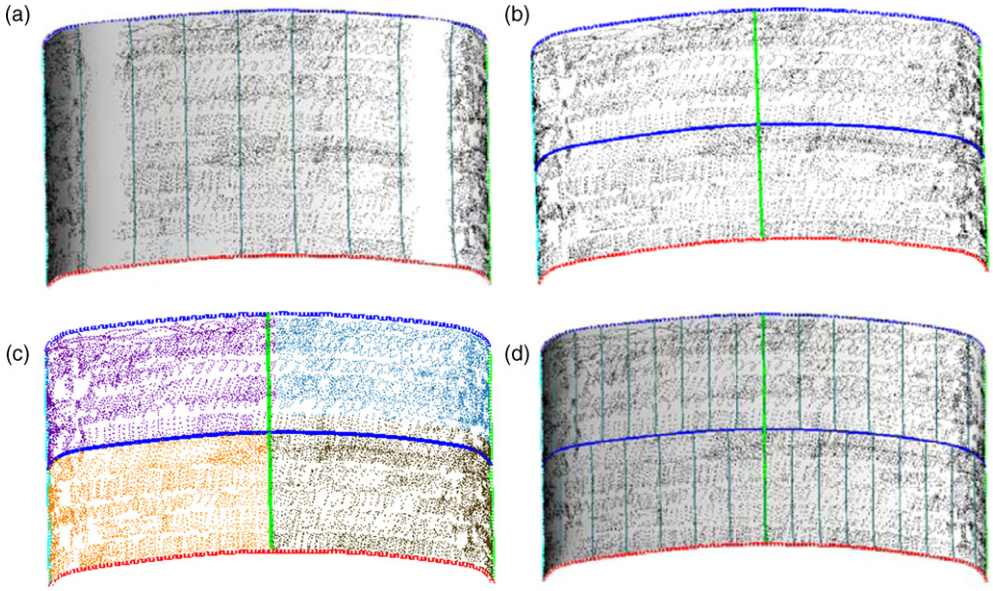

Fig. 4. (a) An initial DBS approximating a point-cloud. (b) The surface "middle" curves projected onto the point-cloud. (c) The subdivision of the point-cloud in four subclouds. (d) Four DBS patches approximating the four different subclouds.

the DBS evolution continues iteratively. Otherwise,

A1.6.1.b the current surface $\mathcal{A}^i(k+1)$ is subdivided and a separate "surface approximating procedure" is started from step A1.1.

A1.6.2 If the terminating criterion is satisfied the current DBS evolution procedure is finalised.

The overall procedure is represented by a quadtree structure as it is shown in Fig. 3. The maximum allowed leaf level (or DBS subdivisions) is denoted by $\ell_{\max}$ and it is usually equal to four. Fig. 4 shows the subdivision of a dynamic base surface and a point-cloud into four sub-surfaces and four subclouds, respectively.

4. Point-cloud parameterization using the recursive DBS method

Every cloud point is parameterized by projecting it orthogonally onto the corresponding DBS surface. The subdivision process described in Algorithm A1 produces a set of dynamic base surfaces which may lie in different quadtree levels. Thus, one has to associate with each $\mathcal{A}^i(\ell)$, where ℓ denotes the leaf level ($1 \le \ell \le \ell_{\max}$) and $i = 1, \ldots, 4$, a rectangular area in the parametric domain $\mathcal{I} = [0,1]^2$ for assigning parameter values to the corresponding points of cloud $\mathcal{C}^i(\ell)$. This is performed by subdividing $\mathcal{I}$ in rectangular areas according to the resulted quadtree structure. Therefore, every leaf in the DBS quadtree structure is associated to a parametric domain $\mathcal{I}^i(\ell) \subset \mathcal{I}$.

The point-cloud subdivision method mentioned in A1.5 (and detailed later in Section 6) produces subclouds which have a limited number of points duplicated in the vicinity of the surfaces' boundaries. Thus, the parameterization procedure should select for every such point the best parameter values from all possible alternatives. The overall parameterization procedure is presented formally in Algorithm A2 below, while technical details are given in subsequent sections.

Algorithm A2 (*Point-cloud parameterization*).

A2.1 For every subcloud $\mathcal{C}^i(\ell)$ and for each point $\mathbf{p}^i_\rho(\ell) \in \mathcal{C}^i(\ell)$, $\rho = 0, \ldots, N^i(\ell) - 1$ (where $N^i(\ell)$ is the number of points in $\mathcal{C}^i(\ell)$) do:

A2.2.1 Project $\mathbf{p}^i_\rho(\ell)$ onto $\mathcal{A}^i(\ell)$. This operation produces a point $\mathbf{q}^i_\rho(\ell)$ on $\mathcal{A}^i(\ell)$ and a pair of parameters $\mathbf{u}^i_\rho(\ell) = (u^i_\rho(\ell), v^i_\rho(\ell)) \in \mathcal{I}^i(\ell)$.

A2.2.2 Compute and store along with $\mathbf{p}^i_\rho(\ell)$ and $\mathbf{u}^i_\rho(\ell)$ the squared distance $d^i_\rho(\ell) = ||\mathbf{p}^i_\rho(\ell) - \mathbf{q}^i_\rho(\ell)||^2$.

A2.2 For every cloud point $\mathbf{p}_\mu \in \mathcal{C}$ do:

A2.2.1 Find all subclouds $\mathcal{C}^i(\ell)$ that contain $\mathbf{p}_\mu$. This step actually produces a set of indices $\mathcal{J} = \{(i, \ell, \rho(\mu)) | \mathbf{p}_\mu \in \mathcal{C}^i(\ell), \ \mathbf{p}_\mu = \mathbf{p}^i_{\rho(\mu)}(\ell)\}$, with $1 \le i \le 4$ and $1 \le \ell \le \ell_{\max}$, such that $\mathbf{p}_\mu = \mathbf{p}^i_{\rho(\mu)}(\ell) \in \bigcap_{(i,\ell) \in \mathcal{J}} \mathcal{C}^i(\ell)$.

A2.2.2 Let $\lambda = \min_{\rho(\mu)} \arg\{d^i_{\rho(\mu)}(\ell) | (i, \ell, \rho(\mu)) \in \mathcal{J}\}$.

A2.2.3 The parameter values for $\mathbf{p}_\mu$ are $\mathbf{u}_\mu = \mathbf{u}^i_\lambda(\ell)$.

A2.2.4 Remove $\mathbf{p}_\mu$ from the "active" point-cloud list and continue until all points $\mathbf{p}_\mu$ have been processed.

5. Grid-projection onto the point-cloud

An important component of Algorithm A1 is a tool for projecting points or point sets onto a point-cloud. This tool is utilized i.e., in steps A1.1 and A1.6 to project either a surface curve or a point-grid onto the cloud of points. In this section we present a new "grid-onto-cloud" projection method that enhances significantly the DBS methodology and also can be used in other problems (i.e., digital curves design).

5.1. Point-projection onto a point-cloud

Let $\mathcal{C}$ be the given point-cloud and let $\mathbf{p} = (x, y, z)$ be an arbitrary 3D point, which must be projected onto $\mathcal{C}$ along the associated projection vector $\mathbf{n} = (n_x, n_y, n_z)$. The pair $\mathbf{p}$, $\mathbf{n}$ is denoted by $\hat{\mathbf{p}} = \langle \mathbf{p}, \mathbf{n} \rangle$. Then, the *directed projection* $\mathbf{p}^*$ of $\hat{\mathbf{p}} = \langle \mathbf{p}, \mathbf{n} \rangle$ onto the point-cloud $\mathbf{C}$ is defined as follows: each $\mathbf{p}_\mu \in \mathbf{C}$ is associated to a positive weight a_μ (these are defined below). Then, $\mathbf{p}^*$ is the solution of the problem (see extensive analysis in [9,10]):[1]

$$\text{find } \mathbf{p}^* \text{ minimizing } E(\mathbf{p}^*) = \sum_{\mu=0}^{N-1} a_\mu ||\mathbf{p}^* - \mathbf{p}_\mu||^2 \tag{1}$$

We describe $\mathbf{p}^* = (x^*, y^*, z^*)$ as

$$\mathbf{p}^* = \mathbf{p}^*(t) = \mathbf{p} + t\mathbf{n}, \quad t \in \Re \tag{2}$$

Then, the solution of problem (1) corresponds to

$$t = \frac{\lambda - \mathbf{pn}}{||n||^2} \tag{3}$$

where

$$\lambda = \frac{c_1 n_x + c_2 n_y + c_3 n_z}{c_0}, \text{ and } c_0 = \sum_{\mu=0}^{N-1} a_\mu, c_1 = \sum_{\mu=0}^{N-1} a_\mu x_\mu, c_2 = \sum_{\mu=0}^{N-1} a_\mu y_\mu, c_3 = \sum_{\mu=0}^{N-1} a_\mu z_\mu \tag{4}$$

The works [9,10] have established that appropriate values for the weights $\{a_\mu\}$ are given by

$$a_\mu = \frac{1}{1 + ||\mathbf{p}_\mu - \mathbf{p}||^2 ||(\mathbf{p}_\mu - \mathbf{p}) \times \mathbf{n}||^2}, \quad a_\mu \in [0, 1] \tag{5}$$

5.2. Grid definition and projection

Given a parametric surface $\mathbf{S} = \mathbf{S}(u, v)$, $u, v \in [0, 1]$, and a point-cloud $\mathcal{C}$:

(a) A set of points $\mathcal{G} = \{p_{i,j}\}$ is selected along the two parametric directions of $\mathbf{S}$, which constitutes the so called *grid of points* denoted by $\mathbf{p}_{i,j} = \mathbf{S}(u_i, v_j)$, $i = 0, \ldots, m-1$, $j = 0, \ldots, n-1$. Each grid point is coupled with a projection direction, which is the actual surface normal direction at the given point. Thus, the grid $\mathbf{p}_{i,j}$ is associated to a *grid of normal vectors* $\mathbf{n}_{i,j}$, where $\mathbf{n}_{i,j} = (\mathbf{S_u}(u_i, v_j))(\mathbf{S_v}(u_i, v_j))/||(\mathbf{S_u}(u_i, v_j))(\mathbf{S_v}(u_i, v_j))||$.

(b) Every grid point is projected onto the point-cloud along the associated direction. This projection is performed "row-" or "column-wise" i.e., an entire row (respective column) of grid points is simultaneously projected onto the point-cloud. Under this way one is able to control the smoothness of a projected *grid section* (i.e., a grid row or column) by minimizing its chord length.

[1] The Euclidean norm of a vector $\mathbf{v}$ is denoted by $||\mathbf{v}||$.

The grid-projection procedure terminates when all grid sections are projected onto the point-cloud.

5.3. The new smooth grid-projection: introducing a second-order smoothing functional in the projection procedure

A new method for projecting the grid of points $\mathbf{p}_{i,j}$ along the associated direction vectors $\mathbf{n}_{i,j}$ onto the point-cloud is proposed in this section. Below, a set of row or column grid-points $\hat{\mathbf{p}}_{i,j0} = \langle \mathbf{p}_{i,j0}, \mathbf{n}_{i,j0} \rangle$, $i = 0, \ldots, m-1$, $0 \le j0 < n$ paired with their corresponding direction vectors will be called as a *polygonal curve*. The basic steps of the new projection algorithm are outlined in the following:

Algorithm A3 (*Smooth polygonal curve projection*).

A3.1 The polygonal curve $\hat{\mathbf{p}}_{i,j0}$ is discretized producing a set $\hat{p} = \{\hat{\mathbf{p}}_k = \langle \mathbf{p}_k, \mathbf{n}_k \rangle | k = 0, \ldots, K-1\}$ of K distinct *nodes* $\mathbf{p}_k$. This discretization constitutes a relatively dense sampling of the initial polygonal curve $\mathbf{p}_{i,j0}$ and it is achieved by setting a constant length between successive sample points.

A3.2 The node-set $\hat{p}$ is projected onto the point-cloud by minimizing an energy function (see details below). During the projection process the first and last nodes, $\hat{\mathbf{p}}_0$ and $\hat{\mathbf{p}}_{K-1}$, are fixed, since they lie onto the point-cloud boundary.

A3.3 The projected node-set (which is a polygonal curve considered to lie onto the cloud) is sampled producing m grid points $\mathbf{p}^*_{i,j0}$.

The smooth projection of $\hat{p}$ onto $\mathcal{C}$ is a polygonal curve $q^* = \{\mathbf{p}^*_k\}$ that minimizes a convex combination of a distance function and a first- and second-order smoothing function:

$$E = (1 - \gamma)D + \gamma(L + C), \quad \gamma \in [0, 1] \tag{6}$$

where

$$D = \sum_{k=1}^{K-1} E(\mathbf{p}^*_k) \tag{7}$$

is a "measure" of the distance between the polygonal curve and the point-cloud,

$$L = \sum_{k=0}^{K-2} ||\mathbf{p}^*_k - \mathbf{p}^*_{k+1}||^2 \tag{8}$$

measures the length of the projected polygonal curve, and

$$C = \sum_{k=0}^{K-3} ||\mathbf{p}^*_k - 2\mathbf{p}^*_{k+1} + \mathbf{p}^*_{k+2}||^2 \tag{9}$$

approximates the second derivative of the polygonal curve q^*. The weight factor γ is used to fine tune the projection process, e.g., one would like to increase γ in point-clouds with high curvature regions and vice versa.

Each $\mathbf{p}_k^*$ is defined by $\mathbf{p}_k^* = \mathbf{p}_k + t_k\mathbf{n}_k$, thus, the whole polygonal curve q^* is defined by the vector $\mathbf{t} = (t_k)$ $(k = 1, \ldots, K-2)$ (we set $t_0 = t_{K-1} = 0$ since the first and last nodes are fixed). Eq. (6) is minimized when $\mathbf{t}$ is the solution of the linear system:

$$[(1-\gamma)\mathbf{I} + \gamma(\mathbf{A}+\mathbf{B})]\mathbf{t} = (1-\gamma)\mathbf{e} + \gamma(a+b) \tag{10}$$

i.e.

$$t = [(1-\gamma)\mathbf{I} + \gamma(\mathbf{A}+\mathbf{B})]^{-1}[(1-\gamma)\mathbf{e} + \gamma(\mathbf{a}+\mathbf{b})] \tag{11}$$

The matrix $\mathbf{A}$ is tridiagonal and symmetric with $(K-2)\times(K-2)$ elements and is given by

$$\mathbf{A} = \begin{pmatrix} 2 & -\mathbf{n}_1\mathbf{n}_2 & & & \\ & \vdots & & & \\ & -\mathbf{n}_{k-1}\mathbf{n}_k & 2 & -\mathbf{n}_k\mathbf{n}_{k+1} & \\ & & & \vdots & \\ & & & -\mathbf{n}_{K-2}\mathbf{n}_{K-3} & 2 \end{pmatrix} \tag{12}$$

Similarly, the matrix $\mathbf{B}$ is banded and symmetric with $(K-2)\times(K-2)$ elements and is given by

$$\mathbf{B} = \begin{pmatrix} 5 & -4(\mathbf{n}_1\mathbf{n}_2) & \mathbf{n}_1\mathbf{n}_3 & & & & \\ -4(\mathbf{n}_1\mathbf{n}_2) & 6 & -4(\mathbf{n}_3\mathbf{n}_2) & \mathbf{n}_4\mathbf{n}_2 & & & \\ & \vdots & & & & & \\ & \mathbf{n}_{k-2}\mathbf{n}_k & -4(\mathbf{n}_{k-1}\mathbf{n}_k) & 6 & -4(\mathbf{n}_{k+1}\mathbf{n}_k) & \mathbf{n}_{k+2}\mathbf{n}_k & \\ & & & & \vdots & & \\ & & & & \mathbf{n}_{K-4}\mathbf{n}_{K-3} & 6 & -4(\mathbf{n}_{K-2}\mathbf{n}_{K-3}) \\ & & & & \mathbf{n}_{K-4}\mathbf{n}_{K-2} & -4(\mathbf{n}_{K-2}\mathbf{n}_{K-3}) & 5 \end{pmatrix} \tag{13}$$

Vectors $\mathbf{a}$ and $\mathbf{b}$ are given by

$$\mathbf{a} = \begin{pmatrix} (\mathbf{p}_0 - \mathbf{p}_1)\mathbf{n}_1 + (\mathbf{p}_1 - \mathbf{p}_2)\mathbf{n}_1 \\ \vdots \\ (\mathbf{p}_{k-1} - \mathbf{p}_k)\mathbf{n}_k + (\mathbf{p}_k - \mathbf{p}_{k+1})\mathbf{n}_k \\ \vdots \\ (\mathbf{p}_{K-3} - \mathbf{p}_{K-2})\mathbf{n}_{K-2} + (\mathbf{p}_{K-2} - \mathbf{p}_{K-1})\mathbf{n}_{K-2} \end{pmatrix} \tag{14}$$

and

$$\mathbf{b} = \begin{pmatrix} -[-2\mathbf{p}_0 + 5\mathbf{p}_1 - 4\mathbf{p}_2 + \mathbf{p}_3]\mathbf{n}_1 \\ -[\mathbf{p}_0 - 4\mathbf{p}_1 + 6\mathbf{p}_2 - 4\mathbf{p}_3 + \mathbf{p}_4]\mathbf{n}_2 \\ \vdots \\ -[\mathbf{p}_{k-2} - 4\mathbf{p}_{k-1} + 6\mathbf{p}_k - 5\mathbf{p}_{k+1} + \mathbf{p}_{k+2}]\mathbf{n}_k \\ \vdots \\ -[\mathbf{p}_{K-5} - 4\mathbf{p}_{K-4} + 6\mathbf{p}_{K-3} - 4\mathbf{p}_{K-2} + \mathbf{p}_{K-1}]\mathbf{n}_{K-3} \\ -[\mathbf{p}_{K-4} - 4\mathbf{p}_{K-3} + 5\mathbf{p}_{K-2} - 2\mathbf{p}_{K-1}]\mathbf{n}_{K-2} \end{pmatrix} \tag{15}$$

While vector $\mathbf{e}$ is

$$\mathbf{e} = \begin{pmatrix} \dfrac{\lambda_1 - \mathbf{p}_1\mathbf{n}_1}{\|\mathbf{n}_1\|^2} \\ \vdots \\ \dfrac{\lambda_k - \mathbf{p}_k\mathbf{n}_k}{\|\mathbf{n}_k\|^2} \\ \vdots \\ \dfrac{\lambda_{K-2} - \mathbf{p}_{K-2}\mathbf{n}_{K-2}}{\|\mathbf{n}_{K-2}\|^2} \end{pmatrix} \tag{16}$$

where $\lambda_k = c_k^1 n_k^x + c_k^2 n_k^y + c_k^3 n_k^z / c_k^0$ and the scalars c_k^0, c_k^1, c_k^2, c_k^3 are defined by Eq. (4) for each $\hat{\mathbf{p}}_k$. Finally, $\mathbf{I}$ is the $(K-2)\times(K-2)$ identity matrix.

Systematic numerical tests establish that the above projection method gives satisfactory results, e.g., in Fig. 5, discrete curvature plots are presented for a grid section projected using three different methods. The "NoSmoothing" plot shows the discrete curvature of the polygonal curve projected with $\gamma = 0$. The "MinLength" case uses $\gamma = 0.5$; this corresponds to the smoothing method [9] that exactly proposed "polygon smoothing" based on minimizing the "length functional" of Eq. (8). Finally, "MinLengthCurv" is the result of the new method using all functionals (7)–(9) with $\gamma = 0.5$. Clearly, the new method results in a projected grid section with a much smoother discrete curvature.

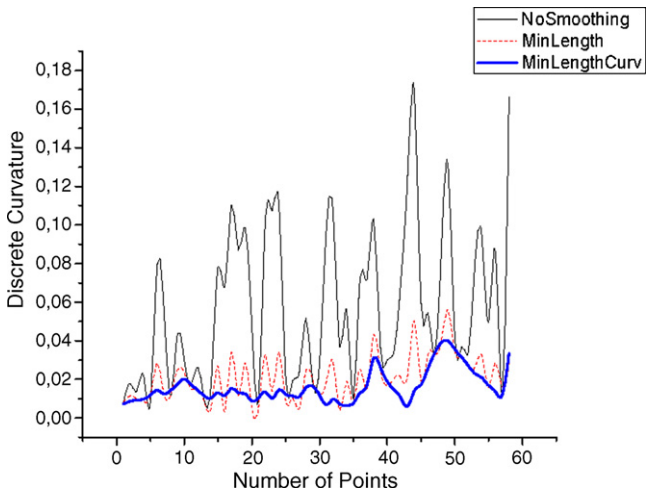

Fig. 5. The discrete curvature plots of a grid section.

6. Point-cloud subdivision

During the subdivision process described in Algorithm A1 the point-cloud has to be portioned into four subclouds (see step A1.5). This subdivision process must identify those cloud points which are sufficiently close to a DBS $\mathcal{A}^i(k+1)$ in order to include them with the corresponding subcloud $\mathcal{C}^i(k+1)$. The following procedure is employed to identify the subcloud $\mathcal{C}^i(k+1)$ corresponding to one DBS patch $\mathcal{A}^i(k+1)$:

Algorithm A4 (*Point-cloud subdivision according to a DBS patch* $\mathcal{A}^i(k+1)$).

A4.1 Project every point of $\mathcal{C}(k)$ onto the dynamic base surface $\mathcal{A}^i(k+1)$. The outcome is
A4.1.1 a point within the boundaries of $\mathcal{A}^i(k+1)$, or
A4.1.2 a point outside the boundaries of $\mathcal{A}^i(k+1)$, or
A4.1.3 no point at all.

A4.2 For each cloud point in $\mathcal{C}(k)$ corresponding to cases A4.1.2 or A4.1.3: a new projection point is calculated by projecting the cloud point onto the nearest DBS boundary.

A4.3 For each cloud point in $\mathcal{C}(k)$: measure its distance to its projection onto $\mathcal{A}^i(k+1)$. All cloud points with a distance *sufficiently small* (we use the heuristic *limit-value* = ''25% of the diagonal dimension of the bounding box of $\mathcal{A}^i(k+1)$'') are included in the subcloud $\mathcal{C}^i(k+1)$.

A4.4 Step A4.3 above produces subclouds which have a number of points in common, since a point may be ''sufficiently close'' to more than one $\mathcal{A}^i(k+1)$. So, this post-processing step reduces the point-sets overlapping according to the following criterion: if the distance between a point in $\mathcal{C}(k)$ and its projection point onto $\mathcal{A}^i(k+1)$ is *very small* (we use the heuristic *limit-value* = ''5% of the diagonal dimension of the bounding box of $\mathcal{A}^i(k+1)$'') then this point is considered to belong only to the subcloud $\mathcal{C}^i(k+1)$ and thus it is removed from any other subclouds at the same subdivision level.

Fig. 4c shows the result of applying Algorithm A4 to the point-cloud shown in Fig. 4b. The initial point-cloud is subdivided into four subclouds illustrated with different colours. Fig. 4c shows also the points of the ''black subcloud'' in the bottom-right overlapping with points on the ''blue subcloud'' above.

7. Examples and discussion

The vast majority of parameterization methods are tested for their accuracy and usefulness through practical applications including i.e., surface reconstruction, remeshing, texture mapping and others. In this section, the proposed method is tested by performing four different tasks with the resulting parameterizations. These tasks are described below.

7.1. Surface reconstruction by triangulation mapping

A standard 2D Delaunay triangulation algorithm [42] is employed to triangulate the point-cloud parameter values in the 2D rectangular domain. The resulting 2D triangulation is mapped onto the initial 3D point-cloud and the resulting triangulated surface is visually examined. With this test, one is able to verify the quality of a parameterization since bad parameter values will result to a false reconstruction of the 3D surface and, hence, to visual artefacts. *Note*: This task unifies all the different sub-parameterizations that are produced via Algorithm A2 into a global point-cloud parameterization.

The above evaluation method (a) has the disadvantage that it is based on ''visual examination'' of a 3D triangulation. The following methods (b) and (d) involve no user involvement, while (c) eases visual examination of 3D triangulations using texture mapping.

7.2. Planar development of the 3D triangulation obtained from method (a)

This method actually performs a re-parameterization in the planar domain [43]. The final result is algorithmically evaluated using the quality control tools in [21], eliminating the need for the user to visually examine any triangulation.

7.3. Texture mapping on the 3D triangulation from (a) using the planar development of (b)

In this way, accuracy shortcomings can be visualized using appropriate texture patterns like i.e., black circle disks in a white background. Parameterization inaccuracies will be noticeable by the deformation of the circle disks into ellipsoids.

7.4. Average distance error between the point-cloud and its projection onto the final DBS patches

If this error is sufficiently small, then the distribution of projected cloud-points does not differ much from that of the given cloud-points.

The first example is the shoe last point-set surface of Fig. 1a. The proposed recursive DBS method converged to an acceptable approximation after two levels of subdivision producing in-total 16 surface patches. Every DBS patch interpolates a grid of 11×11 points and approximates the associated subcloud with an accuracy of 0.1 mm.

Fig. 6a shows the outcome of the evaluation method (a). The obtained 3D mesh is smooth enough, as it is illustrated in the shaded image of Fig. 6b, despite the fact that no smoothing operations have been applied to the original point-cloud. Fig. 6c depicts the outcome of the evaluation method (b). The obtained 2D triangulation is tested against the 3D mesh: The colour map of Fig. 6d displays the length difference between every triangle in the planar development of Fig. 6c and its corresponding one in the 3D mesh; the result is very satisfactory as triangle distortion does not exceed 0.3%. Finally, Fig. 6e examines the accuracy of the obtained parameterization according to method (c): the outcome appears to be flawless indicating that the parameterization is good. ***Conclusion***: In this example, all evaluation criteria agree that the parameterization produced by the new DBS method is very good.

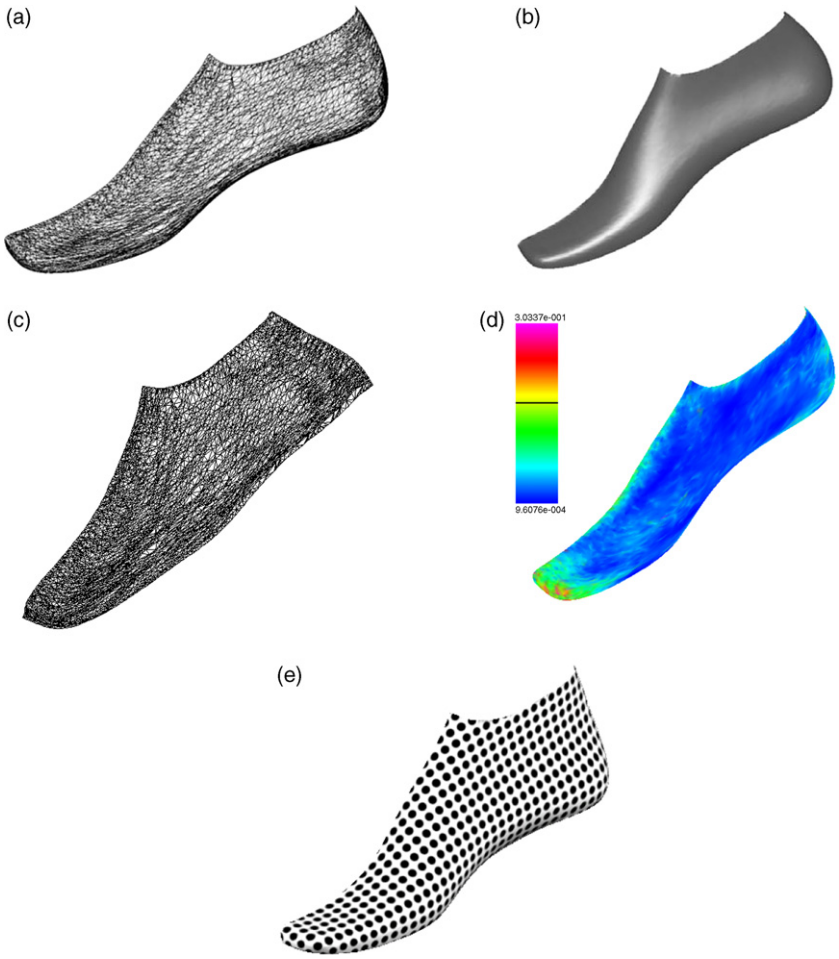

Fig. 6. Validating the parameterization of the point-cloud of a shoe last (Fig. 1a).

Similar results are obtained for the point-set surfaces ‘‘Bunny’’, ‘‘Horse’’ and ‘‘Human’’ shown in Fig. 7. All point-clouds have been parameterized locally using the proposed method. The ‘‘Bunny’’ point-cloud is approximated accurately after four levels of DBS subdivisions and with a maximum grid size equal to 10×10. The ‘‘Horse’’ and ‘‘Human’’ point-clouds required three levels of subdivision in order to approximate the local geometry with a mean error of 0.1 mm. Another interesting feature is that the resulted DBS patches of Fig. 7b and c preserve the symmetry of the underlying geometry.

In Fig. 8, the new parameterization method is applied on the ‘‘human head’’ point-cloud. Although the final 3D triangulation produced through evaluating method (a) seems smooth and accurate (Fig. 8b), application of the evaluation methods (d) and (c) show some accuracy problems in the areas with high curvature; see Fig. 8c and d, respectively. These results imply that although the produced DBS parameterization is adequate

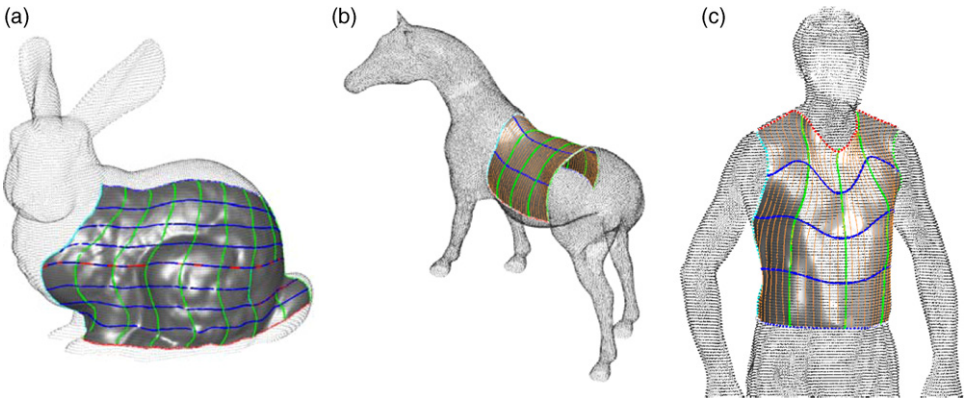

Fig. 7. The obtained dynamic base surfaces of various point-clouds.

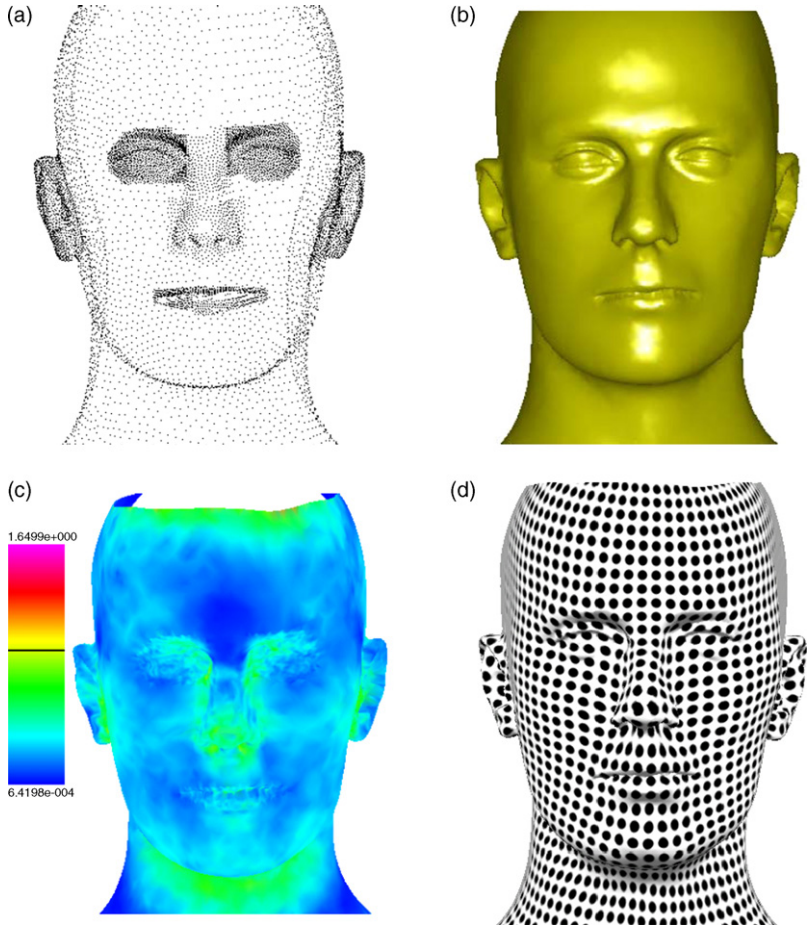

Fig. 8. The results of parameterizing the "human head" point-cloud.

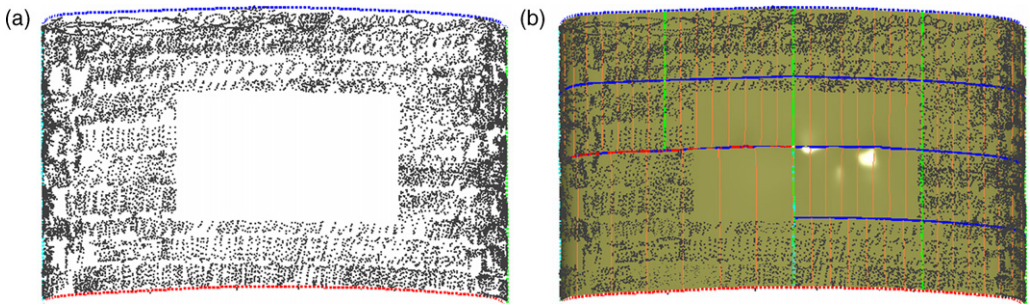

Fig. 9. Parameterizing a trimmed point-cloud surface.

for surface meshing it is necessary to examine alternative methods for surface flattening in order to derive a high-quality texture mapping result.

Finally, Fig. 9 illustrates the performance of the proposed method when applied in a trimmed surface with a large hole as it is shown in Fig. 9a. The result of the DBS method is given in Fig. 9b where one is able to note that the DBS patches are not very smooth in vicinity of the trimmed area without, however, affecting the point-cloud parameterization. Overall, the point-cloud is parameterized quite accurately since the resulting DBS surface approximates the point-cloud with an accuracy of 0.1 mm.

All algorithms, discussed in this paper, have been implemented on a Pentium 4 1.8 GHz machine with 512 MB of RAM. The execution time for the examples of this section varies from 24 s (shoe last) to 117 s ("Buny").

8. Conclusions

A vital prerequisite for efficiently using point-based surface models, either in standard NURBS-based CAD or in emerging cloud-based design systems, is availability of a robust algorithm to parameterize point-clouds. In this paper, a new parameterization method has been introduced based on

recursive dynamic base surfaces constructed using a quadtree structure and a novel grid-based surface-fitting technique. The new method is an improvement of the original one reported in [14] since it allows for a better parameterization of a point-cloud. With the new method it is possible to capture the geometry of small surface features in a point-cloud by dividing the approximating surface recursively. The new approach employs also a better point projection technique which results to smoother DBS patches.

The proposed method in not dealing with point-cloud smoothing or denoising. In fact, the presented concept presumes that the point-cloud is noisy (i.e., thick) and treats this noise a being part of the actual geometry. On the other hand, our previous research on dense point-clouds has proved that the error of the directed projection algorithm with noisy data is bounded [9]. Therefore, this paper focuses on producing smooth DBS patches rather than denoising the raw point data.

Our current research focuses on employing this new parameterization method to solve important open problems in cloud-based geometric modelling and CAD.

Acknowledgements

The human body data set of Fig. 8 was offered by Athens Technology Centre SA and originated in the European Anthropometric Database (Pilot Sizing Survey data/E-Tailor project, IST-1999-10549, funded by the European Commission). This data set was scanned at the Hohenstein Institute with a Human Solutions Vitus-Smart Scanner.

References

[1] M. Alexa, J. Behr, D. Cohen-Or, D. Levin, S. Fleishman, C.T. Silva, Point Set Surfaces in the Proceedings of the IEEE on Visualization, IEEE CS press, 2001, pp. 21–28.
[2] A. Adamson, M. Alexa, Approximating and Intersecting Surfaces from Points. Eurographics Symposium on Geometry Processing ACM International Conference Proceeding Series, 43 (2003) 230–239.
[3] F. Duguet, G. Drettakis, Flexible point-based rendering on mobile devices, IEEE CG&A 24 (4) (2004) 57–63.
[4] X. Guo, J. Hua, H. Qin, Scalar-function-driven editing on point set surfaces, IEEE CG&A 24 (4) (2004) 43–52.
[5] X. Guo, J. Hua, H. Qin, Touch-based haptics for interactive editing on point set surfaces, IEEE CG&A 24 (6) (2004) 31–39.
[6] S.C. Park, Y.C. Chung, Tool-path generation from measured data, Computer-Aided Design 35 (2003) 467–475.
[7] Y.F. Wu, Y.S. Wong, H.T. Loh, Y.F. Zhang, Modelling cloud data using an adaptive slicing approach, Computer-Aided Design 36 (3) (2004) 231–240.
[8] Y. Shon, S. Mcmains, Evaluation of drawing on 3D surfaces with haptics, IEEE CG&A 24 (6) (2004) 40–50.
[9] P. Azariadis, N. Sapidis, Drawing curves onto a cloud of points for point-based modelling, Computer-Aided Design 37 (1) (2005) 109–122.
[10] P. Azariadis, N. Sapidis, Product design using point-cloud surfaces, in: Proceedings of the 6th International Conference on Computer-Aided Industrial Design & Conceptual Design (CAID&CD 2005), Delft University of Technology, pp. 231–236.
[11] T. Várady, R.R. Martin, J. Cox, Reverse engineering of geometric models—an introduction, Computer-Aided Design 29 (4) (1997) 255–268.
[12] I. Horváth, Z. Rusák, Collaborative virtual design environments: collaborative shape conceptualization in virtual design environments, Communications of the ACM 44 (12) (2001) 59–63.
[13] K. Chua Chee, G.K.J. Gan, M. Tong, Interface between CAD and rapid prototyping systems. Part 1. A study of existing interfaces, The International Journal of Advanced Manufacturing Technology 18 (3) (1997) 566–570.
[14] P. Azariadis, Parameterization of clouds of unorganized points using dynamic base surfaces, Computer-Aided Design 36 (7) (2004) 607–623.
[15] P. Azariadis, N. Sapidis, Efficient parameterization of 3D point-sets using recursive dynamic base surfaces, LNCS 3746 (2005) 296–306.
[16] M.S. Floater, K. Hormann, Surface parameterization: a tutorial and survey, in: N.A. Dodgson, M.S. Floater, M.A. Sabin (Eds.), Advances in Multiresolution for Geometric Modelling, Springer-Verlag, Heidelberg, 2004, pp. 157–186.
[17] E. Catmull, A subdivision algorithm for computer display of curved surfaces. PhD thesis, University of Utah, 1974.
[18] S.D. Ma, H. Lin, Optimal texture mapping, Proceedings of the EUROGRAPHICS (1988) 421–428.
[19] J. Maillot, H. Yahia, A. Verroust, Interactive texture mapping, Proceedings of the SIGGRAPH (1993) 27–34.
[20] P. Azariadis, N. Aspragathos, On using planar developments to perform texture mapping on arbitrarily curved surfaces, Computers & Graphics 24 (4) (2000) 539–554.
[21] P. Azariadis, N. Sapidis, Planar development of free-form surfaces: quality evaluation and visual inspection, Computing 72 (1–2) (2004) 13–27.
[22] K. Hormann, G. Greiner, MIPS: an efficient global parametrization method, in: P.-J. Laurent, P. Sablonniere, L.L. Schumaker (Eds.), Curve and Surface Design: Saint-Malo, Vanderbilt University Press, Nashville, 1999–2000, pp. 153–162.
[23] K. Hormann, U. Labsik, G. Greiner, Remeshing triangulated surfaces with optimal parameterizations, Computer-Aided Design 33 (2001) 779–788.
[24] M. Eck, J. Hadenfeld, Local energy fairing of B-spline curves, in: G. Farin, H. Hagen, H. Noltemeier (Eds.), Computing, vol. 10, 1995, pp. 129–145.
[25] M.S. Floater, Parametrization and smooth approximation of surface triangulations, Computer-Aided Geometric Design 14 (3) (1997) 231–250.
[26] K. Hormann, G. Greiner, Hierarchical parameterization of triangulated surfaces., Proceedings of the Vision, Modeling, and Visualization (1999) 219–226.
[27] E.L. Schwartz, A. Shaw, E. Wolfson, A numerical solution to the generalized mapmaker's problem: flattening nonconvex polyhedral surfaces, IEEE Transactions on Pattern Analysis and Machine Intelligence 11 (9) (1989) 1005–1008.
[28] G. Zigelman, R. Kimmel, N. Kiryati, Texture mapping using surface flattening via multi-dimensional scaling, IEEE Transactions on Visualization and Computer Graphics 8 (2) (2002) 198–207.
[29] P.V. Sander, J. Snyder, S. Gortler, H. Hoppe, Texture mapping progressive meshes, in: Proceedings of the ACM SIGGRAPH' 2001, Computer Graphics Proceedings, Annual Conference Proceedings, pp. 409–416.
[30] P. Sander, S. Gortler, J. Snyder, H. Hoppe, Signal-specialized Parameterization, in: Proceedings of 13th Eurographics Workshop on Rendering, 2002, pp. 87–98.
[31] B. Levy, S. Petitjean, N. Ray, J. Maillot, Least squares conformal maps for automatic texture atlas generation, ACM Transactions on Graphics 21 (3) (2002) 362–371.
[32] M. Desbrun, M. Meyer, P. Alliez, Intrinsic parameterizations of surface meshes, in: Proceedings of the Eurographics 2002, Blackwell Publishing, Saarbrucken, G. Drettakis, H.-P. Seidel (Eds.), Computer Graphics Forum, 21 (3) (2002) 210–218.
[33] G. Yu, N.M. Patrikalakis, T. Maekawa, Optimal development of doubly curved surfaces, Computer Aided Geometric Design 17 (6) (2000) 545–577.

[34] C. Wang, S.-F. Chen, M. Yuen, Surface flattening for the fashion industry: a generic approach using spring–mass system, Computers in Industry 1548 (2001) 1–10.
[35] M.S. Floater, M. Reimers, Meshless parameterization and surface reconstruction, Computer Aided Geometric Design 18 (2) (2001) 77–92.
[36] Y. Lee, H.S. Kim, S. Lee, Mesh parameterization with a virtual boundary. Computers & Graphics 26 (5) (2002) 677–686 (special issue of the 3rd Israel–Korea Binational Conference on Geometric Modeling and Computer Graphics).
[37] W. Ma, P. Kruth, Parameterization of randomly measured points for least squares fitting of B-spline curves and surfaces, Computer-Aided Design 27 (9) (1995) 663–675.
[38] H. Pottmann, S. Leopoldseder, M. Hofer, Approximation with active B-spline curves and surfaces, in: Proceedings of the Pacific Graphics 2002, IEEE Computer Society, IEEE Press, New York (2002) 8–25.
[39] H. Hoppe, T. Derose, T. Duchamp, J. Mcdonald, W. Stuetzle, Surface reconstruction from unorganized points, Computer Graphics 26 (2) (1992) 71–78.
[40] N. Amenta, M. Bern, M. Kamvysselis, A new Voronoi-based surface reconstruction algorithm, in: Proceedings of the SIGGRAPH '98, Computer Graphics Proceedings (1998) 415–412.
[41] J.-D. Boissonnat, F. Cazals, Smooth surface reconstruction via natural neighbour interpolation of distance functions, Computational Geometry (2000) 223–232.
[42] J. Shewchuk, Triangle: engineering a 2D quality mesh generator and delaunay triangulator, in: M.C. Lin, Dinesh Manocha (Eds.), Applied Computational Geometry: Towards Geometric Engineering, LNCS 1148 (1996) 203–222.
[43] P. Azariadis, N. Aspragathos, Geodesic curvature preservation in surface flattening through constrained global optimization, Computer-Aided Design 33 (8) (2001) 581–591.

Dr. Philip N. Azariadis is an assistant professor with the Department of Product & Systems Design Engineering at the University of the Aegean. He holds a mathematics degree from the Department of Mathematics (1994) and a Ph.D. from the Mechanical Engineering & Aeronautics Department (1999) of the University of Patras. His research activities are focused in the areas of computer-aided design for manufacture, reverse engineering, computer graphics and robotics.

Nickolas S. Sapidis is currently an Associate Professor of ''computer-aided product design'' with the Department of Product and Systems Design Engineering of the University of the Aegean (Syros, Greece). He holds a diploma in Naval Architecture & Marine Engineering (1985), an M.A. in Applied Mathematics (1987) and an M.S. in Mechanical Engineering (1988). He received his Ph.D. in mechanical and aerospace sciences from the University of Rochester in 1993. N. Sapidis is on the editorial board of the international journals: *Virtual and Physical Prototyping*; *International Journal of Product Lifecycle Management*; *Computer-Aided Design and Applications*; *International Journal of Product Development*; *Computer-Aided Design*. He has also served as ''Guest Editor'' for other journals producing, up to now, four Special Issues focusing on geometric modelling, supply chain management and CAD education. He is a reviewer for 15 international journals, including all major journals in the fields of computer graphics and computer-aided design. N. Sapidis has been pursuing research on ''design of mechanical products'', ''computer-aided design (CAD)'', ''computer-aided engineering (CAE)'', ''solid modelling'', ''geometric modelling of surfaces'', ''geometric algorithms'', ''virtual engineering'', and ''computer graphics''. Focal points of his research have been: *working always on important industrial-problems and, at the same time, going "very deep" into the related mathematics to devise optimal algorithmic solutions*. He is the author of more than 40 papers on the above subjects. His research has been implemented in industrial CAD/CAE-software systems by the Massachusetts Institute of Technology (MIT) and companies including General Motors (GM), Intergraph and Tribon Solutions. N. Sapidis has taught at the Hellenic Air Force Academy, the National Technical University of Athens, the University of Athens and the Polytechnic University of Catalunya (Spain). Also, he has been providing education/training services to MIT as well as major Greek corporations like Elefsis Shipyards and the Bank of Greece. His industrial experience, on CAD/CAE research/development/application, includes GM R&D Center and GM Design Center (USA) as well as Marine Technology Development Co (Greece). N. Sapidis has edited (co-edited) the two volumes: (1) P. Kaklis & N. Sapidis (Eds.), *Computer Aided Geometric Design: From Theory to Practice*, National Technical University of Athens, Athens, 1996; (2) N. Sapidis (Ed.), *Designing Fair Curves and Surfaces: Shape Quality in Geometric Modeling and CAD*, Society for Industrial and Applied Mathematics (SIAM), Philadelphia, USA, 1994, served on 15 international-conference program-committees, and was an ''invited Keynote Speaker'' for many international conferences. N. Sapidis has given ''invited lectures'' at numerous conferences, Universities and corporations in Asia, Europe and the US.