Int J Interact Des Manuf (2015) 9:303–316
DOI 10.1007/s12008-015-0276-1

ORIGINAL PAPER

Triangular mesh parameterization with trimmed surfaces

Oscar E. Ruiz[1] · **Daniel Mejia**[1] · **Carlos A. Cadavid**[1]

Received: 1 April 2015 / Accepted: 17 April 2015 / Published online: 28 April 2015

Abstract Given a 2-manifold triangular mesh $M \subset \mathbb{R}^3$, with border, a parameterization of M is a FACE or trimmed surface $F = \{S, L_0, \ldots, L_m\}$. F is a connected subset or region of a parametric surface S, bounded by a set of LOOPs $L_0, \ldots, L_m$ such that each $L_i \subset S$ is a closed 1-manifold having no intersection with the other L_j LOOPs. The parametric surface S is a statistical fit of the mesh M. L_0 is the outermost LOOP bounding F and L_i is the LOOP of the i-th hole in F (if any). The problem of parameterizing triangular meshes is relevant for reverse engineering, tool path planning, feature detection, re-design, etc. State-of-art mesh procedures parameterize a rectangular mesh M. To improve such procedures, we report here the implementation of an algorithm which parameterizes meshes M presenting holes and concavities. We synthesize a parametric surface $S \subset \mathbb{R}^3$ which approximates a superset of the mesh M. Then, we compute a set of LOOPs trimming S, and therefore completing the FACE $F = \{S, L_0, \ldots, L_m\}$. Our algorithm gives satisfactory results for M having low Gaussian curvature (i.e., M being quasi-developable or developable). This assumption is a reasonable one, since M is the product of manifold segmentation pre-processing. Our algorithm computes: (1) a manifold learning mapping $\phi : M \rightarrow U \subset \mathbb{R}^2$, (2) an inverse mapping $S : W \subset \mathbb{R}^2 \rightarrow \mathbb{R}^3$, with W being a rectangular grid containing and surpassing U. To compute ϕ we test IsoMap, Laplacian Eigenmaps and Hessian local linear embedding (best results with HLLE). For the back mapping (NURBS) S the crucial step is to find a control polyhedron P, which is an extrapolation of M. We calculate P by extrapolating radial basis functions that interpolate points inside $\phi(M)$. We successfully test our implementation with several datasets presenting concavities, holes, and are extremely non-developable. Ongoing work is being devoted to manifold segmentation which facilitates mesh parameterization.

Keywords Triangular mesh parameterization · Trimmed surface · Manifold learning · NURBS · RBFs

Abbreviations

LOOP	Closed (piecewise linear or smooth) curve lying on a surface, and bounding a connected region on the surface. In this manuscript, LOOPs are denoted with Γ or γ
B-REP	Boundary representation
HLLE	Hessian locally linear embedding
NURBS	Non-uniform rational B-spline
RBF	Radial basis function
M	Triangular mesh (with boundary), composed by the set of triangles $T = \{t_1, t_2, \ldots, t_q\}$ with vertex set $X = \{x_1, x_2, \ldots, x_n\}$ ($X \subset \mathbb{R}^3$)
∂M	Boundary of M, whose connected components are LOOPs ($\partial M = \{\Gamma_0, \Gamma_1, \ldots, \Gamma_k\}$)
ϕ	An homeomorphic map $\phi : M \rightarrow \mathbb{R}^2$, implemented here for dimensional reduction or manifold learning. $\phi_{Isom}()$, $\phi_{Lapl}()$, $\phi_{HLLE}()$, are the Isomap, Laplacian Eigenmap and Hessian locally linear embedding implementations, respectively. $\phi()$ is called *forward* map in this manuscript
U	$U = \{u_1, u_2, \ldots, u_n\}$ is the parametric image of vertices of M ($U = \phi(X)$, $U \subset \mathbb{R}^2$)

✉ Oscar E. Ruiz
oruiz@eafit.edu.co

[1] Laboratorio de CAD CAM CAE, Universidad EAFIT, K49, 7-sur-50, 050022 Medellín, Colombia

$\partial(\phi(M))$	Boundary of the parametric image of M. For the sake of simplicity, we assume that $\partial(\phi(M)) = \phi(\partial(M)))$
γ_i	i-th LOOPs of $\partial(\phi(M))$
λ_i	Re-sampling of a LOOP γ_i
W	Rectangular grid in R^2 such that U lies in the convex hull of W
$H(W)$	Rectangular point set in R^2 being the convex hull of W
P	Rectangular grid in $\mathbb{R}^3$ being the control polyhedron for the parametric surface f
f	Function $f : W \rightarrow \mathbb{R}^3$ produces P the control polyhedron of S ($P = f(W)$) by calculating an extrapolation of M in $\mathbb{R}^3$
S	$S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ is a parametric surface which approximates and extends M in $\mathbb{R}^3$. $S()$ is called *backward* map in this manuscript. To simplify notation, S refers here to both: (1) the parametric mapping (i.e., $S()$) and (2) the set of points product of the mapping $S()$ (i.e., $S(H(W)) = \{S(w_1, w_2)\|(w_1, w_2) \in H(W)\}$)
L_i	Trimming curve in $M \subset \mathbb{R}^3$ defined as $L_i = S(\lambda_i)$
F	Trimmed surface (FACE) such that $F = (S, \{L_0, L_1, \ldots\})$
∂F	Boundary of F approximated by the union of all L_i

1 Introduction

In this manuscript, M denotes a triangle-based mesh, which is a 2-manifold with border. Without loss of generality, we consider M as the result of segmenting a larger triangular mesh. M presents a low curvature (i.e., M is near-developable). Therefore, M can also be referred to as a *sub-mesh*.

Being a 2-manifold, M admits a 2-variable parameterization, which is a homeomorphism between M and a connected subset of R^2. However, to be usable in reverse engineering, CAD, CAM, or visualization, the parameterization must be accompanied by a *trimmed FACE*, which approximates the triangular mesh M. A *trimmed FACE* consists of a parametric surface $S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ and a set of LOOPs or closed curves $\Gamma_i \subset M$, which bound a connected sub-region of M.

Trimmed surfaces are indispensable in boundary representations and therefore in computer aided geometric design. In a trimmed surface, $S()$ is usually based on a rectangular control polyhedron P. Triangular topology for P is very unusual because it produces undefinition in tangent and normal vectors, rendering $S()$ and FACE F unusable.

M is a connected triangular mesh $M = (X, T)$, with a border described by the set of LOOPs $\{\Gamma_1, \Gamma_2, \ldots, \Gamma_m\}$. The Γ_i LOOPs are piecewise linear closed 1-manifolds, which do not intersect each other. Given M, a trimmed parametric surface $F = (S, \{L_0, L_1, \ldots, L_m\})$ is pursued, with S being a smooth surface that approximates a (conveniently defined) superset of M. The set of curves $\{L_1, L_2, \ldots, L_m\}$ of the trimmed parametric surface F approximates the boundary ∂M on the surface S.

In this article we propose a procedure for computing the trimmed surface F, as follows: (1) compute a mapping $\phi : M \subset \mathbb{R}^3 \rightarrow U \subset \mathbb{R}^2$ which describes a 2D parameter space for M. (2) Compute a back-mapping (NURBS or RBFs) $S : W \rightarrow \mathbb{R}^3$, with W being a rectangular region in $\mathbb{R}^2$, s.t. W is superset of U. (3) Compute via $S()$ a FACE boundary approximating ∂M.

The remainder of the article is organized as follows: Sect. 2 discusses the connection between interactive design and the problem addressed in this article. Section 3 reviews the relevant literature. Section 4 describes the implemented methodology. Section 5 presents and discusses some results. Section 6 concludes the paper and introduces what remains for future work.

2 Interactive design and mesh parameterization with trimmed surfaces

There are two basic relations of mesh parameterization using trimmed surfaces with interactive design and manufacturing: (1) mesh parameterization being a pre-condition for downstream interactive design or manufacturing processes. (2) Automated mesh parameterization supporting human interaction for design or manufacturing processes. The present article fits into the second relation (Sect. 2.1). However, a short list of interactive processes requiring parameterized surfaces appears in Sect. 2.2.

2.1 Human interaction and mesh parameterization

We consider interactive mesh parameterization with trimmed surfaces a process in which algorithms (as the ones presented here) support human interaction and decisions to achieve such parameterizations. Our algorithms, however, do not close the subject as human interaction is additionally required to: (1) segment a 2-manifold triangular mesh into smaller meshes (*exampli gratia* M above), until the sub-meshes M are nearly developable. (2) Synthesize an underlying parametric surface $S()$ which approximates some M_S super-set of M ($M \subset M_S$).

Mesh segmentation [process (1) above] requires a human expert decision, since this partition includes diverse criteria (e.g., design, manufacturing, analysis, esthetic). Mesh segmentation governs the topological partition of the boundary representation B-Rep of the piece and the geometries (carrier curves and surfaces) underlying such topologies. The

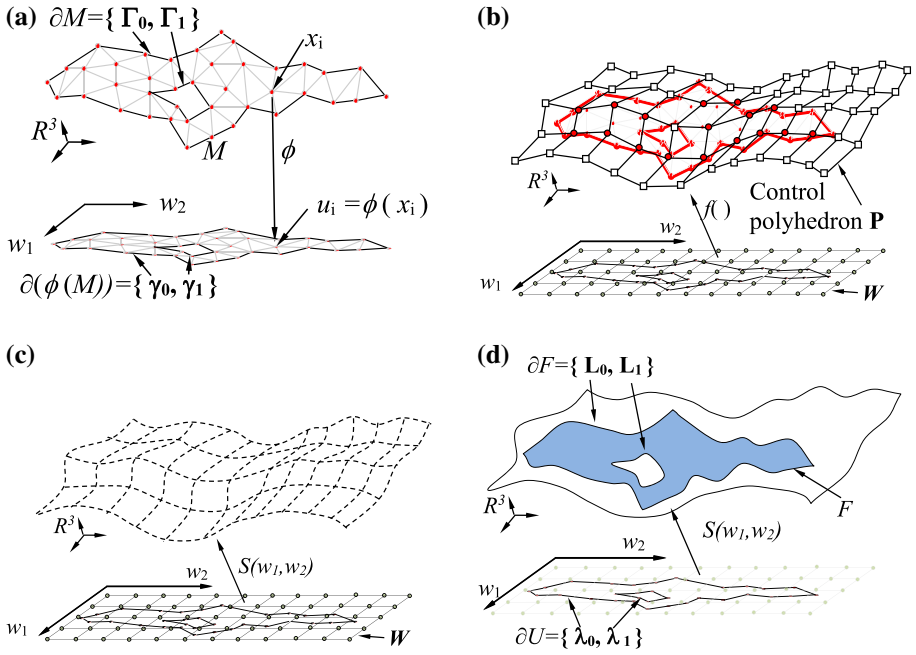

Fig. 1 Parameterization of mesh M with a trimmed surface (FACE F). **a** Determination of homeomorphism $\phi : M \rightarrow \mathbb{R}^2$ and mapping of boundaries of M (∂M). **b** Synthesis of control polyhedron $P = f(W)$. **c** Calculation of carrier surface S. **d** Calculation of boundaries of FACE F, ∂F

synthesis of S [process (2) above] implies a very under-constrainted problem, since a superset M_S of M must be defined, with the condition of M_S being approximately a 'rectangular' over-extension of M. This process presents a clear need for an interactive human input to repair degenerate surfaces, improve the control polyhedron P, etc.

2.2 Interactions enabled by mesh parameterization

Parametric trimmed surfaces are pre-condition for a number of interactive processes, such as interactive model modification in reverse engineering (Ref. [1]), interactive re-meshing for CAE (Ref. [2]), interactive mesh morphing for visualization (Ref. [3]), interactive visualization of complex surfaces (Ref. [4]), interactive processing of surface—embedded images (Ref. [5]), rendering textures on a surgery simulator (Ref. [6]), interactive segmentation and parameterization for mesh quadrangulation (Ref. [7]), finite element isogeometric interactive analysis (Refs. [8,9]), and others.

3 Literature review

In most engineering applications, the sole parameterization of the triangular mesh M does not suffice, and a trimmed parametric surface $(S, \{L_0, L_1, \ldots\})$ approximating M is required. The first process requires a forward function $\phi : \mathbb{R}^3 \rightarrow \mathbb{R}^2$. The second process requires, in addition, a backward mapping $S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$.

When parameterizing triangular meshes, the parameterization of the individual triangles may be sufficient whenever no parametric space common to *all* triangles in the mesh is required. For example, smooth parametric surfaces fit to separate triangular control polyhedra give as result C^0-, C^1-, or C^2-smooth triangular patches [10–16], which do not share a common parametric space. In contrast, Ref. [17] presents a geodesic-based Piecewise Linear parameterization common to the complete triangular mesh M. Although very intuitive, this approach still presents undesirable intersection of PL geodesics in surfaces presenting high Gauss—curvature.

3.1 Dimensional reduction $\phi : \mathbb{R}^3 \supset M \longrightarrow \mathbb{R}^2$

This section discusses the existing algorithms for dimensional reduction $\phi : M \rightarrow \mathbb{R}^2$ (Fig. 1a), which find a 2D underlying parameter space in the 2-manifold triangular mesh $M \subset \mathbb{R}^3$. This parameter space U is the image of the mesh points X under the map ϕ (i.e., $U = \phi(X)$). The algorithms for dimensional reduction may be classified according to the properties that they preserve, as they synthesize (ϕ, U):

1. *Angle-preserving algorithms* Conformal maps seek to preserve the angles formed by intersecting curves. The function ϕ is devised to minimize angle deformations [7,18–21]. An angle-preserving ϕ would not, in general, preserve areas or lengths, causing a strong warping in the back-mapping S.
2. *Area-preserving algorithms* An authalic map $\phi :\rightarrow \mathbb{R}^2$ would satisfy $A(t_i) = A(\phi(t_i))$ for triangle $t_i \in T$. Area distortions are minimized over fixed 2-Dimensional primitives (e.g., disk or rectangle [22,23]). Authalic maps

may result in large angle distortions which in turn would produce important distortions in the back-mapping S.

3. *Distance-preserving algorithms* An isometric map $\phi :\rightarrow \mathbb{R}^2$ would preserve distance among points. This means that $d_M(p,q) = d(\phi(p),\phi(q))$, for $p,q \in M$, with $d_M()$ being a distance measured on M (Refs. [24–28]). Isometric maps do not present distortions in the back-mapping $S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$. However, unlike angle- and area-preserving ones, they require M to be highly *developable* (i.e., with low Gaussian curvature).
Surfaces which are strongly *non-developable* may be partitioned into smaller, more *developable* ones. The issue of Manifold Segmentation, however, is outside the scope of the present manuscript, and is the subject of our future efforts.

3.2 Parametric surface $S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$

Since this article aims to parameterize a mesh M with a trimmed surface or FACE (in CAD sense), the dimensional reduction $\phi : M \rightarrow \mathbb{R}^2$ must be followed by the computation of a parametric surface $S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ and a connected subset within it, which resembles M (with concavities, holes, etc.). This goal is illustrated in Fig. 1d.

Reference [5] describes a method in which an isometric rectangular surface is computed, but requires the input mesh to be already of this rectangular nature (i.e., a triangular mesh of a rectangular patch) which strongly constraints the algorithm.

Reference [8] presents an initial parameterization of $\phi : M \rightarrow U$. The Coons back mapping S has domain in a subset of the parametric space U, which cannot present holes or concavities. As a result, S can only fit a rectangular subset of M, leaving out the possibility of M having holes or concavities.

3.3 Conclusions of literature review

Our Literature Review has found several approaches for mesh parameterization. However, they do not address meshes with concavities and holes. To enable them, it is necessary to complement the sole mesh parameterization with the synthesis of a trimmed surface, which smooths M and expresses the holes and concavities, besides producing a parameterization. Consequently, we present here our approach, in which a triangular mesh (with boundary and holes) $M \subset \mathbb{R}^3$ is approximated by a trimmed surface $F = (S, \{L_1, L_2, \ldots, L_m\})$. First, a parameterization U of M is computed using dimensional reduction (Fig. 1a). Then, a rectangular superset of U, $H(W)$, is mapped back to $\mathbb{R}^3$, via a parametric surface S (Fig. 1b): $H(W) \longrightarrow \mathbb{R}^3$ which fits a superset of M (Fig. 1c). FACE boundaries are drawn on S, to trim a FACE, with holes, which resembles M (Fig. 1d). The calculation of a superset of M in $\mathbb{R}^3$ is, of course, a very sensitive operation, for which we apply a combination of NURBs and radial basis functions.

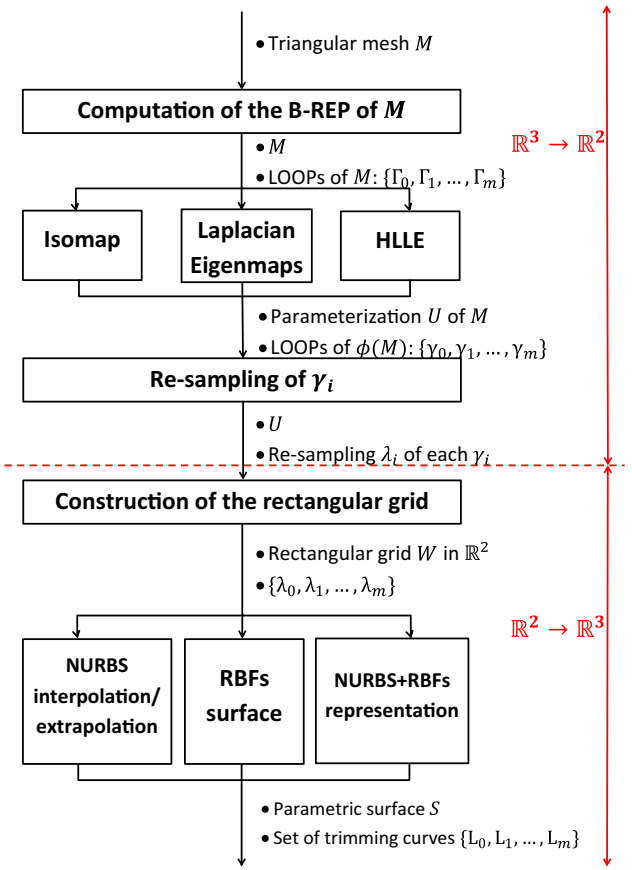

Fig. 2 Construction of a trimmed parameterization of the triangular mesh M. $\mathbb{R}^3 \rightarrow \mathbb{R}^2$ mapping (*up*) and $\mathbb{R}^2 \rightarrow \mathbb{R}^3$ back-mapping (*down*)

4 Methodology

Consider $M = (X, T)$, a connected 2-manifold in $\mathbb{R}^3$ with holes and border. The set of all LOOPs bounding M is noted as $\partial M = \{\Gamma_0, \Gamma_1, \ldots, \Gamma_m\}$. Our goal is to find a FACE F or trimmed surface $F = (S, \{L_0, L_1, \ldots, L_m\})$, composed by a parametric surface $S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ and a set of boundaries on S ($\partial F = \{L_0, L_1, \ldots, L_m\}$ such that $\partial M \approx \partial F$).

In order to achieve this goal, we follow the procedure in Fig. 3, which appears as a data flow in Fig. 2.

1. *Computation of the B-REP of M* In this pre-processing, a boundary representation of the triangular mesh M is computed in order to extract each of the LOOPs Γ_i that bound M.
2. *Computation of the mapping $\phi : M \rightarrow \mathbb{R}^2$* (Fig. 1a) A manifold learning algorithm is applied in this step in order to extract the parameterization U that characterizes the manifold and each LOOP γ_i in the parameter space. The

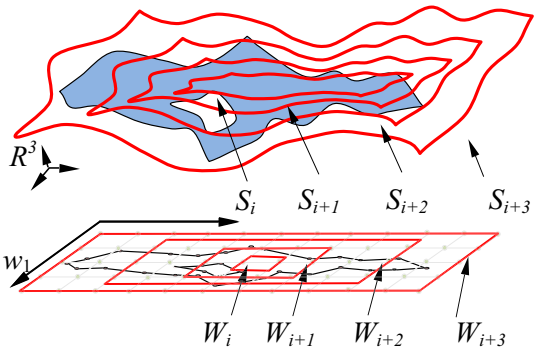

Fig. 3 Stretching of control polyhedron P by iterative extrapolation

procedure was tested using: (1) Isomap, (2) Laplacian Eigenmaps or (3) HLLE.

3. *Construction of the rectangular grid* $W \subset \mathbb{R}^2$ (Fig. 1b) A rectangular grid W is built in $\mathbb{R}^2$ such that U lies inside the convex hull of W ($U \subset H(W)$).
4. *Synthesis of a control polyhedron* P *for* S (Fig. 1b) A control polyhedron $P = f(W)$ is required for the parametric surface S of the Trimmed FACE F. In Fig. 1b, P is represented as a grid of round-icon and square-icon vertices. Round-icon vertices $f(w_1, w_2)$ fall inside M since $(w_1, w_2) \in U$. Square-icon vertices fall outside M since $(w_1, w_2) \notin U$, and therefore $f(w_1, w_2)$ must be estimated. A radial basis function $f : W \longrightarrow \mathbb{R}^3$ is created by using the condition $f(U) = X$, since the pairs $(x_i, u_i) = (x_i, \phi(x_i))$ are known. $f(H(W))$is then used to extrapolate M as needed.
5. *Representation of* M *by a trimmed surface* $F = (S, \{L_0, \ldots L_m\})$ Once the control polyhedron P for S is estimated, the actual calculation of the parametric surface S proceeds using a NURBs formulation. The boundary of F, $\partial F = \{L_0, \ldots L_m\}$, is achieved as $L_i = f(\lambda_i)$, where λ_i is the re-sampling of the straight-edge LOOP γ_i.

The algorithms implemented for the above procedure are briefly discussed below.

4.1 Dimensionality reduction (manifold Learning)

To synthesize $F = (S, \{L_0, L_1, \ldots\})$, first a parameterization (ϕ, U) underlying the mesh M must be found (Fig. 2). Such parameterization must be a homeomorphism $\phi : M \longrightarrow \mathbb{R}^2$, whose existence is guaranteed by the 2-manifold character of M. In addition, the image $U = \phi(X) \subset \mathbb{R}^2$ is itself a manifold. Notice that ϕ affects the geometry X of M and not its topology T. To find U, a dimensionality reduction (manifold learning) algorithm is applied. We implemented and tested three specific algorithms for ϕ: (1) Isomap, (2) Laplacian Eigenmaps and (3) HLLE, as described next.

4.1.1 Isometry-based parameterization

Isometry-based parameterizations assume that geodesic distances on M between any two vertices $x_i, x_j \in X$ are preserved (i.e., $dist_M(x_i, x_j) = dist_{\mathbb{R}^2}(\phi(x_i), \phi(x_j))$). Isomap (Ref. [28]) constructs a graph-based estimator for the geodesic distance on M. The geodesic distance $(G_{i,j})$ between any two mesh vertices x_i, x_j is approximated by the shortest path between them in the triangulation graph T. where $dist_M()$ and $dist_{\mathbb{R}^2}()$ corresponds to the euclidean distance measured on M-geodesics and on $\mathbb{R}^2$, respectively. The Ref. [28] then applies classical multidimensional scaling (CMDS) on G for computing U, by solving Eq. (1) using a singular value decomposition of G:

$$G = U^T U \tag{1}$$

A weakness of using T-graph connectivity for estimation of geodesic paths is that in regions with concavities or holes in M (and therefore absent from T), disable graph-based estimation of geodesics. To evidence this disadvantage of Isomap, we created an M data set (Fig. 9a) with the shape of an 'S' letter drawn onto a Cone surface. Because a cone is a fully developable surface (Gauss curvature is 0 everywhere), the geodesics on a cone are very well defined. However, the 'S' data set causes the graph-based geodesic estimation to fail, because vertices very near on the cone *appear* to be very distant using paths restricted to the 'S' data set. Our investigation uses the Floyd's shortest-path algorithm to estimate the geodesic distances in M.

4.1.2 Laplacian Eigenmaps-based parameterization

Laplacian Eigenmaps (Ref. [20]) is a Dimensionality Reduction algorithm which runs faster than Isomap, but distance preservation is not pursued. Laplacian Eigenmaps poses the optimization problem in Eq. (2).

$$\min \sum_{i,j} a_{ij} \| u_i - u_j \|^2 \tag{2}$$

with a_{ij} being the adjacency weight between the vertices x_i and x_j in M. D is a diagonal matrix defined as $d_{ii} = \sum_j a_{ij}$ and L is a symmetric matrix defined as $L = D - A$. Then, Eq. (2) can be solved (under adequate constraints) by computing the following eigenproblem:

$$LU = \Lambda DU \tag{3}$$

where Λ is a diagonal matrix containing the eigenvalues of L. The matrix L is known as the Laplacian of the mesh graph because it is closely related to the Laplace–Beltrami operator on manifolds.

4.1.3 HLLE-based parameterization

Hessian linear local embedding (HLLE, Ref. [29]) assumes that the Hessian of an (unknown) parametric space is the same as the Hessian defined on the tangent space of the manifold. Following this idea, HLLE builds the Hessian functional $\mathcal{H}$ on such manifold:

$$\mathcal{H}(\psi) = \int_{\mathcal{M}} \left\| H_x^{tan}(\psi) \right\|_F^2 \tag{4}$$

where ψ is a function defined on the manifold $\mathcal{M}$, $\|\cdot\|_F$ is the Frobenius norm and $H_x^{tan}(\psi)$ is the tangent Hessian of ψ i.e.:

$$H_x^{tan}(\psi) = \begin{bmatrix} \frac{\partial^2 \psi}{\partial \xi_1^2} & \frac{\partial^2 \psi}{\partial \xi_1 \xi_2} \\ \frac{\partial^2 \psi}{\partial \xi_2 \xi_1} & \frac{\partial^2 \psi}{\partial \xi_2^2} \end{bmatrix} \tag{5}$$

with ξ_i being the local coordinates of the tangent space at x.

A discretization of the integral in Eq. (4) is then posed:

$$\mathcal{H}(\psi) = \sum_{i=1}^{n} \int_{\mathcal{X}_i} \omega_i \left\| H_x^{tan}(\psi) \right\|_F^2 \approx \boldsymbol{\psi}' K \boldsymbol{\psi} \tag{6}$$

with $\mathcal{X}_i$ being a portion of the surface containing x_i, ω_i is a weighting function, $\boldsymbol{\psi} = [\psi_1, \psi_2, \ldots, \psi_n]^T$ is a vector of the function ψ evaluated at each x_i, and K is a positive semidefinite matrix that approximates $\mathcal{H}$ on M.

The matrix K (known also as the Hessian kernel) is estimated by computing the local tangent plane at each x_i (Ref. [29]). Finally, the kernel of $\mathcal{H}$ (i.e., $\{\psi | \mathcal{H}(\psi) = 0\}$) is spanned by the constant function and a pair of (linearly independent) linear functions (which the authors choose to parameterize M). Given that K is just an approximation of the functional $\mathcal{H}$, a minimization problem is posed in order to estimate the kernel of $\mathcal{H}$:

$$\min \boldsymbol{\psi}^T K \boldsymbol{\psi} \tag{7}$$

By imposing $\langle \boldsymbol{\psi}, \boldsymbol{\psi} \rangle = 1$, an eigenproblem arises involving K. Since the eigenvalues of K are the local minima of Eq. (7) and their corresponding eigenvectors are the functions that achieve such minima, the desired parameterization is found by setting U equal to the second and third eigenvector of K (the first eigenvector corresponds to the constant function which collapses every vertex to a single point).

4.2 Surface representation

The purpose of this investigation is not only parameterize M. Further, one wishes to find a trimmed surface $(S, \{L_0, L_1, \ldots\})$ which approximates M and its border in a smooth manner. The surface S is an over-extension of a smoothing of M. This very loose specification for S is precisely the difficulty behind finding a trimmed surface to approximate M.

The surface $S : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$ has the form in Eq. (8).

$$S = S(w_1, w_2) = \begin{bmatrix} x(w_1, w_2) \\ y(w_1, w_2) \\ z(w_1, w_2) \end{bmatrix} \tag{8}$$

where $x(), y(), z()$ are parametric forms used in CAGD. For this article, we use NURBs and radial basis functions. The initial mesh M is smoothly approximated by a subset of S. S extends beyond the boundaries of M because it must be well defined also where M does not exist (holes, concavities, etc.).

To define S, in this manuscript we present three alternatives: (1) To use NURBs representation. The full control polyhedron P is built by starting with a small grid, well defined in a very reduced rectangular neighborhood of M. This grid is iteratively enlarged until it covers M and its holes and concavities, using the information in M to lock P in the $\mathbb{R}^3$ space. (2) To use a radial basis function based on the known couples $(u_i, x_i), i = 1, 2, \ldots$ since $u_i = \phi(x_i)$. (3) To use a NURBs surface, whose control polyhedron is calculated with a radial basis function.

A control polyhedron P is required in cases (1) and (3) above. Let us assume that the parameterization $\phi(X) = U \subset \mathbb{R}^2$ has been found. To obtain P, we first compute a rectangular point grid $W \subset \mathbb{R}^2$, whose convex hull $H(W)$ covers U (Figs. 1b, 4b). A function $f() : W \longrightarrow \mathbb{R}^3$ (yet to determine) will compute the control polyhedron $P = f(W)$ for the parametric surface S. It is easy to determine $f(u)$ for $u \in \phi(M)$ since in such a region the function $f()$ basically inverts $\phi()$. However, outer points $(u \in W, u \notin \phi(M))$ do not have an obvious $f(u)$ image in $\mathbb{R}^3$. The determination of $f()$, for the cases (1) NURBs, (2) RBFs and (3) NURBs + RBFs is described next.

4.2.1 NURBS interpolation/extrapolation

In this approach (Fig. 3), the synthesis a parametric surface S approximating a superset of M is carried out iteratively. A parametric space W_i, its control polyhedron P_i and its surface S_i are stretched until $W_i = W$. The control polyhedron P_{i+1} in iteration $i + 1$ is calculated by extrapolating $S_i()$ (i.e., calculating $S_i(w_1, w_2)$ with (w_1, w_2) being beyond the parametric space W_i). The first parameter space, W_0 is sufficiently small for the control polyhedron P_0 to include only points in M (i.e., no extrapolation of S is needed).

A parametric surface of the NURBs type has the form in Eq. (9).

$$S(w_1, w_2) = \sum_{ij} \alpha_i(w_1) \cdot \beta_j(w_2) p_{ij} \tag{9}$$

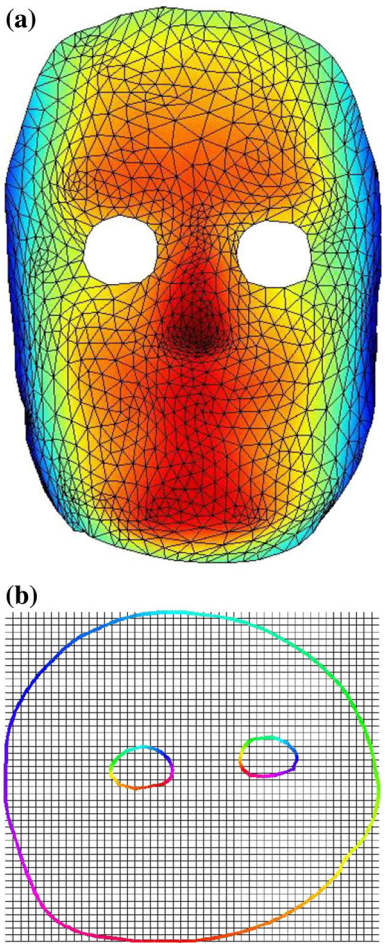

Fig. 4 *Mask* dataset M and parametric grid W associated with Hessian local linear embedding (ϕ). **a** *Mask* dataset $M \subset \mathbb{R}^3$. **b** Border of $\phi(M) \subset R^2$ with ϕ = HLLE. Border $\partial(\phi(M)) = \{\gamma_0, \gamma_1, \gamma_2\}$ (3 LOOPs). Rectangular parameter grid W covering $\phi(M)$

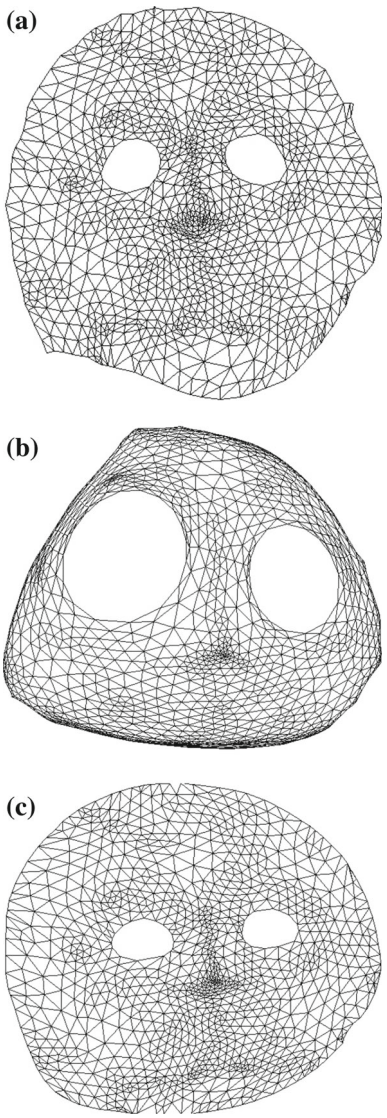

Fig. 5 Parameterization ϕ of the *Mask* dataset. *Techniques* IsoMap, Laplacian Eigenmap, Hessian locally linear embedding. **a** Isomap parameterization. **b** Laplacian Eigenmaps parameterization. **c** HLLE parameterization

where (w_1, w_2) are parameters. $\alpha_i()$ and $\beta_j()$ are the weight functions. $P = \{p_{ij}\}$ is the control polyhedron. We present here an iterative extrapolation to obtain P for a S surface of NURBs type.

1. Locate a subgrid $W_i = W_0$ inside W such that $\phi^{-1}(W_0)$ lies inside M.
2. Compute P_i using the barycentric coordinates of W_i.
3. Compute the parametric surface S_i from the control polyhedron P_i.
4. Extend W_i to obtain an enlarged domain W_{i+1}.
5. For each new pair w in $W_{i+1} - W_i$: if $\phi^{-1}(w)$ falls in M, the corresponding control point is $\phi^{-1}(w)$. Otherwise, the control point is the extrapolation $S_i(w)$ of S_i (w is outside the domain W_i).
6. Repeat 4–5 until $W_i = W$.
7. Compute the parametric surface S using the control polyhedron P using Eq. (9).

4.2.2 Radial basis function surface

A RBFs surface $S : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$ has the form in Eq. (10).

$$
\begin{aligned}
x(w_1, w_2) &= \sum_{i=1}^{n} \alpha_{ix} \Phi_{ix} (\parallel (w_1, w_2) - u_i \parallel) + R_{ix}(w_1, w_2) \\
y(w_1, w_2) &= \sum_{i=1}^{n} \alpha_{iy} \Phi_{iy} (\parallel (w_1, w_2) - u_i \parallel) + R_{iy}(w_1, w_2) \\
z(w_1, w_2) &= \sum_{i=1}^{n} \alpha_{iz} \Phi_{iz} (\parallel (w_1, w_2) - u_i \parallel) + R_{iz}(w_1, w_2)
\end{aligned}
\tag{10}
$$

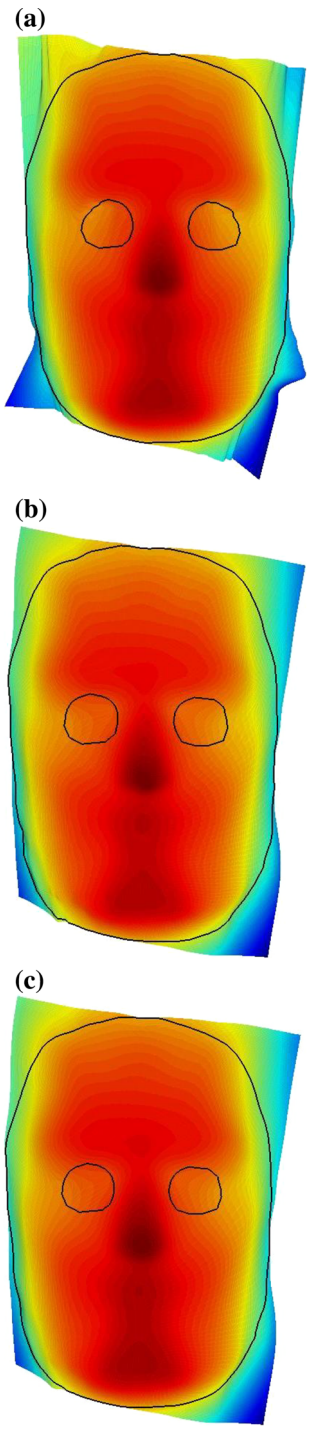

Fig. 6 Trimmed surfaces $F = \{S, L_0, L_1, L_2\}$ of the *Mask* dataset. $\phi()$ parameterization is HLLE. *Trimmings* (i) NURBs, (ii) RBFs and (iii) NURBS + RBFs. **a** NURBS interpolation/extrapolation. **b** RBFs surface. **c** NURBS + RBFs representation

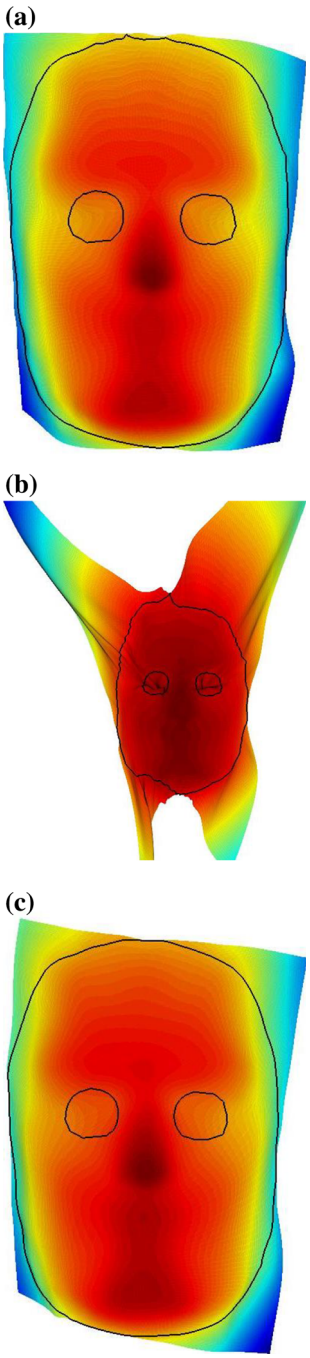

Fig. 7 Dataset *Mask*. Trimmed surfaces S using NURBs based on RBFs control polyhedron P. $\phi()$ parameterizations are: **a** Isomap trimmed surface. **b** Laplacian Eigenmaps trimmed surface. **c** HLLE trimmed surface

Dimensional reduction (Sect. 4.1) indicates that each point $u_i \in \mathbb{R}^2$ satisfies $\phi(x_i) = u_i$. For the particular case of radial basis functions, $f = S$ and S satisfies $S(u_i) = x_i$. The functions $\Phi_{i*}()$ are RBFs. The weights α_i are estimated by least squares and $R_{i*}(w)$ are stabilizing polynomials. Notice that an RBF works with an unordered, non-degenerated set of conditions $S(u_i) = x_i$ and that an RBF is well defined even outside of the bounds defined by the boundary $\partial(\phi(M))$ in the parameter space.

Within the RBF alternative, this investigation used $S(w)$ as defined by Eq. (10). Thin plate splines $\Phi(r) = r^2 \ln(r)$

Table 1 *Mask* dataset (Fig. 4a)

Algorithm stage	Figures	Comments
Isomap M Parameterization $\phi()$	Fig. 5a	Nearly isometric mapping
Laplacian Eigenmaps M parameterization $\phi()$	Fig. 5b	High geometrical distortion specially near the boundary
Hessian local linear embedding M parameterization $\phi()$	Fig. 5c	Non-isometric mapping with low shape distortion

Appraisal of dimensionality reduction methods (map $\phi : M \longrightarrow \mathbb{R}^2$)

were chosen as basis functions for computing the RBFs since they minimize a *bending energy* functional E in $\mathbb{R}^2$ [30]:

$$E(f) = \iint_{\mathbb{R}^2} \left(\frac{\partial^2 f}{\partial w_1^2} \right)^2 + 2 \left(\frac{\partial^2 f}{\partial w_1 \partial w_2} \right) + \left(\frac{\partial^2 f}{\partial w_2^2} \right)^2 dA \tag{11}$$

which results in a minimization of curvature on the reconstructed smooth surface.

4.2.3 NURBs+RBFs representation

In this strategy, a NURBs formulation is implemented for the surface S, but RBFs are used to construct the control polyhedron that underlies S. The process is: (1) use an RBF formulation [similar to Eq. (10)] to build a control polyhedron P. (2) Compute S by fitting a NURBS [Eq. (9)] over the control polyhedron P. This approach presents advantages over the previous (NURBs alone, RBFs alone) ones:

1. RBFs are a more natural manner to extrapolate (i.e., to evaluate the surface beyond the parametric domain used to create it), when compared to NURBs, splines or other surfaces who use a rectangular control polyhedron P.
2. RBFs require no particular order in the (u_i, x_i) pairs. NURBs require a spatial grid in the control polyhedron P.
3. RBF can be fed the information included in M. The resulting RBF is then able to calculate a control P already uses the whole information included in M. In contrast, iteratively growing a NURBs patch until it approximates the whole M uses, by definition, only partial information.
4. RBFs are very expensive in terms of data storage and computation, while NURBs require only the storage of the control polyhedron P. On the other hand, RBFs for not require a control polyhedron P (difficult to find as discussed earlier).
5. Computing a surface point in a NURBS is faster than in the RBFs formulation.
6. NURBS are a standard in CAD CAM CAE tools for representing trimmed surfaces. RBFs are less popular for downstream CAD algorithms (e.g., solid boolean operations).

4.3 Surface trimming

In order to trim the surface S, a set of LOOPs $\{L_0, L_1, \ldots, L_m\}$ must be laid down on the parametric surface such that the boundaries of the original mesh are represented by the LOOPs. The procedure for computing each L_i is:

1. Compute the boundary representation (B-REP) of M, which produces $\partial M = \{\Gamma_0, \Gamma_1, \ldots, \}$ with each piecewise linear Γ_i being a closed sequence of triangle EDGEs $\overline{x_i x_j}$.
2. Map each LOOP $\Gamma_i \in M$ to $\phi(\Gamma_i) = \gamma \in R^2$. These piecewise linear LOOPs γ_i form $\partial\phi(M) = \{\gamma_0, \gamma_1, \ldots, \}$.
3. Compute λ_i as a tight re-sample of LOOP γ_i.
4. Map each λ_i back to S (near M) by computing $L_i = S(\lambda_i)$, with S as in Sect. 4.2. The trimmed surface $FACE = (S, \{L_0, Ł_1, \ldots, \}) \approx M$ is complete.

5 Results and discussion

Section 5.1 presents an appraisal of the combination of the available and developed methods for trimmed surface synthesis using one (*Mask*) dataset. Then, Sect. 5.2 shows results with additional datasets. In these examples, particular combinations of methods were applied (i.e., no appraisal of possible combinations was conducted).

5.1 *Mask* dataset results

Table 1 presents, for the *Mask* dataset the results of dimensionality reduction (manifold learning or $\phi()$ mapping). For this well connected M dataset, nearly isometric or nearly conformal mappings give low distortion. In contrast, Laplacian Eigenmaps produce large distortions near the border ∂M, as it only seeks to preserve the M graph neighborhoods.

Table 2 summarizes the results of surface trimming applied on the set $\phi_{HLLE}(M) \in \mathbb{R}^2$ ($\phi_{HLLE}(M)$: result of applying HLLE-type $\phi()$ on mesh M). NURBs extrapola-

Table 2 *Mask* dataset (Fig. 4a)

Algorithm stage	Figures	Comments
NURBs extrapolation	Fig. 6a	Several folds arise in extrapolated zones inducing high curvatures on such zones
RBFs interpolation	Fig. 6b	Smoothness along the complete surface is achieved (even in the extrapolated areas)
RBFs Control polyhedron $P+$ NURBs parametric S	Fig. 6c	The surface is a bit smoother than the RBFs one (this can be evidenced by comparing the boundaries) but points of the original triangulation may not lay on the trimmed surface

Appraisal of surface trimming methods applied on $\phi_{HLLE}(M) \in \mathbb{R}^2$. The trimmed surface is $F = (S, \{L_0, L_1, L_2\})$

Table 3 *Mask* dataset (Fig. 4a)

Dimensionality reduction method	Figures	Comments
Isomap M parameterization $\phi()$	Fig. 7a	Low curvatures and low distortions
Laplacian Eigenmaps M parameterization $\phi()$	Fig. 7b	High distortions and self-intersections of the surface in the eye area. Several folds arise high curvatures along the surface. Irregular boundary
Hessian local linear embedding M parameterization $\phi()$	Fig. 7c	Low curvatures and low distortions

Impact of dimensionality reduction algorithms on the resulting trimmed surface F. Constant surface trimming algorithm (hybrid RBF + NURBs). variate dimensionality reduction algorithms (Isomap, Laplacian Eigenmap, HLLE)

Table 4 *Beetle* dataset

Parameterization algorithm	Backmapping algorithm	Figures	Comments
HLLE	NURBS extrapolation	Fig. 8a	High curvatures and high distortions at extrapolated areas. Misrepresentation of the boundary at several areas (car windows)
Isomap	NURBs + RBFs	Fig. 8b	Smooth surface, low curvatures and low distortions.
Laplacian Eigen-maps	NURBs + RBFs	Fig. 8c	High curvatures and high distortions. Irregular boundary
HLLE	NURBs + RBFs	Fig. 8d	Smooth surface, low curvatures and low distortions

Appraisal of results under different combinations of available methods for parameterization and surface trimming

Table 5 S-shape dataset (Fig. 9a)

Algorithm stage	Figures	Comments
Isomap parameterization $\phi()$	Fig. 9b	The boundary and several triangles intersect in the parameter space (ϕ is no longer a bijection). The triangulation does not preserve a consistent orientation
HLLE parameterization $\phi()$	Fig. 9c	Adequate representation of the non-convex parametric domain ϕ of the S dataset
Isomap-based trimmed surface F	Fig. 9d	Self intersection of the trimmed surface F. Additionally, F does not resemble the original dataset
HLLE-based trimmed surface F	Fig. 9e	Smooth surface with low distortions. Extrapolated areas follow consistent geodesic paths despite the non-convexity of the parametric space

Appraisal of the algorithm applied to a dataset which is highly non-convex in the parametric space

Table 6 Additional data sets: *Partial-Venus*, *Teddy*, *Half-Glove*

Dataset	Used parameterization and Backmapping algorithm	Figures	Comments
Partial-Venus	(1) Parameterization with Isomap. (2) Surface trimming with hybrid NURBs + RBFs	Fig. 10a	Smooth surface with low distortion
Teddy	(1) Parameterization with Isomap (2) Surface trimming with hybrid NURBs + RBFs	Fig. 10b	Smooth surface with low distortion. Small holes do not affect drastically the Isomap-based parameterization
Half-Glove	(1) Parameterization with HLLE (2) Surface trimming with hybrid NURBs + RBFs	Figs. 10c–e	Surface with low distortions. High curvatures and self-intersections arise at extrapolated areas

Back mapping algorithm is hybrid (RBFs for control polyhedron P and NURBS for parametric surface $S()$)

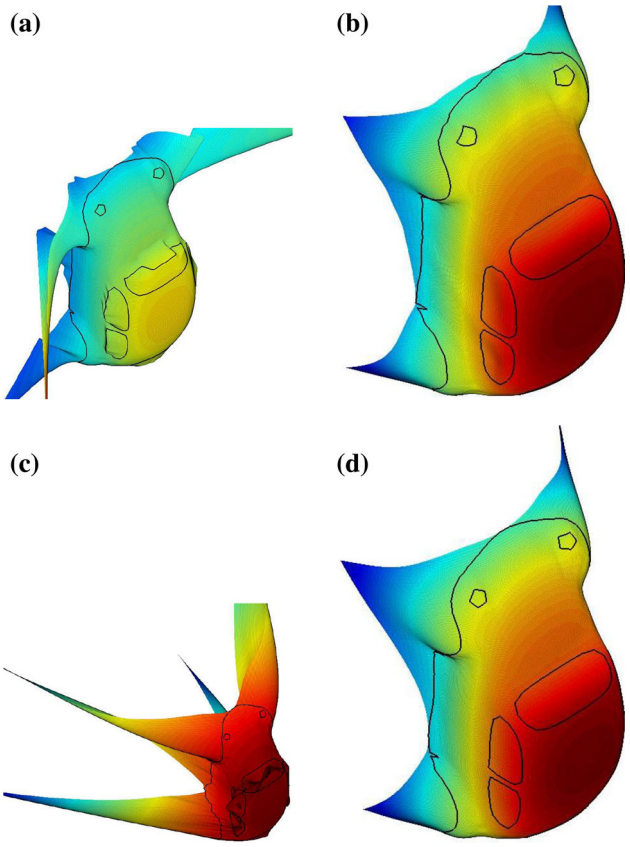

Fig. 8 Results for the *Beetle* dataset under several combinations of the developed methods. **a** HLLE and NURBS extrapolation. **b** Isomap and NURBs + RBFs. **c** Laplacian Eigenmaps and NURBs + RBFs. **d** HLLE and NURBs + RBFs

tion is definitively inadequate for synthesizing the control polyhedron P, producing folds in the surface S. RBFs produce a smooth surface but they are extremely expensive to set up and to use. In addition, they are not popular as a B-Rep standard in commercial CAD. The third option (using RBFs to calculate solely the control polyhedron P of S and NURBs for calculate S itself), produces a NURBs surface as good as RFBs, in an economic manner, with the advantage of being fully applicable in commercial CAD software.

Table 3 cites the final results (trimmed surface $F = (S, \{L_0, L_1 \ldots\})$) with the hybrid back-mapping (RBFs to compute the control polyhedron P plus NURBS to compute the parametric Surface S). This hybrid mapping is fed with three inputs: $\phi_{Isom}(M)$, $\phi_{Lapl}(M)$ and $\phi_{HLLE}(M)$. In this manner, we compare the quality of these three inputs by keeping S constant. As expected from previous tests, Isomap and HLLE perform well, while Laplacian Eigenmaps produces a distorted final trimmed surface.

5.2 Results for other datasets

Table 4 considers the *Beetle* dataset. The test 1 confirms the poor performance of a NURBs extrapolation for Surface Trimming (back mapping). The tests 2, 3 and 4 use the hybrid back mapping (RBFs for control polyhedron P+ NURBs for S) and variate the $\phi()$ mapping [$\phi_{Isom}(M)$, $\phi_{Lapl}(M)$, $\phi_{HLLE}(M)$]. As discussed before, Isomap and HLLE perform better than Laplacian Eigenmaps.

Table 5 is devoted to the S-shape dataset. This dataset (Fig. 9a) violates the fundamental assumption of Isomap, that geodesic curves on the mesh $M = (X, T)$ can be approximated by shortest paths in the graph T. Test 1 confirms the failure of ϕ_{Isom}. As expected, feeding $\phi_{Isom}(M)$ into the back mapping F produces another failure (test 3) In contrast, HLLE produces a good quality parameterization ϕ_{HLLE} (test 2). Feeding ϕ_{HLLE} into the back-mapping F produces the good results in test 4.

Table 6 relates to additional datasets (*Partial-Venus*, *Teddy*, *Half-Glove*). The concavities and/or holes in the *Partial-Venus* and *Teddy* datasets are small and do not distort the graph-based estimation of the geodesics on M. Therefore, the Isomap $\phi_{Isom}()$ parameterization is still stable. In the *Half-Glove* case, Isomap is clearly inadequate and therefore a HLLE $\phi_{HLLE}()$ parameterization is used instead, with good results. However, the hybrid NURBS $S : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$

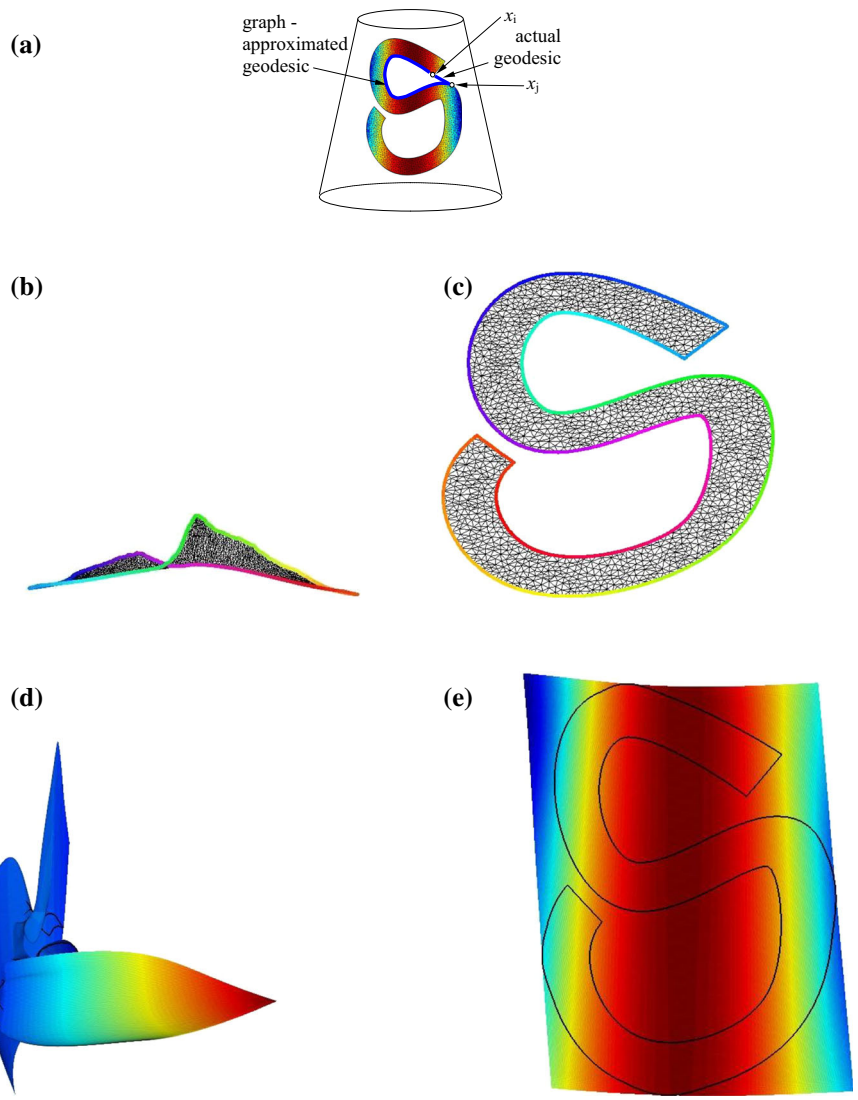

Fig. 9 Results of the algorithm for the S dataset under the RBF + NURBS approach using Isomap (*left*) and HLLE (*right*). **a** S dataset. **b** Failed Isomap parameterization ϕ of the S dataset. **c** Successful HLLE parameterization ϕ of the S dataset. **d** Failed back-mapping $\mathbb{R}^2 \rightarrow \mathbb{R}^3$ of the Isomap parameterization. **e** Successful back-mapping $\mathbb{R}^2 \rightarrow \mathbb{R}^3$ of the HLLE parameterization

with RBFs—computed control polyhedron P, results to be self-intersecting in the extrapolated neighborhoods.

6 Conclusions

This manuscript presents a method for approximating a triangle-based mesh $M \in \mathbb{R}^3$ with a trimmed surface $F = (S, \{L_0, L_1, \ldots\})$ in which $S : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$ is a parametric surface and $\{L_0, L_1, \ldots\} = \partial F$ is a set of LOOPs which constitute the boundary of F (approximating ∂M, the boundary of M).

The determination of F includes as main steps: (1) to devise a *forward* dimension reduction $\phi : M \longrightarrow \mathbb{R}^2$ which finds a parametric space U for M ($U = \phi(M)$). (2) To synthesize a *backward* parametric surface $S : \mathbb{R}^2 \longrightarrow \mathbb{R}^3$, to compute $S(\phi(M)) = F \approx M$. The previous procedure implies to map $\partial M \in \mathbb{R}^3$, the boundary of M to parametric space $\mathbb{R}^2$ and then back to $\mathbb{R}^3$ via the parametric surface $S()$.

Three manifold learning algorithms were tested: (1) Isomap, (2) Laplacian Eigenmaps and (3) HLLE. Isomap and HLLE presented in general good results for *quasi-developable* meshes. Laplacian Eigenmaps presented high distortions. In addition, we devise a data set that causes Isomap to fail, when the triangulation graph T fails to estimate the geodesic curves on M. This failure occurs when M presents significant holes or concavities. In conclusion, HLLE was found to be most reliable algorithm for the $\phi()$ mapping.

The back-mapping $S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$ was computed by following (1) a NURBS interpolation/extrapolation approach, (2) an RBFs surface approach and (3) a hybrid approach which includes the calculation of the control polyhedron P with RBFs and the parametric surface S with a NURBS for-

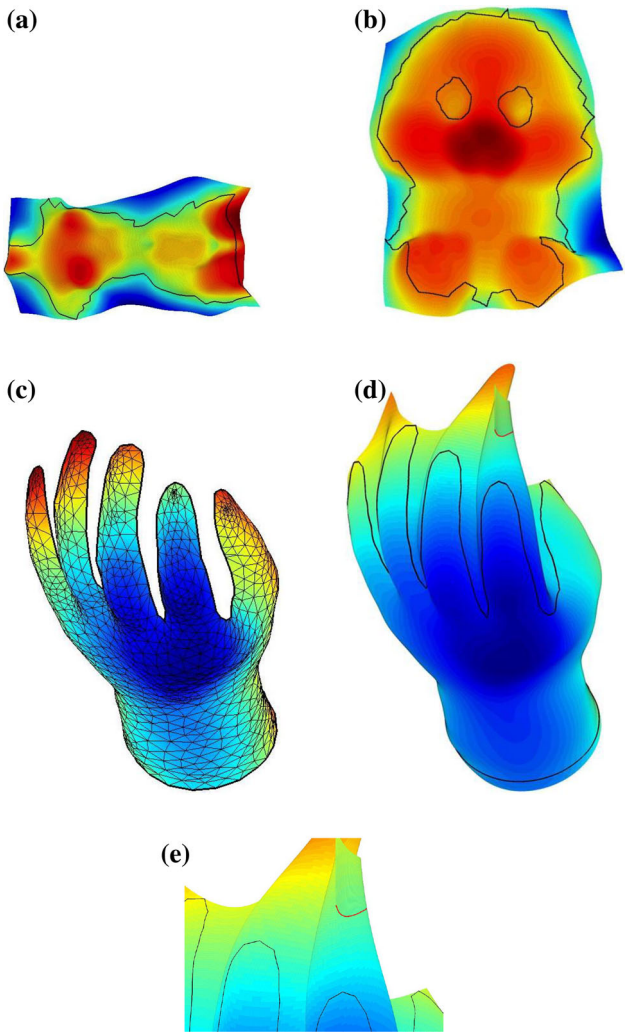

Fig. 10 Trimmed surfaces for several datasets using (i) IsoMap parameterization (*up*) and (ii) HLLE parameterization (*down*). RBF + NURBS were used for the back-mapping $S : \mathbb{R}^2 \rightarrow \mathbb{R}^3$. **a** *Partial-Venus*. Hybrid RBFs + NURBS $S()$ from $\phi_{Isom}(M)$. **b** *Teddy*. Hybrid RBFs + NURBS $S()$ from $\phi_{Isom}(M)$. **c** *Half-Glove* dataset. **d** *Half-Glove*. Hybrid RBFs + NURBS $S()$ from $\phi_{HLLE}(M)$. **e** *Half-Glove*. Self-intersection of Hybrid RBFs + NURBS $S()$ trimmed surface F

mulation. This hybrid approach was found to be the most robust technique for back mapping.

The chosen combination of mesh parameterization ($\phi_{HLLE}(\ )$) and parametric surface (NURBS + RBF $S(\ )$) works correctly for the *Teddy*, *Beetle*, *Partial-Venus*, S-strip and *Mask* data sets. The *Half-Glove* data set shows to challenge our method in that the S surface results being self-intersecting. However, the trimmed extent of S that actually constitutes the FACE $F = (S, \{L_0, L_1, \ldots\})$ is not self-intersecting and therefore is is still a 2-manifold. Nevertheless, our implemented algorithms do not guarantee 2-manifoldness of the trimmed surface F in all cases.

6.1 Ongoing work

The problem of mesh M approximation by a trimmed surface is easier to solve if M is developable or near-developable. An obvious future research direction is the segmentation of M into near-developable sub-meshes M_i.

Mesh parameterization with trimmed surfaces requires an underlying parametric surface $S()$ which approximates some M_S super-set of M ($M \subset M_S$). Such a synthesis is highly under-constrained and requires further work. An opportunity is present by the fact that, even if a S surface is self intersecting far away of the region of interest F, it might not self—intersect in the region F, which makes S still usable.

Datasets

The datasets were downloaded from the public sites: http://www.cse.buffalo.edu/~jryde/cse673/04.html, http://gpeyre.github.io/numerical-tours/matlab/fastmarching_4bis_geodesic_mesh/ and http://liris.cnrs.fr/meshbenchmark/.

References

1. Zheng, J., Chan, K., Gibson, I.: Constrained deformation of freeform surfaces using surface features for interactive design. Int. J. Adv. Manuf. Technol. (2003). doi:10.1007/s00170-002-1442-8
2. Yoshizawa, S., Belyaev, A., Seidel, H.P.: A fast and simple stretch-minimizing mesh parameterization. In: Proc. Shape Model. Appl. (2004). doi:10.1109/SMI.2004.1314507
3. Specht, M., Lebrun, R., Zollikofer, C.P.: Visualizing shape transformation between chimpanzee and human braincases. Vis. Comput. (2007). doi:10.1007/s00371-007-0156-1
4. Krishnamurthy, A., Khardekar, R., McMains, S.: Optimized GPU evaluation of arbitrary degree NURBS curves and surfaces. Comput.-Aided Des. (2009). doi:10.1016/j.cad.2009.06.015
5. Pietroni, N., Massimiliano, C., Cignoni, P., Scopigno, R.: An interactive local flattening operator to support digital investigations on artwork surfaces. IEEE Trans. Vis. Comput. Graph. (2011). doi:10.1109/TVCG.2011.165
6. Liu, X.M., Wang, S.M., Hao, A.M., Liu, H.: Realistic rendering of organ for surgery simulator. Comput. Math. Appl. (2012). doi:10.1016/j.camwa.2011.11.030
7. Tierny, J., Daniels II, J., Nonato, L.G., Pascucci, V., Silva, C.T.: Interactive quadrangulation with reeb atlases and connectivity textures. IEEE Trans. Vis. Comput. Graph. (2012). doi:10.1109/TVCG.2011.270
8. Zhu, X.F., Hu, P., Ma, Z.D., Zhang, X., Li, W., Bao, J., Liu, M.: A new surface parameterization method based on one-step inverse forming for isogeometric analysis-suited geometry. Int. J. Adv. Manuf. Technol. (2013). doi:10.1007/s00170-012-4251-8
9. Zuo, B.Q., Huang, Z.D., Wang, Y.W., Wu, Z.J.: Isogeometric analysis for CSG models. Comput. Methods Appl. Mech. Eng. (2015). doi:10.1016/j.cma.2014.10.046
10. Li, G., Ren, C., Zhang, J., Ma, W.: Approximation of Loop Subdivision Surfaces for Fast Rendering. IEEE Trans. Vis. Comput. Graph. (2011). doi:10.1109/TVCG.2010.83
11. Yang, L., He, D., Zhang, Z.: Construct G1 Smooth surface by using triangular gregory patches. In: Fifth Int. Conf. Image Graph—ICIG '09. (2009). doi:10.1109/ICIG.2009.55

12. Dyken, C., Reimers, M., Seland, J.: Real-time GPU silhouette refinement using adaptively blended Bézier patches. Comput. Graph. Forum **27**(1), 1–12 (2008)
13. Zhang, Z., Wang, Z., He, D.: A new bi-cubic triangular gregory patch. In: Int. Conf. Comput. Sci. Softw. Eng. (2008). doi:10.1109/CSSE.2008.298
14. Boubekeur, T., Reuter, P., Schlick, C.: Scalar tagged PN triangles. In: Eurographics Short Papers, Eurographics Association and Blackwell, Dublin, Ireland (2005)
15. Mao, Z., Ma, L., Tan, W.: A modified nielsons side-vertex triangular mesh interpolation scheme. In: Gervasi, O., Gavrilova, M., Kumar, V., Laganà, A., Lee, H., Mun, Y., Taniar, D., Tan, C. (eds.) Computational science and its applications ICCSA 2005, pp. 776–785. Springer, Berlin (2005)
16. Vlachos, A., Peters, J., Boyd, C., Mitchell, J.L.: Curved PN triangles. In: Proc. 2001 Symp. Interact. 3D Graph (2001). doi:10.1145/364338.364387
17. Acosta, D.A., Ruiz, O.E., Arroyave, S., Ebratt, R., Cadavid, C., Londono, J.J.: Geodesic-based manifold learning for parameterization of triangular meshes. Int. J. Interact. Des. Manuf. (IJIDeM) (2014). doi:10.1007/s12008-014-0249-9
18. Yu, H., Lee, T.Y., Yeh, I.C., Yang, X., Li, W., Zhang, J.J.: An RBF-based reparameterization method for constrained texture mapping. IEEE Trans. Vis. Comput. Graph. (2012). doi:10.1109/TVCG.2011.117
19. Guo, Y., Wang, J., Sun, H., Cui, X., Peng, Q.: A novel constrained texture mapping method based on harmonic map. Comput. Graph. (2005). doi:10.1016/j.cag.2005.09.013
20. Belkin, M., Niyogi, P.: Laplacian Eigenmaps and spectral techniques for embedding and clustering. In: Neural Inf. Process. Syst.: Nat. and Synth.—NIPS, MIT Press, Vancouver, Canada (2001)
21. Sheffer, A., de Sturler, E.: Parameterization of faceted surfaces for meshing using angle-based flattening. Eng. Comput. (2001). doi:10.1007/PL00013391
22. Zhao, X., Su, Z., Gu, X.D., Kaufman, A., Sun, J., Gao, J., Luo, F.: Area-preservation mapping using optimal mass transport. IEEE Trans. Vis. Comput. Graph. (2013). doi:10.1109/TVCG.2013.135
23. Zou, G., Hu, J., Gu, X., Hua, J.: Authalic parameterization of general surfaces using lie advection. IEEE Trans. Vis. Comput. Graph. (2011). doi:10.1109/TVCG.2011.171
24. Pietroni, N., Tarini, M., Cignoni, P.: Almost isometric mesh parameterization through abstract domains. IEEE Trans. Vis. Comput. Graph. (2010). doi:10.1109/TVCG.2009.96
25. Liu, L., Zhang, L., Xu, Y., Gotsman, C., Gortler, S.J.: A local/global approach to mesh parameterization. Comput. Graph. Forum (2008). doi:10.1111/j.1467-8659.2008.01290.x
26. Sun, X., Hancock, E.R.: Quasi-isometric parameterization for texture mapping. Pattern Recogn. (2008). doi:10.1016/j.patcog.2007.10.027
27. Desbrun, M., Meyer, M., Alliez, P.: Intrinsic parameterizations of surface meshes. Comput. Graph. Forum (2002). doi:10.1111/1467-8659.00580
28. Tenenbaum, J.B., de Silva, V., Langford, J.C.: A global geometric framework for nonlinear dimensionality reduction. Sci. (NY) (2000). doi:10.1126/science.290.5500.2319
29. Donoho, D.L., Grimes, C.: Hessian eigenmaps: locally linear embedding techniques for high-dimensional data. Proc. Natl. Acad. Sci. (2003). doi:10.1073/pnas.1031596100
30. Bookstein, F.L.: Principal warps: thin-plate splines and the decomposition of deformations. IEEE Trans. Pattern Anal. Mach. Intell. (1989). doi:10.1109/34.24792