Graphics and Visual Computing 6 (2022) 200043

Contents lists available at ScienceDirect

Graphics and Visual Computing

journal homepage: www.elsevier.com/locate/gvc

Technical section

Multiresolution surface blending for detail reconstruction ☆

Hono Salval *, Andy Keane, David Toal

Computational Engineering and Design group, University of Southampton, Rolls Royce University Technology Centre for Computational Engineering, Boldrewood Innovation Campus, Burgess Rd, Southampton, SO16 7QF, Hampshire, United Kingdom

ARTICLE INFO

Article history:
Received 6 August 2021
Received in revised form 7 January 2022
Accepted 22 January 2022
Available online 31 January 2022

Keywords:
Surface reconstruction
Assembly based modelling
Shape blending
Reverse engineering

ABSTRACT

While performing mechanical reverse engineering, 3D reconstruction processes often encounter difficulties capturing small, highly localized surface information. This can be the case if a physical part is 3D scanned for life-cycle management or robust design purposes, with interest in corroded areas or scratched coatings. The limitation partly is due to insufficient automated frameworks for handling – localized – surface information during the reverse engineering pipeline. We have developed a tool for blending surface patches with arbitrary irregularities, into a base body that can resemble a CAD design. The resulting routine preserves the shape of the transferred features and relies on the user only to set some positional references and parameter adjustments for partitioning the surface features.

1. Introduction

New computer technologies for processing and capturing geometric data have emerged in the previous decade. Several methods in engineering deal with digital models, which are altered in real-time to mirror properties of real-world objects, one of these methods being the *digital twin* [1–3], used to create a real-time updated digital representation of a device in service. Aside from technical applications, visuals for animation and video games have shifted their focus to incorporating captured geometry from real-world objects rather than creating it artificially. With various applications for simulation and manufacturing, we believe that incorporating outsourced data into engineering design frameworks is more approachable now than it was previously. Simulating the effects of surface flaws, for example, is not a simple operation; it necessitates either sculpting the design's surface or transferring irregularities from an external source while maintaining key shape attributes. For example, Dawes et al. created a digital twin of a turbine blade to reflect surface flaws experienced during service [4]. This method deforms a surface mesh using mathematical functions and real-time sensor data until it resembles the fault patterns of the actual blade. While this method can be used to analyse a generic case of surface abrasion, its routines must be built from the ground up to imitate each particular defect pattern or region of interest.

In an attempt to address that issue and generalize surface feature transference, this article presents a blending approach to transfer surface data between meshes, deforming the details to match the underlying curvature while keeping the relevant bits (see Fig. 1).

While building models from diverse combined shapes is typical in the videogame business, the engineering design process is generally confined to a distinct framework with its own methodologies and geometric formats. The necessity of employing efficient processes in large projects, such as when design, simulation, and product management occur collaboratively across a complete team, reinforces this tendency. The rigidity of NURBS geometry (Non-Uniform Rational Bézier Spline), also known as solid geometry and often used in engineering design frameworks, is another incentive to stay in a single framework. Meshes, on the other hand, are easier to modify and manipulate locally, whereas solids typically require a reconfiguration step to accommodate additional entities, and any local modification necessitates the addition of new NURBS patches.

The combined use of solid and mesh format is often found when reconstructing 3D scans, exporting geometry for simulation or preparing the model for 3D printing, but they rarely coexist during the design process. Nevertheless, some popular packages offer tools for handling meshes along with solids. Take Siemens PLM as an example; it provides a geometric object called "convergent geometry" that handles the so-called free-form formats for tasks like sculpting, filling holes, or shape deformation. These tools are fundamentally manual and rely on the user to blend and stitch shapes together, leaving an isolated framework that does not fully integrate into the engineering design workflow. This deficit is commented on by Bodgan et al. [5] when performing the reconstruction of a Francis turbine scan, during which localized

☆ This document contains results of a Ph.D. program partially funded by Rolls-Royce PLM.

* Corresponding author.
E-mail addresses: hsv1e19@soton.ac.uk (H. Salval), Andy.Keane@soton.ac.uk (A. Keane).

https://doi.org/10.1016/j.gvc.2022.200043

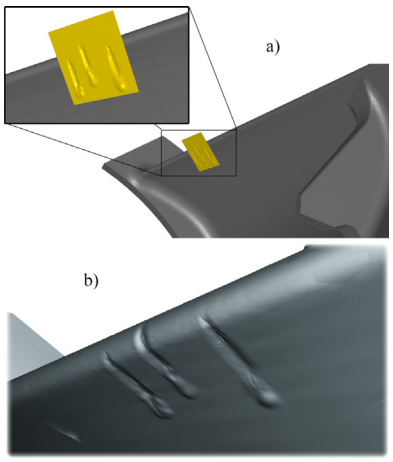

Fig. 1. Top: a surface patch placed near a blade's trailing edge. The patch contains typical scratches found in turbine coatings. Bottom: blended scratches added into the blade's surface.

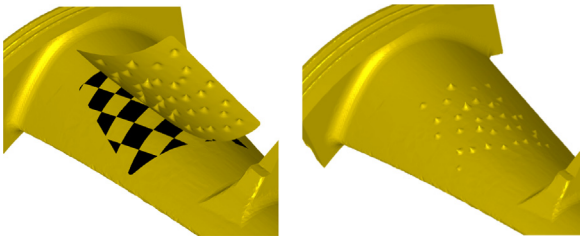

Fig. 2. A surface patch with bumps is blended into another shape, which overall curvature does not match the patch. (a) The fitting region (black/yellow square UV map) is projected onto the base. (b) Surface features are then reconstructed on the fitted thin-plate.

defect information is lost because of the lack of a framework for managing it.

Our algorithm handles such localized irregularities by extracting surface regions and transferring them to another model's surface. Referring to Fig. 1, the routine should be capable of deforming the extracted region to match the model's curvature at the blending area, and it should avoid distortions on the relevant detail. As scratches in the figure resemble a particular material displacement defect, any alteration of the relative proportions might affect later simulation results. A tool that provides good and automated detail transfers also makes it possible to build a repository of geometric features (e.g. scanned defects on physical objects, textures and other resources for simulation). The routine is achieved by separating the patch into a feature layer and a thin-plate layer, which supports the features. Then the thin-plate is mapped onto the design, being folded to wrap on the surface. To finish the transfer, features are reconstructed with new positioning and orientation onto the fitted thin-plate (see Fig. 2).

2. Related work

The task of blending shapes has been discussed many times: building models from a precomputed database of parts [6] is an early approach that ensures the outcome by restricting the choices to a given part set. More flexibility has been gained since then, by connecting arbitrary boundaries with biharmonic surfaces [7], to optimizing and deforming the boundaries to blend [8]. A recent approach is to provide hierarchical reasoning that helps decompose geometry into a certain set of features, and later reassembling new shapes upon those features. Sometimes this hierarchy is hard-coded into the routine using predefined relationships [9–12], but with the increasing popularity of machine learning techniques much attention has shifted to self-learning algorithms, trained with hundreds of examples, including assembled and disassembled shapes [13–18].

The utility of these tools on product design affects various processes: they can suggest design aspects to the user [19–22], facilitate the recovery of shapes from images [23–27] and scan data [26,28,29]. Even further, well developed assembly algorithms enhance design search for purposes such as *generative* and *robust modelling*, where certain constraints are imposed and an algorithm tries different designs in a search for the one matching the imposed conditions [14,30–33]. Algorithms for performing design exploration are of the most interest for us, along with other practices such as surface inspection. As previously commented, our routine aims to transfer surface features between non-shape-matching surfaces; it merges both geometries while preserving the relative proportions of each individual feature. Straight applications of such a tool are: shape reconstruction for surface defect inspection (such as coating scratches, abrasion) [34,35], simulation of surface irregularities in robust design exploration [36], and real time diagnosis of devices through sensor data processing [4].

Building the proposed tool require a high-level perspective of industrial design requirements and the different subprocesses involved in an automated blending algorithm. Most CAD software shares several core aspects; the design philosophy is to build models by introducing new elements and constraints to the geometry. Also, the usual way to encode shapes is by assembling solid primitives (as planes, spheres, cages) and surface patches defined by scalar functions. The typical CAD design is composed of well defined geometric parameters such as lengths or angles, and even more abstract characteristics can be introduced such as rotational symmetries. This practice enhances compatibility with manufacturing processes and simplifies the design process for the user through a programmatic framework.[1] When bringing arbitrary mesh geometry to an engineering design framework, it is often processed to isolate and classify the different parts of its shape through segmentation, so the same engineering, programmatic-like framework can be applied for the imported mesh.

Segmentation of surfaces. Performing a segmentation in a surface mesh means isolating and – often – classifying regions from a model. It is often utilized in processes such as topological optimization, surface reconstruction and computational object recognition. A common situation is wanting a triangular mesh to be reconstructed as a NURBS surface, and there is plenty variety in publications regarding this topic [37–40]. In particular, Laplacian and Bi-Laplacian based segmentation are pretty flexible in terms of recognizing features of any nature. They are applicable on acute mechanical shapes or organic bodies, yielding in both cases an intuitive separation of the different parts. Publications such as Mejia's et al. combine seeding and Laplacian techniques to accelerate the feature isolation [41,42], referring to this approach as spectral-based mesh segmentation. In the standard terminology of digital geometry, spectral mesh processing refers to the manipulation and analysis of shapes by exploiting the properties of operators like the graph Laplacian applied on a surface mesh. Laplacian methods capture geometric features in different scales, often also described as decoupling the curvature in different frequencies of varying magnitude.

There are still limitations, particularly for recognizing high abstraction features as components of an assembly; these can

[1] A programmatic framework allows to introduce new aspects of the shape procedurally.

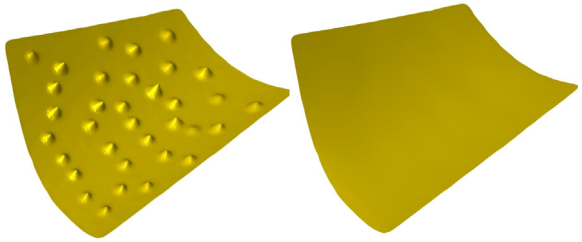

Fig. 3. (a) Surface patch with bump irregularities. (b) Thin-plate version of the patch.

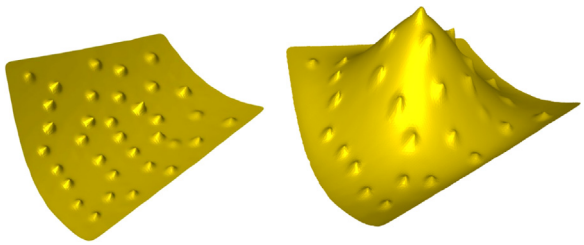

Fig. 4. Biharmonic deformation of a patch, where a central vertex has been displaced upwards. The result is an interpolating surface that smoothly extends the deformation along the surface.

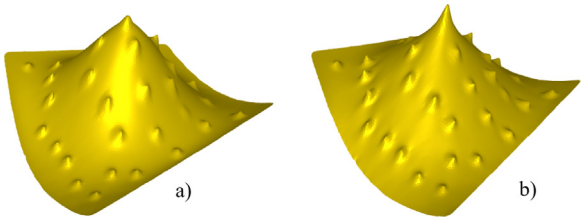

Fig. 5. (a) Deformed patch through biharmonic deformation. (b) Deformed patch through rigid deformation. The bumps in the biharmonic approach get tilted towards their original orientation, while for the rigid deformation, they correctly reorient to match the new disposition.

be addressed by novel routines based on hardcoded hierarchical reasoning [12] and deep learning optimizers [43,44]. Another capability lacking in segmentation algorithms to date is handling localized detail surface information. As scanning technology reaches greater accuracy, there is a growing interest in more localized surface details. These can be introduced into simulation pipelines for application life-cycle management (ALM) [4], or as part of a reverse engineering process. While scanning a heavily eroded Francis Turbine, it is noted by Sedai et al. [45] that there is no tool for handling erosions and defects during the reverse engineering process, which discards those in favour of larger scale features. Apart from disregarding small detail during the segmentation process, encoding this kind of shape is another obstacle, as complex irregularities require a large number of NURBS patches to be defined. For that reason, another technique is often applied to handle mesh deformations, usually referred to as free-form deformation (FFD) by industrial side publications, and more commonly named *skinning* in audiovisual, creative and machine learning circles.

Skinning and deformation. Probably one of the best definitions for presenting this concept is provided by Jacobson, Deng, et al. *"Skinning is the process of controlling deformations of a given object using a set of deformation primitives"* [46]. Skinning can be applied in many different forms, regarding the chosen handlers: e.g. skeleton, bones [46], point, lines and cages [47], or point lattices [44,48]. And also regarding how the geometry deforms when moving the handlers: e.g. biharmonic deformation fields [47,49] or spline interpolation [44,48]. In particular, for handling surface patches, it is common practice to generate a thin-plate version of the surface where any irregularity is removed and use it to guide detail to new positions and orientations (see Fig. 3). Generating the thin-plate can be done by minimizing acute curvature along the surface, leading to a softening effect. As an example, the Bi-Laplacian operator Δ^2 can be discretized for a triangular mesh and serves to smooth away high-frequency features of a given shape.

Solving the following equation, with fixed boundaries $\mathbf{v}_{i\in bc} = \mathbf{v}_{bc}$, leads to a smoothed version of the patch:

$$\Delta^2\mathbf{v}_i = 0 \qquad \text{subject to} \quad \mathbf{v}_b = \mathbf{v}_{bc}. \tag{1}$$

The resulting surface would have the minimal curvature imposed by the boundary conditions, leaving a biharmonic surface. After performing any transformation on the thin-plate, details need to be reconstructed with the new shape. For this process, there are various propositions, one of the simplest ones being based on Laplacian processing. It can be used to soften irregularities on surface positions. It can also be applied on deformation fields to remove abrupt deformations out of a given deformation field [50]. To perform what is called a biharmonic deformation, we solve Eq. (1), this time applied on a deformation field $\{\mathbf{d}_i\}$ over the mesh vertices $\{\mathbf{v}_i\}$:

$$\Delta^2\mathbf{d}_i = 0 \quad \text{subject to} \quad \mathbf{d}_b = \mathbf{d}_{bc}. \tag{2}$$

The displacements $\mathbf{d}_{bc}$ are imposed at selected indices $i \in b$, the solution $\{\mathbf{d}_i\}$ is a soft interpolation of the deformation field passing through the imposed values (see Fig. 4).

Biharmonic deformation has limitations, though, being a linear method, it fails to calculate local rotations after deformation correctly, and the reconstructed features appear tilted towards their original orientation. This behaviour is often referred to as a non-rotation-invariant deformation. To avoid this problem and accomplish shape preservation, non-linear techniques come into play, such as rigid deformation algorithms. As-rigid-as-possible (ARAP) deformation minimizes the stretching and bending energy of the mesh, introduced by the imposed transformation. Roughly describing the mathematical details, rigid deformation methods aim to find, for given boundary conditions, the new positions $\{\mathbf{v}'_i\}$ and the local rotations $\{\mathbf{R}_i\}$ satisfying the following difference:

$$(\mathbf{v}'_i - \mathbf{v}'_j) = \mathbf{R_i}(\mathbf{v}_i - \mathbf{v}_j), \tag{3}$$

where $\mathbf{v}_i$ are the original positions of the mesh vertices, $\mathbf{p}'$ the deformed positions and $\mathbf{R_i}$ the local rotation matrix for each vertex. A variational approach to solve this equation was proposed to decouple the search for $\mathbf{R_i}$ and $\mathbf{v}'_i$ [51]. This approach provides more physically intuitive deformations (see Fig. 5), but often at the cost of stability and processing power. Not directly involved in our work but worth of mention is another method offering rotation-invariant results, which aims to force the deformation gradient of a thin-plate to be inherited by the original geometry [52].

This collection of techniques (segmentation, skinning, deformation) have been combined together for various surface blending solutions [20,53], highlighting the ones that include Laplacian techniques on their approach [8,49,54–56].

3. Multiresolution surface blending

To understand how the term *Multiresolution* fits into the description of this routine, consider the surface patch as a combination of two geometric entities: one is a soft surface that depicts the overall curvature, and the other defines surface irregularities that are distributed *along* the soft surface. This results

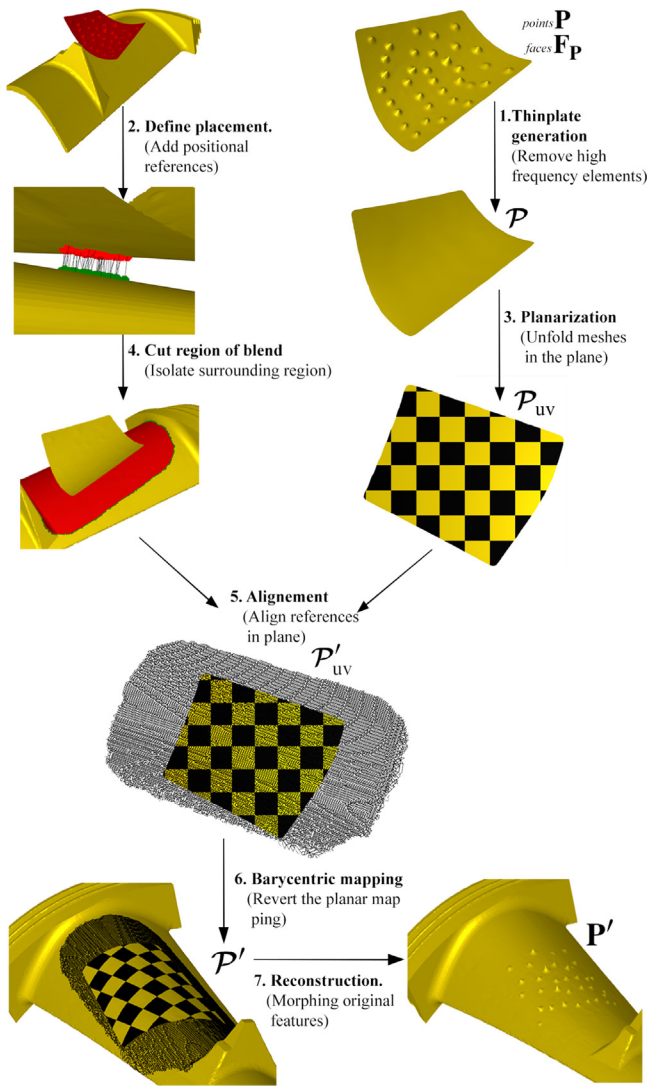

Fig. 6. Flow diagram of the blending routine. *1. Thin-plate generation:* High frequency features are removed from the patch **P** to yield a smooth supporting thin-plate $\mathcal{P}$. *2. Define placement:* Placement references are provided by the user or inferred by some heuristics. *3. Planarization:* The thin-plate is planarized by unfolding while keeping proportions and distances. *4. Cut region of blend:* From the base model, a region where the blending should happen is isolated and cut leaving an edge manifold surface; which is also mapped to the plane in a shape preserving manner. *5. Alignment:* Both planar meshes are aligned, according to the positional references. *6. Barycentric mapping:* The planar thin-plate vertices are mapped to positions in the planar base surface. Such map serves to obtain the thin-plate fitted on the original base model. *7. Reconstruction:* Details are reconstructed on their updated disposition, over the thin-plate wrapped over the model.

in a two-level description of the shape, which is known as a multiresolution scheme. This view is useful since each of these levels must be addressed individually when fitting the patch onto the base surface. The supporting level should inherit the base's curvature while keeping its surface area and shape, while the detail level should calculate the supporting level's inherited deformation. The schematic of Fig. 6 presents an exploded view of the steps.

This section outlines the proposed method for transferring imperfections from a triangle edge manifold mesh (the patch) to another surface (the base). It will be illustrated using a toy model of a turbine blade as the base model, as well as a transfer patch with bumps. Consider a triangular mesh patch of vertex positions $\mathbf{p}_i$ stacked on the matrix **P**; the mesh connectivity is described by a face matrix $\mathbf{F_P}$, which remains unaltered through the process and therefore ignored while describing the transformations applied to the shape. To guide the placement, some vertices of the patch are mapped to positions in the model's surface receiving the features, referred to by its matrix of vertex positions **B**. To this notation, is also added the matrix $\mathcal{P}$ denoting the smoothed (thin-plate) version of **P**. The complete blending process performs a transformation $\mathbf{P} \rightarrow \mathbf{P}'$ so that the shell version of blended patch $\mathcal{P}$ matches the base surface.

Regarding the involved software and tools, most of the routine is built with the Libigl library [57] in Python, while the samples were prepared with Blender [58] and MeshLab [59].

3.1. Thin-plate generation

As briefly described in the literature section, it is often helpful to look at surface details as features lying on a smooth curved shell, called the thin-plate. This two-level structure is essential for this approach, as transferring the features onto a new model can be seen as deforming the thin-plate to fit the model's surface and then reconstructing the detail elements in the new dispositions. The thin-plate $\mathcal{P} = [\boldsymbol{p}_1, \boldsymbol{p}_2, \ldots, \boldsymbol{p}_N]$ we seek is a softened version of **P**, and it should match the original surface on those vertices which are not part of irregular areas while interpolating the feature-occupied regions. There is no alteration of mesh topology:

$$\begin{aligned} \mathbf{T}_{thin} : \mathbb{R}^{3\times N} \times \mathbb{N}^* &\longrightarrow \mathbb{R}^{3\times N} \\ (\mathbf{P}, \mathcal{I}_{smooth}) &\longmapsto \mathcal{P} = \{\boldsymbol{p}_i\}_N \end{aligned} \tag{4}$$

As an overview, $\mathbf{T}_{thin}$ is a surface smoothing transformation which alters only those vertices contained in high-frequency shape regions $i \in \mathcal{I}_{feature}$.

Segmentation of the patch. It is required to differentiate the vertices contained in high-frequency regions from the rest, and the Laplacian serves as metric to differentiate smooth from detailed regions. The implementation applies a single step Laplacian smoothing [50] and evaluates the suffered displacements and normal rotation. A single step Laplacian softening

$$\begin{aligned} \mathbf{T}_{smooth} : \mathbb{R}^{3\times N} &\longrightarrow \mathbb{R}^{3\times N} \\ \mathbf{P} &\longmapsto (\mathbf{P} - \mathbf{E}), \end{aligned} \tag{5}$$

were $\mathbf{E} = \{\boldsymbol{\epsilon}_i \in \mathbb{R}^3\}$ are vertex-wise displacements after smoothing, which is obtained by solving:

$$(\mathbf{M} + \mathbf{L})(\mathbf{P} - \mathbf{E}) = \mathbf{L}\mathbf{P}, \tag{6}$$

which leads to

$$(\mathbf{M} + \mathbf{L})\mathbf{E} = \mathbf{M}\mathbf{P}, \tag{7}$$

where **M** is the mass matrix of the mesh, **L** is the cotangent Laplacian. We then evaluate per-vertex displacement and rotation, although choosing per-face computations would be an option too:

$$R(i) = \hat{F}_d(i) + \hat{F}_r(i) \tag{8}$$

$$F_d(i) = \|\boldsymbol{\epsilon}_i\| \tag{9}$$

The rotational term $F_r(i)$ can be calculated in several ways, it is a measure of angle shift for each vertex. Doing a polar decomposition for each triangle deformation is an option, comparing the normal map before and after the smoothing would be a valid approach too. The hats in $\hat{F}_d$ and $\hat{F}_r$ indicate that the corresponding fields are normalized to the unit. $R(i)$ is a height map over the vertices, which highlights high inflexion areas. By imposing a threshold value on $R(i)$, we can isolate the vertices in detail containing regions (see Fig. 7), stored as an index set

$$\mathcal{I}_{feature} = \{i \mid R(i) > tresh\},$$

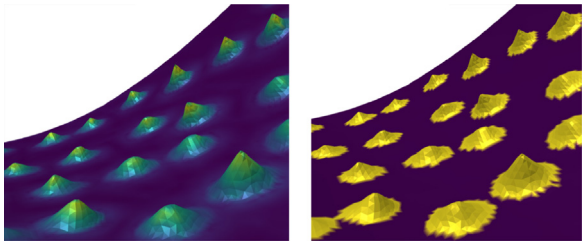

Fig. 7. Height field $R(i)$ of position and normal displacement; and segmentation map obtained from imposing a threshold on the height-map. Light areas resemble the vertices in $\mathcal{I}_{feature}$, the rest (darker areas) are considered non-feature containing regions $\mathcal{I}_{smooth}$.

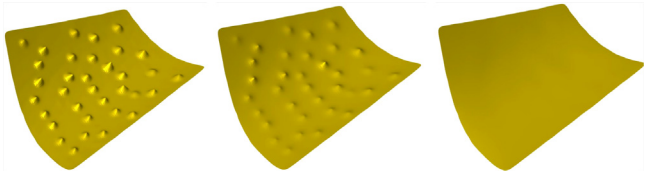

Fig. 8. *From left to right:* the original mesh has their irregularities (high-frequency features) removed by softening until only a soft surface is left (low-frequency feature).

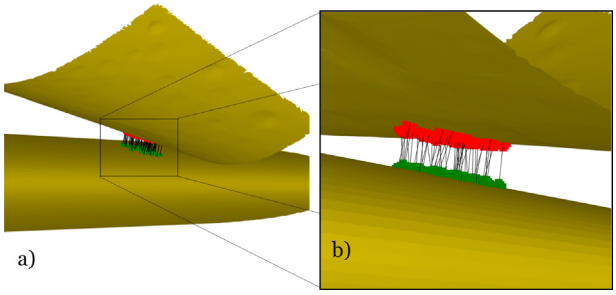

Fig. 9. Visual representation of the placement reference map $m(i)$. (a) In our experiments, references are provided as the closer vertex pairs between base and patch. (b) The green dots represent $\{\mathbf{b}_{m(i)}\}$ and the red dots are their corresponding vertices in the patch.

It is also useful to denote the index set of vertices excluded from high-frequency regions,

$$\mathcal{I}_{smooth} = \{i \mid R(i) < tresh\}.$$

Removing features from the patch. The thin-plate $\boldsymbol{\mathcal{P}}$ is generated by minimizing the biharmonic energy (Eq. (1)) in $\mathcal{I}_{feature}$ while holding the soft regions $\mathcal{I}_{smooth}$ in their original disposition. Fixing the position of the non-feature-containing regions guarantees that the resulting softened surface accurately portrays the patch's underlying curvature:

$$\Delta^2 \boldsymbol{p}_i = 0 \quad \text{subject to} \quad \boldsymbol{p}_i = \mathbf{p}_i \text{ if } i \in \mathcal{I}_{smooth}. \tag{10}$$

This softened version of the patch can be thought of as a supporting plate for the features, which will be utilized to compute the mapping to the base surface (see Fig. 8). The details are then recreated in the new configuration.

3.2. Fitting the thin-plate onto the base

The next step is to place the thin-plate onto the base's surface, following criteria of shape preservation. The placement is decided by providing positional references between both surfaces (Fig. 9). Those can be defined by mapping selected vertices of the thin-plate $\boldsymbol{\mathcal{P}}$ (with indices stored in $\mathcal{I}_{map}$) to vertices of $\mathbf{B}$:

$$m : \mathcal{I}_{map} \longrightarrow \mathbb{N} \tag{11}$$

The method through which those references are chosen varies depending on the intention behind blending the surfaces. For a precise placement, the user could provide piecewise correspondence between a few vertices. For the test presented here, the twentieth closest points between the patch's and base's surface are selected (see Fig. 9).

Unfolding meshes to the planar space. An intuitive way to compare surface locations is mapping both meshes $\boldsymbol{\mathcal{P}}$, $\mathbf{B}$ to the two-dimensional plane while preserving geodesic distances. The planarized – aka unfolded – meshes are later aligned by minimizing distances between each pair $\{\mathbf{p}_i, \mathbf{b}_{m(i)}\}$. To simplify computations, the surface region of the base that concerns the blending process is isolated from the rest of the model (see Fig. 10). For that, the vertices $\{\mathbf{b}_{m(i)}\}$ are selected, and the adjacent ones are iteratively added, up to a certain number of iterations. Another approach to this region isolation step would be to measure the maximum geodesic distance d_{max} from a vertex $\boldsymbol{p}_i$ in $\mathcal{I}_{map}$ to the edge of the thin-plate, and then from $\mathbf{b}_{m(i)}$ measure its geodesic distance to the rest of the base's vertices, gathering the ones within d_{max}. That alternative could be much more computationally expensive in dense meshes.

The thin-plate and isolated base region are then planarized using the Libigl library's rigid deformation method (ARAP) [51,60,61] (see Fig. 11). We chose this method as it maintains the relative distance between vertices after they unfold. The vertex positions of the planar surfaces are stored in matrices $\boldsymbol{\mathcal{P}}_{\text{uv}}$ and $\mathbf{B}_{\text{uv}}$.

$$\mathbf{T}_{unfold} : \mathbb{R}^{3\times N} \longrightarrow \mathbb{R}^{2\times N}, \quad \mathcal{P} \longmapsto \mathcal{P}_{\text{uv}} \tag{12}$$

Alignment of meshes. Aligning both unfolded surfaces is a matter of translating and rotating the patch until the references $\{\mathbf{p}_i, \mathbf{b}_{m(i)}\}$ match as much as possible (see Fig. 13a). The alignment process is formulated as a Procrustes problem: we seek a rigid transformation matrix to align the reference vertices, stored in a matrix $\mathbf{A} = \left[\boldsymbol{p}_{\text{uv},i}\right]_{\mathcal{I}_{map}}$ to their corresponding vertices of the base, stored in a matrix $\mathbf{C} = \left[\boldsymbol{b}_{\text{uv},m(i)}\right]$. The derived transformation $\mathbf{R}$ is later applied to all vertices of $\boldsymbol{\mathcal{P}}_{\text{uv}}$:

$$\boldsymbol{\mathcal{P}}'_{\text{uv}} = \mathbf{R}\boldsymbol{\mathcal{P}}_{\text{uv}}, \tag{13}$$

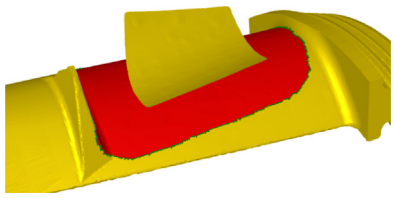

Fig. 10. Thin-plate and base in disposition for blending. The highlighted region is the cut region isolated from the base surface, obtained from a seeding and grow technique.

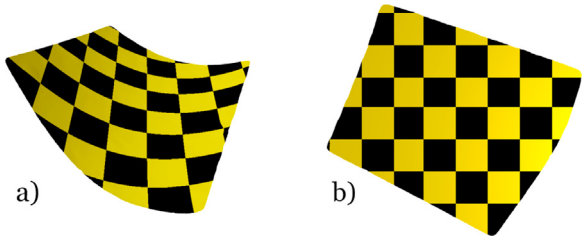

Fig. 11. (a) Thin-plate $\boldsymbol{\mathcal{P}}$ resembling the overall shape of the patch. (b) Planarized thin-plate $\boldsymbol{\mathcal{P}}_{\text{uv}}$.

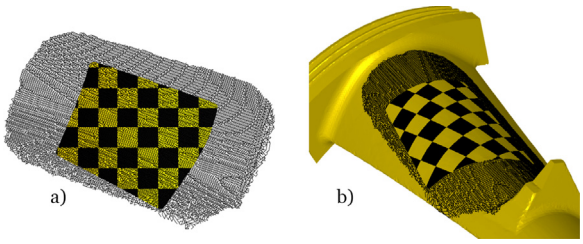

Fig. 12. (a) Planar base $\mathbf{B}_{uv}$ (wireframe) and aligned thin-plate $\mathcal{P}'_{uv}$ (square texture). (b) Fitted thin-plate $\mathcal{P}'$ onto the base surface.

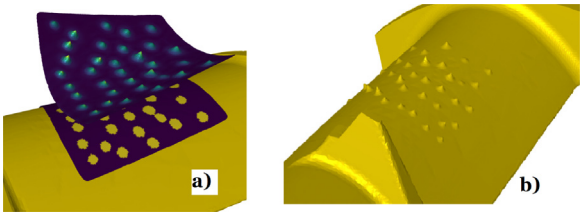

Fig. 13. (a) Original patch $\mathbf{P}$ and fitted thin-plate $\mathcal{P}'$. The segmentation map is represented over both meshes to illustrate the new disposition of the features.

$\mathbf{R} = \text{argmin}_{\mathbf{R}} \|\mathbf{RA} - \mathbf{C}\|^2$ subject to $\mathbf{R}^T\mathbf{R} = \mathbf{I}$.

Reverting the planarization. With the meshes aligned in the planar space, it is only required to undo the planarization of the base surface while carrying the thin-plate vertex positions along with it. The vertices $\boldsymbol{p}'_{uv,i}$ are reinterpreted as barycentric coordinates respect to the triangles in $\mathbf{B}_{uv}$. Each vertex $\boldsymbol{p}'_{uv,i}$ lies inside one base's mesh triangle $t_i = \{a, b, c\}$, the correlation can be represented as an $|\mathcal{P}_{uv}| \times |\mathbf{B}_{uv}|$ adjacency matrix $\mathbf{A}$ so that

$$A_{i,j} = \begin{cases} 1 & j \in t_i \\ 0 & \text{otherwise} \end{cases} \tag{14}$$

The barycentric coordinates $\mathbf{w}_i = w_{i,j}$ are first derived within the planar space, and later used to calculate the thin-plate vertices along the original (non-planar) base's surface:

$$\mathbf{W} = \mathcal{P}'_{uv} \left(\mathbf{AB}_{uv}\right)^{-1}, \tag{15}$$

$$\mathcal{P}' = \mathbf{WAB} \tag{16}$$

being $\boldsymbol{p}'_i$ the vertex positions of the, now fitted onto the base, thin-plate mesh $\mathcal{P}'$ (see Fig. 12).

3.3. Feature reconstruction

The reconstruction parts from the thin-plate are fitted to the base surface (see Fig. 13b). The original patch $\mathbf{P}$ is now deformed to match $\mathcal{P}'$ at smooth regions, while the features deform to follow the new curvature and position. To minimize distortions, this step is driven by a rigid deformation algorithm, although later, biharmonic methods are also discussed.

The ARAP routine utilized is provided by the Libigl library. This transformation takes as input the original mesh $\mathbf{P}$, the fitted thin-plate $\mathcal{P}'$ and the index set of vertices excluded from feature regions $\mathcal{I}_{smooth}$.

$$\begin{aligned} T_{rec}: \ & \mathbb{R}^{3\times N} \times \mathbb{R}^{3\times N} \times \mathbb{N}^* \longrightarrow \mathbb{R}^{3\times N} \\ & (\mathbf{P}, \mathcal{P}', \mathcal{I}_{smooth}) \longmapsto \mathbf{P}' \end{aligned} \tag{17}$$

Better described by the specific implementation, the reconstruction comprises a rigid deformation $\mathbf{P} \rightarrow \mathbf{P}'$ for which the fitted thin-plate $\mathcal{P}'$ provides the boundary condition $\mathbf{p}_i = \boldsymbol{p}'_i \ \ \forall i \in \mathcal{I}_{smooth}$. After the reconstruction process, the blended patch $\mathbf{P}'$ matches the base shape, and features inherit the new positions, orientations and underlying curvature with minimal bending, keeping their shape appearance as much as possible (see Fig. 13b). Compared to other reconstruction methods such as a biharmonic deformation, ARAP algorithms do a much better work inferring the local surface rotation and, therefore, the new orientations. Biharmonic deformation methods are affected by the original disposition, yielding tilt and distortion on the features (see Fig. 14).

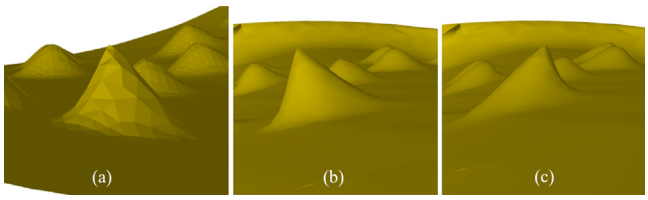

Fig. 14. (a): A particular bump on the original, undeformed patch. (b): same bump after a rigid reconstruction of detail. (c): same bump after a biharmonic reconstruction. The bump reconstructed by biharmonic means is tilted because of a substantial change of orientation from the original.

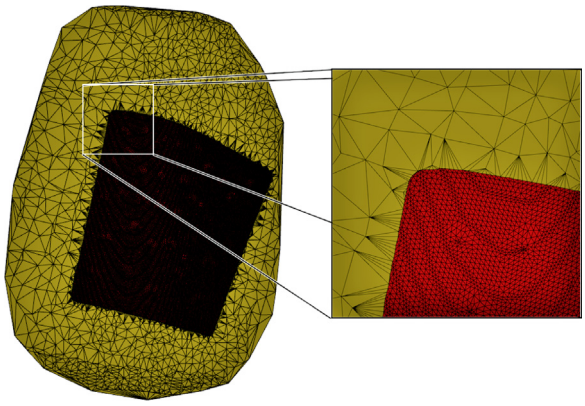

Fig. 15. Planar patch and based stitched at the edge of the patch, through a Delaunay triangulation. Low-quality meshes are used in this example to stress test the routine.

Stitching vs Morphing meshes together. At this point, the patch is already wrapped around the blade's surface, but features are yet stored in separated meshes. Getting a single unique mesh from the fit can be solved through different methods. This process has not been explored in depth during this research because it varies depending on the application. For aesthetic purposes, re-meshing might be allowed as mesh quality is not such an essential constraint, while for simulation, altering the connectivity of the mesh grid might be a concern, as it affects simulation results. It also must take into account mismatches between mesh topology and resolution. For the sake of discussion, we explored a couple of implementations, but an actual evaluation of their effect on mesh quality is kept for further work. These mesh merging methods comprise stitching the meshes together and morphing the base mesh to mimic features after the wrapping of the patch.

Both approaches have pros and cons; stitching the patch's edge to the base mesh is an option, and it could be performed right after the planar alignment. The triangle quality at the stitching region will be deficient without a post-processing optimization of the meshes to get and smooth transition (Fig. 15). Most mesh optimization packages could solve this issue (e.g. Pygmsh, trimesh, GDAL...). Also, note that the original mesh connectivity is lost. Morphing the original mesh is an option that would preserve the original connectivity. The resolution and topology of either mesh might not coincide with the outcome of obtaining a creased,

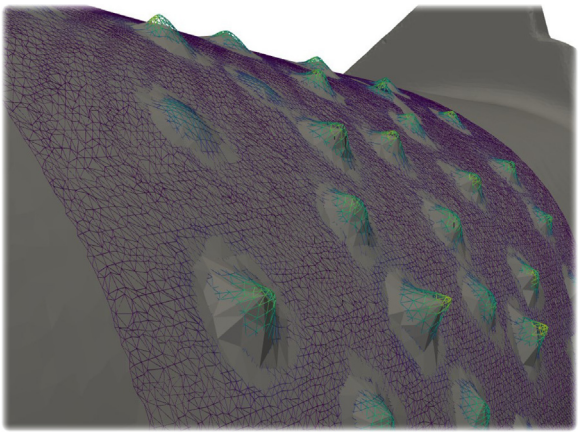

Fig. 16. Wireframe: wrapped patch $\mathbf{P}'$. The base model's mesh is morphed to mimic the features by calculating its barycentric coordinates with respect to the patch's triangles. The resolution of the base model is lower than the patch's, and therefore, its representation of the features is limited, adding crease edges.

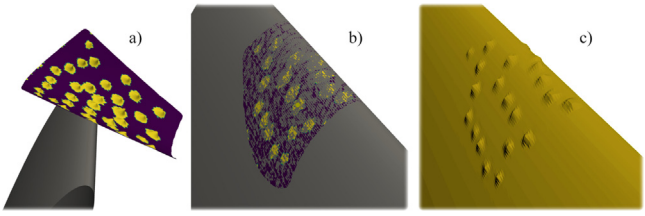

Fig. 17. Bump patch transferred to the leading edge of a blade model. (a) Patch and base in disposition for blending. (b) Thin-plate mapped to the base surface, the change of curvature forces certain bending of the patch to preserve its proportions. This requirement of proportion conservation was bypassed if required by the user. (c) Bump features inherited by the base model.

low-affinity representation of the irregularities (Fig. 16). The issue could be solved by refining the base mesh resolution and applying post-processing optimizations (e.g. smoothing).

4. Discussion and example cases

Some experiments are shown here to demonstrate the routine's capabilities and limitations. Fitting curved patches over surfaces with acute bending appears to be possible with this routine, as shown in Fig. 17. Note that the final distribution of the features depends on how the thin-plate is mapped to the base. The particular mapping method described before strives to preserve the relative distances between every feature but, if reallocating individual features was required, the thin-plate could be allowed to stretch to match the imposed allocations.

Scratch patterns can also be transferred from non curved patches, getting the scratches to bend around the leading edge of a blade and then extending through the airfoil. The resolution of the base model is a limiting factor to represent the inherited details, if no refining step is undertaken (see Fig. 18). To workaround this problem we propose to keep the patch as a separate entity, until a final output geometry is required by the user, with an specified process that might vary depending on the application.

Another interesting situation emerges when dealing with features defined by sharp edges, an extreme case being extruded cube cages over the patch (see Fig. 19). For these features, the segmentation algorithm still performs, and so do the mapping and reconstruction steps, but light distortions result from the curvature change (see Fig. 20).

These particular bends appear as a result of imposing rigid deformation constraints, because the inherited underlying curvature

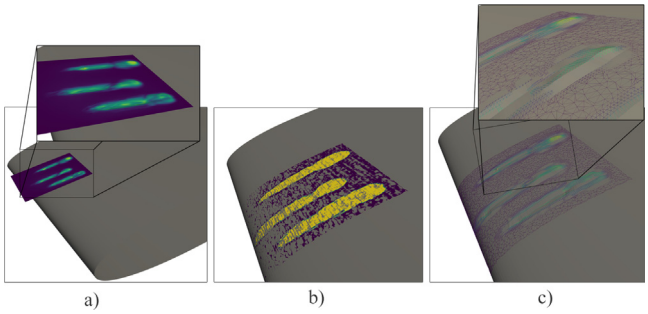

Fig. 18. Scratch pattern patch, blended onto the edge of a turbine blade. (a) Patch and base model in disposition for blending. (b) Thin-plate mapped onto the model's surface. (c) Patch features reconstructed with their new disposition, shown as a wireframe for mesh quality demonstration.

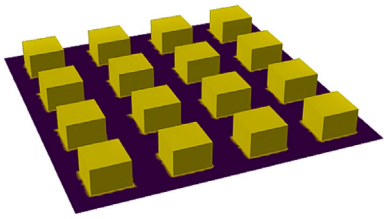

Fig. 19. Patch with applied segmentation map. The cages are correctly isolated from the rest of the surface.

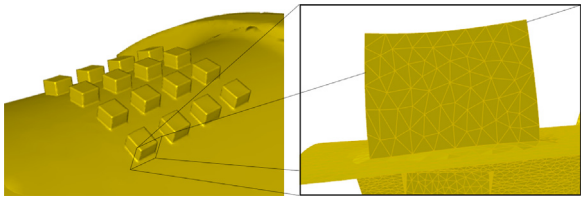

Fig. 20. Blended cage patch, the cages present light bending as a result of the rigid reconstruction.

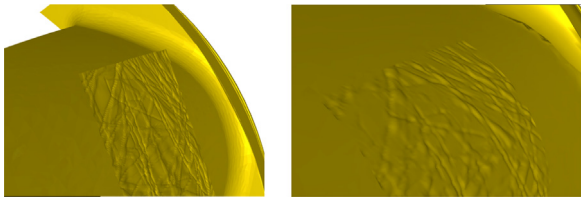

Fig. 21. Texture containing patch being blended onto a curved base. The irregularities get distorted, and the patch boundaries are not integrated with the base surface.

spreads from the feature boundary and distributes along their shape. Distortions are hard to be measured in terms of quality, as they depend of the intentions when transferring features, different methods behave differently in terms of preserving the appearance of the original details. Biharmonic methods might avoid bending distortions at the price of being non rotation-invariant and reconstruct the geometry merely as an interpolation process.

The last example includes a patch fully covered with an irregular noise texture, manually produced with the software Blender. The segmentation algorithm does not fully decouple the texture from the overall shape, as the texture covers every spot of the patch area. Also, the boundaries of the patch would require an extra step to soften the transition between the soft base surface and the patch irregularities (see Fig. 21).

5. Conclusions

An algorithm for performing surface feature transfer has been described and demonstrated with triangular mesh models and patches. The tool relies on the user to set reference locations between both shapes and for adjusting a threshold to partition the mesh into soft and irregular regions. More control could be added to allow the patch to be stretched and meet more specific positioning requirements. The resulting algorithm provides reliable results even if dramatic deformations occur, performing exceptionally well in the case of isolated features embedded on a soft supporting surface. In situations where those features fill the entire patch area (Fig. 21), the reconstructed details get softened if the initial partitioning is not generated with careful control. If the features reach the patch boundary, or in other words, if the patch boundary contains important features, it is necessary adding extra computation to blend them with the base. Stability is another issue, as non-linear methods, like rigid deformation, are sensitive to low-quality meshes and may flip polygons during the optimization process or never get to converge. The solution to this problem was to introduce a coarser version of the thin-plate mesh that serves as skinning lattice to simplify deformation steps.

Further study is aimed towards extensive testing of the limitations, a more generalized blending algorithm that deals with patches of arbitrary topology and, at some point, further integration into reverse engineering pipelines might be proposed.

Declaration of competing interest

The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: The results presented on this manuscript are the output of a PhD program on its midterm, partially funded by Rolls-Royce PLM. Any data or samples presented in figures are not property of Rolls Royce, and are not directly related to Rolls Royce activities.

References

[1] Booyse W, Wilke DN, Heyns S. Deep digital twins for detection, diagnostics and prognostics. Mech Syst Signal Process 2020;140:106612. http://dx.doi.org/10.1016/j.ymssp.2019.106612.

[2] Tarkhov DA, Malykhina GF. Neural network modelling methods for creating digital twins of real objects. J Phys Conf Ser 2019;1236:012056. http://dx.doi.org/10.1088/1742-6596/1236/1/012056.

[3] Lermer M, Reich C. Creation of digital twins by combining fuzzy rules with artificial neural networks. In: IECON 2019 - 45th annual conference of the IEEE industrial electronics society, Vol. 1. 2019, p. 5849–54. http://dx.doi.org/10.1109/IECON.2019.8926914.

[4] Dawes WN, Meah N, Kudryavtsev A, Evans R, Hunt M, Tiller P. Digital geometry to support a gas turbine digital twin. In: AIAA scitech 2019 forum. American Institute of Aeronautics and Astronautics; 2019.

[5] Bogdan SL, Nedelcu D, Pădurean I. The reverse engineering technique performed on a francis runner geometry through photogrammetry. IOP Conf Ser Mater Sci Eng 2019;477:012021. http://dx.doi.org/10.1088/1757-899X/477/1/012021.

[6] Funkhouser T, Kazhdan M, Shilane P, Min P, Kiefer W, Tal A, et al. Modeling by example. In: ACM SIGGRAPH 2004 papers. SIGGRAPH '04, New York, NY, USA: Association for Computing Machinery; 2004, p. 652–63. http://dx.doi.org/10.1145/1186562.1015775.

[7] Lin J, Jin X, Wang C, Hui K-C. Mesh composition on models with arbitrary boundary topology. IEEE Trans Vis Comput Graph 2008;14(3):653–65. http://dx.doi.org/10.1109/TVCG.2007.70632.

[8] Huang X, Fu H, Au OK-C, Tai C-L. Optimal boundaries for Poisson mesh merging. In: Proceedings of the 2007 ACM symposium on solid and physical modeling. SPM '07, Beijing, China: Association for Computing Machinery; 2007, p. 35–40. http://dx.doi.org/10.1145/1236246.1236254.

[9] Kammann L, Menzel S, Botsch M. A compact patch-based representation for technical mesh models. The Eurographics Association; 2020, http://dx.doi.org/10.2312/vmv.20201182.

[10] Buonamici F, Carfagni M, Furferi R, Governi L, Lapini A, Volpe Y. Reverse engineering of mechanical parts: A template-based approach. J Comput Des Eng 2018;5(2):145–59. http://dx.doi.org/10.1016/j.jcde.2017.11.009.

[11] Buonamici F, Lapini A. An overview of constrained fitting optimization techniques for reverse engineering of mechanical parts, Vol. 11. 2017, p. 9.

[12] Li Z, Zhou X, Liu W. A geometric reasoning approach to hierarchical representation for b-rep model retrieval. Comput Aided Des 2015;62:190–202. http://dx.doi.org/10.1016/j.cad.2014.05.008.

[13] Yin K, Chen Z, Chaudhuri S, Fisher M, Kim VG, Zhang H. COALESCE: component assembly by learning to synthesize connections. 2020, [Cs] arXiv:2008.01936. arXiv:2008.01936.

[14] Li J, Xu K, Chaudhuri S, Yumer E, Zhang H, Guibas L. GRASS: generative recursive autoencoders for shape structures. ACM Trans Graph 2017;36(4):1–12. http://dx.doi.org/10.1145/3072959.3073613, arXiv:1705.02090.

[15] Qi CR, Yi L, Su H, Guibas LJ. PointNet++: Deep hierarchical feature learning on point sets in a metric space. In: Proceedings of the 31st international conference on neural information processing systems. NIPS'17, Red Hook, NY, USA: Curran Associates Inc. 2017, p. 5105–14.

[16] Groueix T, Fisher M, Kim VG, Russell BC, Aubry M. 3D-CODED : 3D correspondences by deep deformation. 2018, [Cs]. arXiv:1806.05228. arXiv:1806.05228.

[17] Zhu C, Xu K, Chaudhuri S, Yi R, Zhang H. SCORES: Shape composition with recursive substructure priors. ACM Trans Graph 2018;37(6):211:1–211:14. http://dx.doi.org/10.1145/3272127.3275008.

[18] Li J, Niu C, Xu K. Learning part generation and assembly for structure-aware shape synthesis. 2020, [Cs] arXiv:1906.06693. arXiv:1906.06693.

[19] Sung M, Su H, Kim VG, Chaudhuri S, Guibas L. ComplementMe: Weakly-supervised component suggestions for 3D modeling. ACM Trans Graph 2017;36(6):226:1–226:12. http://dx.doi.org/10.1145/3130800.3130821.

[20] Schmidt R, Singh K. Drag, drop, and clone: an interactive interface for surface composition. Technical report CSRG-611, 2010, p. 10.

[21] Jaiswal P, Huang J, Rai R. Assembly-based conceptual 3D modeling with unlabeled components using probabilistic factor graph. Comput Aided Des 2016;74:45–54. http://dx.doi.org/10.1016/j.cad.2015.10.002.

[22] Chaudhuri S, Koltun V. Data-driven suggestions for creativity support in 3D modeling. ACM Trans Graph 2010;29(6):183:1–183:10. http://dx.doi.org/10.1145/1882261.1866205.

[23] Li Y, Mo K, Shao L, Sung M, Guibas L. Learning 3D part assembly from a single image. 2020, [Cs] arXiv:2003.09754. arXiv:2003.09754.

[24] Chen Z, Zhang H. Learning implicit fields for generative shape modeling. In: 2019 IEEE/CVF conference on computer vision and pattern recognition. 2019, p. 5932–41. http://dx.doi.org/10.1109/CVPR.2019.00609.

[25] Sinha A, Bai J, Ramani K. Deep learning 3D shape surfaces using geometry images. In: Leibe B, Matas J, Sebe N, Welling M, editors. Computer vision – ECCV 2016. Lecture notes in computer science, Cham: Springer International Publishing; 2016, p. 223–40. http://dx.doi.org/10.1007/978-3-319-46466-4_14.

[26] Han X, Li Z, Huang H, Kalogerakis E, Yu Y. High-resolution shape completion using deep neural networks for global structure and local geometry inference. In: 2017 IEEE international conference on computer vision. 2017, p. 85–93. http://dx.doi.org/10.1109/ICCV.2017.19.

[27] Jogin M, Mohanam, Madhulika MS, Divya GD, Meghana RK, Apoorva S. Feature extraction using convolution neural networks (CNN) and deep learning. In: 2018 3rd IEEE international conference on recent trends in electronics, information communication technology. 2018, p. 2319–23. http://dx.doi.org/10.1109/RTEICT42901.2018.9012507.

[28] Xu Z, Kang R, Lu R. 3D reconstruction and measurement of surface defects in prefabricated elements using point clouds. J Comput Civ Eng 2020;34(5):04020033. http://dx.doi.org/10.1061/(ASCE)CP.1943-5487.0000920.

[29] Zeng A, Song S, Nießner M, Fisher M, Xiao J, Funkhouser T. 3DMatch: learning local geometric descriptors from RGB-d reconstructions. 2017, [Cs] arXiv:1603.08182. arXiv:1603.08182.

[30] Brock A, Lim T, Ritchie JM, Weston N. Generative and discriminative voxel modeling with convolutional neural networks. 2016, [Cs, Stat] arXiv:1608.04236. arXiv:1608.04236.

[31] Khan S, Awan MJ. A generative design technique for exploring shape variations. Adv Eng Inform 2018;38:712–24. http://dx.doi.org/10.1016/j.aei.2018.10.005.

[32] Li H, Lachmayer R. Generative design approach for modeling creative designs. IOP Conf Ser Mater Sci Eng 2018;408:012035. http://dx.doi.org/10.1088/1757-899X/408/1/012035.

[33] Nordin A. Challenges in the industrial implementation of generative design systems: an exploratory study. AI EDAM 2018;32(1):16–31. http://dx.doi.org/10.1017/S0890060416000536.

[34] Inoue S, Konishi M, Imai J. Surface defect inspection of a cutting tool by image processing with neural networks. Mem Fac Eng Okayama Univ 2009;43:55–60.

[35] Schleich B, Anwer N, Mathieu L, Wartzack S. Skin model shapes: A new paradigm shift for geometric variations modelling in mechanical engineering. Comput Aided Des 2014;50:1–15. http://dx.doi.org/10.1016/j.cad.2014.01.001.
[36] Forrester JA. On the characterization of measured geometry for the facilitation of uncertainty propagation and robust design (Ph.D. thesis), University of Southampton; 2019.
[37] Chang K-H, Chen C. 3D shape engineering and design parameterization. Comput Aided Des Appl 2011;8(5):681–92. http://dx.doi.org/10.3722/cadaps.2011.681-692.
[38] Marinov M, Amagliani M, Barback T, Flower J, Barley S, Furuta S, et al. Generative design conversion to editable and watertight boundary representation. Comput Aided Des 2019;115:194–205. http://dx.doi.org/10.1016/j.cad.2019.05.016.
[39] Benkő P, Martin RR, Várady T. Algorithms for reverse engineering boundary representation models. Comput Aided Des 2001;33(11):839–51. http://dx.doi.org/10.1016/S0010-4485(01)00100-2.
[40] Vorray RA, O'Driscoll G, Steed A. Reverse engineering polygonal meshes using discrete differential geometry. Comput Aided Des Appl 2008;5(1–4):86–98. http://dx.doi.org/10.3722/cadaps.2008.86-98.
[41] Mejia D, Ruiz-Salguero O, Sánchez JR, Posada J, Moreno A, Cadavid CA. Hybrid geometry / topology based mesh segmentation for reverse engineering. Comput Graph 2018;73:47–58. http://dx.doi.org/10.1016/j.cag.2018.03.004.
[42] Mejia Parra D, Ruiz O, Cadavid C. Spectral-based mesh segmentation. Int J Interact Des Manuf (IJIDeM) 2017;11:503–14. http://dx.doi.org/10.1007/s12008-016-0300-0.
[43] Leloudas SN, Strofylas GA, Nikolos IK. Airfoil Optimization Using Area-preserving free-form deformation. In: ASME 2015 international mechanical engineering congress and exposition. American Society of Mechanical Engineers Digital Collection; 2016, http://dx.doi.org/10.1115/IMECE2015-50904.
[44] Leloudas SN, Strofylas GA, Nikolos IK. Constrained airfoil optimization using the area-preserving free-form deformation. Aircr Eng Aerosp Technol 2018;90(6):914–26. http://dx.doi.org/10.1108/AEAT-10-2016-0184.
[45] Sedai A, Thapa BS, Thapa B, Kapali A, Qian Z, Guo Z. Application of reverse engineering method to model eroded francis runner. J Phys Conf Ser 2020;1608:012012. http://dx.doi.org/10.1088/1742-6596/1608/1/012012.
[46] Jacobson A, Deng Z, Kavan L, Lewis JP. Skinning: real-time shape deformation. In: ACM SIGGRAPH 2014 courses. 2014.
[47] Jacobson A, Baran I, Popović J, Sorkine O. Bounded biharmonic weights for real-time deformation. ACM Trans Graph 2011;30(4):78:1–8. http://dx.doi.org/10.1145/2010324.1964973.
[48] Coppedé A, Vernengo G, Villa D. A combined approach based on subdivision surface and free form deformation for smart ship hull form design and variation. Ships Offshore Struct 2018;13(7):769–78. http://dx.doi.org/10.1080/17445302.2018.1457235.
[49] Xu D, Zhang H, Wang Q, Bao H. Poisson shape interpolation. In: Proceedings of the 2005 ACM symposium on solid and physical modeling. SPM '05, New York, NY, USA: Association for Computing Machinery; 2005, p. 267–74. http://dx.doi.org/10.1145/1060244.1060274.
[50] Sorkine O, Cohen-Or D, Lipman Y, Alexa M, Rössl C, Seidel H-P. Laplacian surface editing. In: Proceedings of the 2004 Eurographics/ACM SIGGRAPH symposium on geometry processing - SGP '04. Nice, France: ACM Press; 2004, p. 175. http://dx.doi.org/10.1145/1057432.1057456.
[51] Sorkine O, Alexa M. As-rigid-as-possible surface modeling. In: Proceedings of the fifth eurographics symposium on geometry processing. SGP '07, Goslar, DEU: Eurographics Association; 2007, p. 109–16.
[52] Botsch M, Sumner RW, Pauly M, Gross M. Deformation transfer for detail-preserving surface editing. In: Vision, modeling, and visualization 2006 : proceedings, November 22-24, 2006, Aachen, Germany. Akademische Verlagsgesellschaft AKA; 2006, p. 357–64.
[53] Huang H, Gong M, Cohen-Or D, Ouyang Y, Tan F, Zhang H. Field-guided registration for feature-conforming shape composition. ACM Trans Graph 2012;31(6):179:1–179:11. http://dx.doi.org/10.1145/2366145.2366198.
[54] Sorkine O. Laplacian mesh processing. Eurographics 2005 - state of the art reports, The Eurographics Association; 2005, http://dx.doi.org/10.2312/egst.20051044.
[55] Roy M, Foufou S, Koschan A, Truchetet F, Abidi M. Multiresolution analysis for irregular meshes. In: Truchetet F, editor. Photonics technologies for robotics, automation, and manufacturing. Providence, RI; 2004, p. 249. http://dx.doi.org/10.1117/12.515974.
[56] Helenbrook BT. Mesh deformation using the biharmonic operator. Int J Numer Methods Eng 2003;56(7):1007–21. http://dx.doi.org/10.1002/nme.595.
[57] Jacobson A, Panozzo D, et al. Libigl: A simple c++ geometry processing library. 2018.
[58] Community BO. Blender - a 3D modelling and rendering package. Amsterdam: Blender Foundation, Stichting Blender Foundation; 2018.
[59] Cignoni P, Callieri M, Corsini M, Dellepiane M, Ganovelli F, Ranzuglia G. MeshLab: An open-source mesh processing tool. In: Eurographics Italian chapter conference. The Eurographics Association; 2008, p. 8. http://dx.doi.org/10.2312/LOCALCHAPTEREVENTS/ITALCHAP/ITALIANCHAPCONF2008/129-136.
[60] Chao I, Pinkall U, Sanan P, Schröder P. A simple geometric model for elastic deformations. ACM Trans Graph 2010;29(4):1–6. http://dx.doi.org/10.1145/1778765.1778775.
[61] McAdams A, Selle A, Tamstorf R, Teran J, Sifakis E. Computing the singular value decomposition of 3x3 matrices with minimal branching and elementary floating point operations. Technical report, University of Wisconsin-Madison Department of Computer Sciences; 2011.