Shape optimisation with multiresolution subdivision surfaces and immersed finite elements

Kosala Bandara, Thomas Rüberg, Fehmi Cirak*

Department of Engineering, University of Cambridge, Trumpington Street, Cambridge CB2 1PZ, UK

Abstract

We develop a new optimisation technique that combines multiresolution subdivision surfaces for boundary description with immersed finite elements for the discretisation of the primal and adjoint problems of optimisation. Similar to wavelets, multiresolution surfaces represent the domain boundary using a coarse control mesh and a sequence of detail vectors. Based on the multiresolution decomposition efficient and fast algorithms are available for reconstructing control meshes of varying fineness. During shape optimisation the vertex coordinates of control meshes are updated using the computed shape gradient information. By virtue of the multiresolution editing semantics, updating the coarse control mesh vertex coordinates leads to large-scale geometry changes and, conversely, updating the fine control mesh coordinates leads to small-scale geometry changes. In our computations we start by optimising the coarsest control mesh and refine it each time the cost function reaches a minimum. This approach effectively prevents the appearance of non-physical boundary geometry oscillations and control mesh pathologies, like inverted elements. Independent of the fineness of the control mesh used for optimisation, on the immersed finite element grid the domain boundary is always represented with a relatively fine control mesh of fixed resolution. With the immersed finite element method there is no need to maintain an analysis suitable domain mesh. In some of the presented two and three-dimensional elasticity examples the topology derivative is used for introducing new holes inside the domain. The merging or removing of holes is not considered.
© 2015 The Authors. Published by Elsevier B.V. This is an open access article under the CC BY license (http://creativecommons.org/licenses/by/4.0/).

Keywords: Shape optimisation; Immersed finite elements; Multiresolution surfaces; Subdivision schemes; Isogeometric analysis; Catmull–Clark

1. Introduction

We consider the shape optimisation of two- and three-dimensional solids by combining multiresolution subdivision surfaces with immersed finite elements. As widely discussed in isogeometric analysis literature, the geometry representations used in today's computer aided design (CAD) and finite element analysis (FEA) software are inherently incompatible [1]. This is particularly limiting in shape optimisation during which a given CAD geometry model is to be iteratively updated based on the results of a finite element computation. The inherent shortcomings

---

(a) Original control mesh and corresponding geometry.
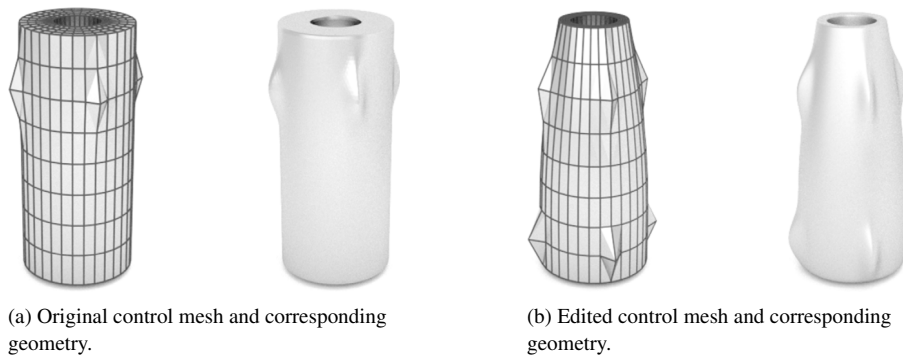
(b) Edited control mesh and corresponding geometry.

Fig. 1. Multiresolution editing of a cylindrical component with small scale geometric details in the form of small bumps. The original geometry is shown on the left. The edited version on the right contains both large and small scale changes. The change of the overall shape from a cylinder to a cone constitutes a large scale change and the additional bumps close to the bottom constitute small scale changes. The bumps in the original geometry are automatically preserved during multiresolution editing.

of present geometry and analysis representations have motivated the proliferation of various shape optimisation techniques. In the most prevalent approaches a surrogate geometry model [2–8] or the analysis mesh [9,10] instead of the true CAD model is optimised, see also [11] and references therein. Generally, it is tedious or impossible to map the optimised surrogate geometry model or analysis mesh back to the original CAD model, which is essential for continuing with the design process and later for manufacturing purposes. Moreover, geometric design features are usually defined with respect to the CAD model and cannot be easily enforced on the surrogate model. Recently, the shape optimisation of shells, solids and other applications using isogeometric analysis has been explored; that is, through directly optimising the CAD geometry model [12–15].

In the present work the domain boundary is represented with subdivision curves (in 2D) or surfaces (in 3D). Although historically subdivision and related techniques have originated in computer graphics, they recently became available in several CAD software packages, including Autodesk Fusion 360, PTC Creo and CATIA. As will be demonstrated in this paper, subdivision curves/surfaces provide an elegant isogeometric, bidirectional mapping between the geometry and analysis models. In subdivision a geometry is described using a control mesh and a limiting process of repeated refinement [16,17]. The refinement rules are usually adapted from knot refinement rules for b-splines [18–20]. The specific subdivision rules used in this work are derived from cubic b-splines. For surfaces we use the subdivision rules proposed by Catmull and Clark [20], which lead to smooth surfaces even in case of unstructured meshes with extraordinary vertices (i.e., domain vertices with number of adjacent edges different than four). The hierarchy of control meshes underlying a subdivision surface lends itself naturally to multiresolution decomposition of geometry [21,22]. To this end, suitable operators are available for decomposing a geometry in a coarse control mesh and a sequence of detail vectors similar to wavelets. Subsequently, it becomes possible to reconstruct on-the-fly control meshes of any fineness and to edit their vertex positions. The size of the geometric region influenced by each vertex depends on the resolution of the control mesh, editing coarser levels leads to large-scale changes while editing finer levels leads to small-scale changes. Importantly, after applying large-scale changes the available finer level detail vectors can be automatically added to obtain the new limit geometry, cf. Fig. 1.

During gradient-based shape optimisation the *shape gradient* is computed to determine the boundary perturbations which lead to a reduction in a given cost function. We consider the adjoint approach as applied to the continuous shape optimisation problem for computing the shape gradient [23–25]. The shape gradient is a function of the solution of the original (primal) and a complimentary adjoint boundary value problem. In case of compliance minimisation the primal and adjoint solutions are up to the sign identical and, hence, only the primal boundary value problem has to be solved. We discretise the primal and adjoint problems with immersed, or embedded, finite elements, see, e.g., [26–29], which have clear advantages when applied to structural shape optimisation [25,30,31]. The geometry of the domain boundary can be updated without needing to generate a new, or to smooth an existing, domain mesh. On the fixed non-boundary-conforming immersed finite element grid, we use tensor-product b-splines as basis functions and enforce Dirichlet boundary conditions with the Nitsche technique.

As mentioned, the developed multiresolution optimisation approach relies on multiresolution subdivision curves/surfaces for geometry description and the immersed finite element method for domain discretisation. The

multiresolution representation of the domain boundaries allows to describe the same geometry with control meshes of different resolution for analysis and optimisation purposes. For finite element analysis a relatively fine control mesh is used in order to faithfully describe the domain boundaries on the immersed grid. In contrast, the degrees of freedom in optimisation (i.e., design variables) are chosen as the vertex coordinates of a coarser control mesh. Usually, better shapes can be found by starting with a coarse control mesh and optimising increasingly finer control meshes. During the optimisation iterations the refinement level of the control mesh is increased each time a minimum is reached. Informally, the superior performance of the multiresolution approach can be explained with the correlation between the number of design variables and the number of local minima of the cost function. Having initially fewer local minima reduces the possibility of landing in a local minimum. It bears emphasis that with the employed, wavelet-like multiresolution decomposition, the coarse control mesh for optimisation can be chosen independently of the size of the features present in the geometry. For instance, the control mesh for optimisation can be chosen much coarser than any fillets or small-scale surface undulations present on the to be optimised geometry.

There are a number of prior works, especially in aerodynamic shape optimisation, that use hierarchical and adaptive geometry representations, see e.g. [32] and references therein. For instance, in [7] Bezier basis functions and degree elevation and in [8] b-splines and knot insertion are considered to create a hierarchy of geometry representations. Most of these papers primary aim is to speed up the optimisation process by reducing the number of optimisation variables or by employing multigrid techniques. However, the added benefit of hierarchical representations in reducing the parameterisation dependency of the final optimisation results is also often noted. In comparison to the mentioned techniques, the advantages of multiresolution subdivision surfaces are: (i) the ability to represent geometries with arbitrary topology, (ii) wavelet-like multiresolution representation of the geometry, and (iii) the ease of integration with CAD packages that use subdivision.

This paper is organised as follows. Section 2 introduces the governing equations for linear elastic shape optimisation. Specifically, the derivation of the continuous shape gradient using the adjoint approach is illustrated. Subsequently, Section 3 discusses the discretisation of the primary and the adjoint boundary value problems with an immersed finite element method. Section 4 forms the core of the paper and introduces multiresolution optimisation. First the derivation of the cubic b-spline and Catmull–Clark subdivision rules from the well known b-spline refinement relations is demonstrated. After that, the multiresolution decomposition of subdivision surfaces and its use for multiresolution editing are shown. Upon this basis we introduce our multiresolution shape optimisation algorithm. Finally, Section 5 presents several two- and three-dimensional examples of increasing complexity to demonstrate the efficiency and robustness of multiresolution optimisation. In some of the examples the topology derivative is considered for introducing new holes inside the domain.

2. Governing equations

We introduce in this section the shape optimisation of two- and three-dimensional linear elastic solids. In addition to the linear elasticity boundary value problem a cost function that is to be minimised is considered. For computing the shape derivatives, that is the derivatives of the cost function with respect to the domain perturbations, we use the analytic adjoint approach. As it will become clear in Section 3, with the analytic approach it is straightforward to compute the shape gradients using the immersed finite element technique. In the following, we briefly review few key results from shape calculus. See, for instance, the monographs [23–25,33] for a more detailed discussion.

2.1. Linear elasticity

The equilibrium equation for a solid body with the domain Ω is given by

$$\nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}) + \boldsymbol{f} = \boldsymbol{0} \quad \text{in } \Omega, \tag{1a}$$

$$\boldsymbol{u} = \boldsymbol{0} \quad \text{on } \Gamma_D, \tag{1b}$$

$$\boldsymbol{\sigma}(\boldsymbol{u})\boldsymbol{n} = \bar{\boldsymbol{t}} \quad \text{on } \Gamma_N, \tag{1c}$$

where $\boldsymbol{\sigma}$ is the stress tensor, $\boldsymbol{u}$ is the displacement vector, $\boldsymbol{f}$ is the external load vector and $\bar{\boldsymbol{t}}$ is the prescribed traction on the Neumann boundary Γ_N with the outward normal $\boldsymbol{n}$. On the Dirichlet boundary Γ_D, for simplicity, only homogeneous boundary conditions are prescribed.

We assume a homogeneous linear elastic material model

$$\boldsymbol{\sigma}(\boldsymbol{u}) = \boldsymbol{C} : \boldsymbol{\epsilon}(\boldsymbol{u}) \tag{2}$$

with the fourth order constitutive tensor $\boldsymbol{C}$ and linear elastic strain tensor

$$\boldsymbol{\epsilon}(\boldsymbol{u}) = \frac{1}{2}(\nabla\boldsymbol{u} + \nabla^{\mathsf{T}}\boldsymbol{u}). \tag{3}$$

2.2. Shape optimisation

2.2.1. Formulation
The cost function to be minimised

$$J(\Omega, \boldsymbol{u}) \rightarrow \min \tag{4}$$

depends on the to be optimised domain Ω and the unknown solution $\boldsymbol{u}$. The most common cost functions are integrals over the domain Ω or its boundary Γ. For brevity and without loss of generality, we consider in the following only the structural compliance as the cost function

$$J(\Omega, \boldsymbol{u}) = \int_{\Omega} \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\epsilon}(\boldsymbol{u}) \, d\Omega = \int_{\Omega} \boldsymbol{f} \cdot \boldsymbol{u} \, d\Omega + \int_{\Gamma_N} \bar{\boldsymbol{t}} \cdot \boldsymbol{u} \, d\Gamma. \tag{5}$$

It is straightforward to adapt the subsequent derivations to other common cost functions, such as integrals of stresses or displacements.

The minimisation of the cost function $J(\Omega, \boldsymbol{u})$ with the boundary value problem (1) as a constraint can be achieved by means of the Lagrangian function

$$
\begin{aligned}
L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda}) = {} & J(\Omega, \boldsymbol{u}) + \int_{\Omega} \nabla\boldsymbol{\lambda} : \boldsymbol{\sigma}(\boldsymbol{u}) \, d\Omega - \int_{\Omega} \boldsymbol{\lambda} \cdot \boldsymbol{f} \, d\Omega \\
& - \int_{\Gamma_D} \boldsymbol{u} \cdot (\boldsymbol{C} : \nabla\boldsymbol{\lambda})\boldsymbol{n} + \boldsymbol{\lambda} \cdot \boldsymbol{\sigma}(\boldsymbol{u})\boldsymbol{n} \, d\Gamma - \int_{\Gamma_N} \boldsymbol{\lambda} \cdot \bar{\boldsymbol{t}} \, d\Gamma,
\end{aligned}
\tag{6}
$$

where $\boldsymbol{\lambda}$ is a Lagrange parameter. $L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda})$ depends on the unknown domain Ω, the displacement field $\boldsymbol{u}$ and the Lagrange parameter $\boldsymbol{\lambda}$. The stationarity condition for the Lagrangian (6), i.e. $\delta L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda}) = 0$, provides the complete set of equations that describe the shape optimisation problem. In the following we consider one after the other the variation of $L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda})$ with respect to the Lagrange parameter $\boldsymbol{\lambda}$, displacements $\boldsymbol{u}$ and domain Ω.

2.2.2. Primal problem
The variation of the Lagrangian $L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda})$ with respect to $\boldsymbol{\lambda}$ reads

$$
\begin{aligned}
\frac{\partial L}{\partial \boldsymbol{\lambda}}\delta\boldsymbol{\lambda} + \frac{\partial L}{\partial(\nabla\boldsymbol{\lambda})}\delta(\nabla\boldsymbol{\lambda}) = {} & -\int_{\Omega} \delta\boldsymbol{\lambda} \cdot [\nabla \cdot \boldsymbol{\sigma}(\boldsymbol{u}) + \boldsymbol{f}] \, d\Omega - \int_{\Gamma_D} \boldsymbol{u} \cdot (\boldsymbol{C} : \nabla(\delta\boldsymbol{\lambda})) \, \boldsymbol{n} \, d\Gamma \\
& - \int_{\Gamma_N} \delta\boldsymbol{\lambda} \cdot [\bar{\boldsymbol{t}} - \boldsymbol{\sigma}(\boldsymbol{u})\boldsymbol{n}] \, d\Gamma = 0,
\end{aligned}
\tag{7}
$$

where we used the divergence theorem and $\delta(\nabla\boldsymbol{\lambda}) = \nabla(\delta\boldsymbol{\lambda})$. For arbitrary $\delta\boldsymbol{\lambda}$ it is evident that (7) is equivalent to the linear elasticity equations (1).

2.2.3. Adjoint problem
Next, we consider the variation of the Lagrangian $L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda})$ with respect to the displacements $\boldsymbol{u}$

$$
\begin{aligned}
\frac{\partial L}{\partial \boldsymbol{u}}\delta\boldsymbol{u} + \frac{\partial L}{\partial(\nabla\boldsymbol{u})}\delta(\nabla\boldsymbol{u}) = {} & \frac{\partial J}{\partial \boldsymbol{u}}\delta\boldsymbol{u} + \int_{\Omega} \nabla\boldsymbol{\lambda} : \boldsymbol{C} : \delta(\nabla\boldsymbol{u}) \, d\Omega \\
& - \int_{\Gamma_D} \delta\boldsymbol{u} \cdot (\boldsymbol{C} : \nabla\boldsymbol{\lambda})\boldsymbol{n} + \boldsymbol{\lambda} \cdot (\boldsymbol{C} : \delta(\nabla\boldsymbol{u})) \, \boldsymbol{n} \, d\Gamma = 0.
\end{aligned}
\tag{8}
$$

After introducing the cost function (5) and reformulating the domain term with the divergence theorem we obtain

$$
\int_{\Omega} \boldsymbol{f} \cdot \delta \boldsymbol{u} \ \mathrm{d}\,\Omega + \int_{\Gamma_N} \bar{\boldsymbol{t}} \cdot \delta \boldsymbol{u} \ \mathrm{d}\,\Gamma - \int_{\Omega} \delta \boldsymbol{u} \cdot (\nabla \cdot \boldsymbol{\sigma}\,(\boldsymbol{\lambda})) \ \mathrm{d}\,\Omega
$$
$$
- \int_{\Gamma_D} \boldsymbol{\lambda} \cdot (\boldsymbol{C} : \nabla(\delta \boldsymbol{u}))\,\boldsymbol{n} \ \mathrm{d}\,\Gamma + \int_{\Gamma_N} \delta \boldsymbol{u} \cdot \boldsymbol{\sigma}\,(\boldsymbol{\lambda})\boldsymbol{n} \ \mathrm{d}\,\Gamma = 0. \tag{9}
$$

The corresponding boundary value problem, referred to as the adjoint problem, reads

$$
\nabla \cdot \boldsymbol{\sigma}\,(\boldsymbol{\lambda}) - \boldsymbol{f} = \boldsymbol{0} \quad \text{in } \Omega, \tag{10a}
$$
$$
\boldsymbol{\lambda} = \boldsymbol{0} \quad \text{on } \Gamma_D, \tag{10b}
$$
$$
\boldsymbol{\sigma}\,(\boldsymbol{\lambda})\boldsymbol{n} = -\bar{\boldsymbol{t}} \quad \text{on } \Gamma_N. \tag{10c}
$$

By comparing this adjoint problem with the primal problem (1) we deduce that $\boldsymbol{\lambda} = -\boldsymbol{u}$. Note that this holds only when the cost function is the structural compliance (5).

2.2.4. Shape derivative

Finally, we consider the variation of $L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda})$ with respect to the problem domain Ω, which is also referred to as the shape derivative. To this end, we first define a linear mapping which maps a given domain Ω into a perturbed domain Ω_t, see Fig. 2. With this mapping a material point with the coordinate $\boldsymbol{x} \in \Omega$ is mapped onto

$$
\boldsymbol{x}_t = \boldsymbol{x} + t\delta \boldsymbol{v}, \tag{11}
$$

where $\delta \boldsymbol{v}$ is a prescribed vector field and t is a scalar parameter. In the usual continuum mechanics terminology, Ω is the reference configuration, Ω_t is the current configuration, $\delta \boldsymbol{v}$ is the material velocity vector and t is the (pseudo-)time. In shape optimisation literature the mapping (11) is usually expressed as

$$
\Omega_t = \Omega + t\delta \boldsymbol{v}. \tag{12}
$$

Before attempting the variation of $L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda})$, we first give the variation of generic volume and surface integrals with respect to $\delta \boldsymbol{v}$. The variation of a domain integral of the scalar function $\psi(\boldsymbol{x})$

$$
I_1(\Omega) = \int_{\Omega} \psi(\boldsymbol{x}) \ \mathrm{d}\,\Omega \tag{13}
$$

at the reference configuration Ω in the direction of $\delta \boldsymbol{v}$ is defined as

$$
\frac{\partial I_1}{\partial \Omega} \delta \boldsymbol{v} = \frac{d}{dt} I_1(\boldsymbol{x} + t\delta \boldsymbol{v}) \Big|_{t=0} = \frac{d}{dt} \int_{\Omega_t} \psi(\boldsymbol{x}_t) \ \mathrm{d}\,\Omega_t \Big|_{t=0}. \tag{14}
$$

Transforming this integral into the reference configuration yields

$$
\frac{\partial I_1}{\partial \Omega} \delta \boldsymbol{v} = \frac{d}{dt} \int_{\Omega} \psi(\boldsymbol{x}_t) j(\boldsymbol{x}_t) \ \mathrm{d}\,\Omega \Big|_{t=0} = \int_{\Omega} \left(\frac{d}{dt} \psi(\boldsymbol{x}_t) j(\boldsymbol{x}_t) + \psi(\boldsymbol{x}_t) \frac{d}{dt} j(\boldsymbol{x}_t) \right) \ \mathrm{d}\,\Omega \Big|_{t=0} \tag{15}
$$

with the determinant of the mapping

$$
j(\boldsymbol{x}_t) = \det \frac{\partial \boldsymbol{x}_t}{\partial \boldsymbol{x}},
$$

which has, according to, e.g., [33,34], the derivative

$$
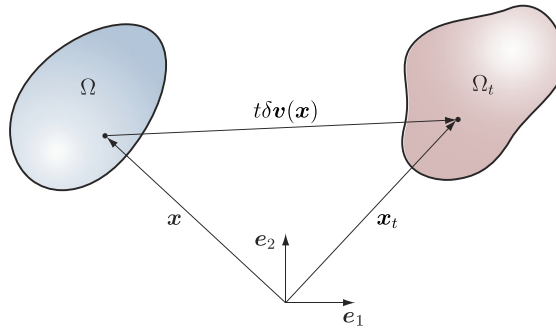\frac{d}{dt} j(\boldsymbol{x}_t) \Big|_{t=0} = \nabla \cdot \delta \boldsymbol{v}. \tag{16}
$$

Fig. 2. Reference and the perturbed domains (left and right, respectively).

After introducing (16) into (15) and applying the divergence theorem we obtain

$$\frac{\partial I_1}{\partial \Omega} \delta v = \int_{\Omega} (\nabla \psi(\boldsymbol{x}) \cdot \delta v + \psi(\boldsymbol{x}) \nabla \cdot \delta v) \, \mathrm{d}\Omega = \int_{\Gamma} \psi(\boldsymbol{x})(\delta v \cdot \boldsymbol{n}) \, \mathrm{d}\Gamma, \tag{17}$$

where $\boldsymbol{n}$ is the unit normal to the boundary. Notice that this integral is zero when the perturbation direction δv is chosen tangential to the boundary. Perturbations tangential to the boundary do not lead to a change in I_1.

Although not used in this work, we give for completeness the variation of the boundary integral of the scalar function $\psi(\boldsymbol{x})$,

$$I_2(\Gamma) = \int_{\Gamma} \psi(\boldsymbol{x}) \, \mathrm{d}\Gamma. \tag{18}$$

The variation of this integral at the reference configuration Ω in the direction of δv reads

$$\frac{\partial I_2}{\partial \Gamma} \delta v = \int_{\Gamma} (\nabla \psi(\boldsymbol{x}) \cdot \boldsymbol{n} + H(\boldsymbol{x})\psi(\boldsymbol{x})) \, (\delta v \cdot \boldsymbol{n}) \, \mathrm{d}\Gamma, \tag{19}$$

where $H(\boldsymbol{x})$ is the mean curvature on Γ, see [24,33].

We can now write the variation of the Lagrangian $L(\Omega, \boldsymbol{u}, \boldsymbol{\lambda})$ in the direction of δv using the results (17) and (19). For practical shape optimisation problems in solid mechanics we usually have only boundary variations of the form

$$\begin{aligned} \delta \hat{v} &= \boldsymbol{0} \quad \text{on } \Gamma_D, \\ \delta \hat{v} &= \boldsymbol{0} \quad \text{on } \Gamma_N \text{ with } \boldsymbol{\sigma n} = \bar{\boldsymbol{t}}, \\ \delta \hat{v} &\neq \boldsymbol{0} \quad \text{on } \Gamma_N \text{ with } \boldsymbol{\sigma n} = \boldsymbol{0}. \end{aligned} \tag{20}$$

This means that only parts of the Neumann boundary Γ_N with no traction are free to move during shape optimisation. With the result (17) at hand the variation of the Lagrangian (6) in the direction $\delta \boldsymbol{\mathcal{V}}$ reads

$$\frac{\partial L}{\partial \Omega} \delta \hat{v} = \frac{\partial J(\Omega, \boldsymbol{u})}{\partial \Omega} \delta \hat{v} + \int_{\Gamma_N} (\nabla \boldsymbol{\lambda} : \boldsymbol{\sigma}(\boldsymbol{u}) - \boldsymbol{\lambda} \cdot \boldsymbol{f})(\delta \hat{v} \cdot \boldsymbol{n}) \, \mathrm{d}\Gamma. \tag{21}$$

For structural compliance (5) as the cost function (i.e., $\boldsymbol{\lambda} = -\boldsymbol{u}$) we obtain

$$\frac{\partial L}{\partial \Omega} \delta \hat{v} = \int_{\Gamma_N} (2\boldsymbol{u} \cdot \boldsymbol{f} - \nabla \boldsymbol{u} : \boldsymbol{\sigma}(\boldsymbol{u})) \, (\delta \hat{v} \cdot \boldsymbol{n}) \, \mathrm{d}\Gamma = \int_{\Gamma_N} g(\boldsymbol{u})(\delta \hat{v} \cdot \boldsymbol{n}) \, \mathrm{d}\Gamma, \tag{22}$$

where $g(\boldsymbol{u})$ is the shape kernel function. It is worth emphasising that without restricting $\delta \hat{v}$ as stated in (20), the shape derivative would contain several more terms. Moreover, for cost functions other than the structural compliance, the kernel function usually is also dependent on the adjoint solution $\boldsymbol{\lambda}$.

During the iterative shape optimisation the shape kernel function $g(\boldsymbol{u})$ is used as gradient information. In order to achieve a maximum decrease in the cost function the boundary perturbation is chosen proportional to

$$\delta \hat{v} = -g(\boldsymbol{u})\boldsymbol{n} \tag{23}$$

such that

$$\frac{\partial L}{\partial \Omega} \delta \hat{\boldsymbol{v}} = - \int_{\Gamma_N} g(\boldsymbol{u})^2 \, \mathrm{d}\, \Gamma. \tag{24}$$

3. Immersed finite element discretisation

The shape derivatives introduced in the previous section depend on the solution of the primal and adjoint problems (1) and (10), respectively. Although for compliance optimisation the primal and adjoint solutions are identical (up to sign), this is not generally the case for other cost functions. During the iterative shape optimisation, both boundary value problems have to be repeatedly solved on constantly evolving domains. In a conventional finite element setting this requires frequent mesh smoothing or updating. Therefore, immersed, or embedded, grid finite element approaches that do not require remeshing have clear advantages in shape optimisation [25,30,35]. In the present work, we use an immersed finite element technique that we previously developed in the context of incompressible fluid–structure interaction [29,36]. The key features of which are: (i) the weak enforcement of Dirichlet boundary conditions with the Nitsche technique, (ii) the use of isoparametric b-spline basis functions for discretisation, and (iii) the numerically robust boundary and cut-cell treatment. In the following we provide a brief summary of our discretisation method. Although we only discuss the discretisation of the primal problem (1), the same derivations also apply to the adjoint problem (10).

3.1. Weak form of the equilibrium equations

For the linear elastic solid introduced in Section 2.1, the weak form of the equilibrium equations (1) reads

$$\int_{\Omega} \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\epsilon}(\delta \boldsymbol{u}) \, \mathrm{d}\, \Omega = \int_{\Omega} \boldsymbol{f} \cdot \delta \boldsymbol{u} \, \mathrm{d}\, \Omega + \int_{\Gamma_N} \bar{\boldsymbol{t}} \cdot \delta \boldsymbol{u} \, \mathrm{d}\, \Gamma + \int_{\Gamma_D} \boldsymbol{t}(\boldsymbol{u}) \cdot \delta \boldsymbol{u} \, \mathrm{d}\, \Gamma, \tag{25}$$

where $\delta \boldsymbol{u}$ are test functions, which are here assumed not to be zero on the Dirichlet boundary. This assumption is necessary because we use non-boundary-fitting meshes. The weak form (25) is not coercive and would lead, for instance, to a singular system of equations when discretised. In order to render (25) coercive we use the consistent penalty method proposed by Nitsche [37], which leads to

$$\int_{\Omega} \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\epsilon}(\delta \boldsymbol{u}) \, \mathrm{d}\, \Omega - \int_{\Gamma_D} \boldsymbol{t}(\boldsymbol{u}) \cdot \delta \boldsymbol{u} \, \mathrm{d}\, \Gamma - \int_{\Gamma_D} \boldsymbol{u} \cdot \boldsymbol{t}(\delta \boldsymbol{u}) \, \mathrm{d}\, \Gamma + \frac{\gamma}{h} \int_{\Gamma_D} \boldsymbol{u} \cdot \delta \boldsymbol{u} \, \mathrm{d}\, \Gamma$$
$$= \int_{\Omega} \boldsymbol{f} \cdot \delta \boldsymbol{u} \, \mathrm{d}\, \Omega + \int_{\Gamma_N} \bar{\boldsymbol{t}} \cdot \delta \boldsymbol{u} \, \mathrm{d}\, \Gamma \tag{26}$$

with the penalty parameter $\gamma > 0$ and the characteristic finite element size h. In contrast to conventional penalty methods, the parameter γ in the Nitsche method is only required for numerical stability and typically a small value is sufficient.

3.2. Finite element discretisation

We use a logically Cartesian grid and the associated tensor-product b-spline basis functions for discretising the weak form (26). The grid has to have the connectivity of a Cartesian grid but the cell sizes need not to be uniform. Fig. 3 shows a typical setup in two space dimensions. The Cartesian grid facilitates the use of tensor-product b-spline basis functions, which have a number of appealing properties known from isogeometric analysis. Specifically, in shape optimisation the smoothness of higher-order b-splines leads to a shape kernel function (22), which is continuous across element boundaries. This leads to optimisation algorithms that are more robust than the ones based on C^0-continuous shape functions and discontinuous shape gradients.

According to the isoparametric concept, we approximate the domain geometry and the solution with tensor-product b-splines.

$$\boldsymbol{x} \approx \boldsymbol{x}_h(\boldsymbol{\xi}) = \sum_i B_i^\alpha(\boldsymbol{\xi}) \boldsymbol{x}_i, \qquad \boldsymbol{u} \approx \boldsymbol{u}_h(\boldsymbol{\xi}) = \sum_i B_i^\alpha(\boldsymbol{\xi}) \boldsymbol{u}_i, \tag{27}$$
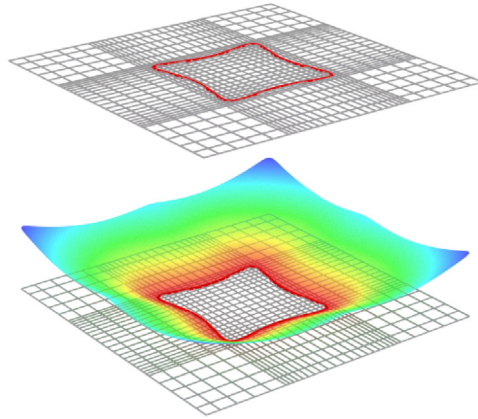
Fig. 3. Domain discretisation with the immersed finite element method. The top figure shows a (logically) Cartesian grid and a spline curve describing the domain boundary. On the Cartesian grid the boundary is represented using the signed distance function shown below. The zero isocontour of the distance function provides an approximation to the original spline curve.

where $\boldsymbol{\xi}$ are the parametric coordinates in the knot space and $\boldsymbol{i}$ is the control point index with the corresponding b-spline $B_{\boldsymbol{i}}^{\alpha}$ of degree α. In optimisation computations, we usually use quadratic b-splines with $\alpha = 2$, which are C^1-continuous and lead to continuous shape gradients. The control point coefficients $\boldsymbol{x_i}$ and $\boldsymbol{u_i}$ are the nodal coordinates and displacements, respectively. The discrete finite element equations are obtained by introducing the approximation equations (27) into the weak form (26) and subsequent element-by-element numerical integration. The elements which are only partially covered by the solid, so-called cut-cells, are first triangulated prior to integration, see [36]. The ill-conditioning of the system matrix due to small cut-cells is avoided with an extension approach originally proposed in [38]. Finally, note that the shape gradient in the cut-cells is computed by simply evaluating (22). The advantage of the analytic adjoint formulation here is that the derivatives of the cut-cells with respect to the boundary position are not needed.

On the logically Cartesian grid we represent the domain boundaries implicitly with a signed distance function (or, in other terms, a level set) [27,29,36]. This is done despite the fact that we have a parametric representation of the domain in the form of a multiresolution subdivision surface, see Section 4. The aim of the switch from a parametric to an implicit representation is to eliminate pathological geometries and topologies, like multiple crossings of a cell edge by the boundary. For optimisation problems with large boundary deformations and topology changes the low-pass filtering of the geometry provided through the switch to an implicit representation makes the finite element computations more robust.

4. Multiresolution optimisation

On the logically Cartesian discretisation grid we represent the domain boundaries either with subdivision curves or subdivision surfaces, depending on the dimensionality of the problem. The inherent hierarchy of subdivision schemes lends itself to multiresolution representation and editing of curves and surfaces. The specific subdivision scheme that we use yields in the curve case cubic b-splines and in the surface case cubic tensor-product b-splines [20]. Subdivision surfaces yield smooth surfaces even for unstructured surface meshes with non-tensor product structure. We refer to the monograph [16] as an introduction to subdivision surfaces and multiresolution editing in geometric modelling and animation.

4.1. Subdivision scheme for one-dimensional cubic b-splines

The refinability property of cubic b-splines can be utilised to derive a corresponding subdivision scheme. To illustrate this, we consider the coarse knot sequence $\xi_i = 0, 1, 2, 3, \ldots$ and the corresponding fine knot sequence $\tilde{\xi}_i = 0, 0.5, 1, 1.5, 2, 2.5, 3, \ldots$. We denote the b-splines on the coarse knot sequence with $B_i(\xi)$ and the ones on the fine knot sequence with $\tilde{B}_i(\xi)$. According to the b-spline refinability equation, see, e.g., [16,39], it is possible to
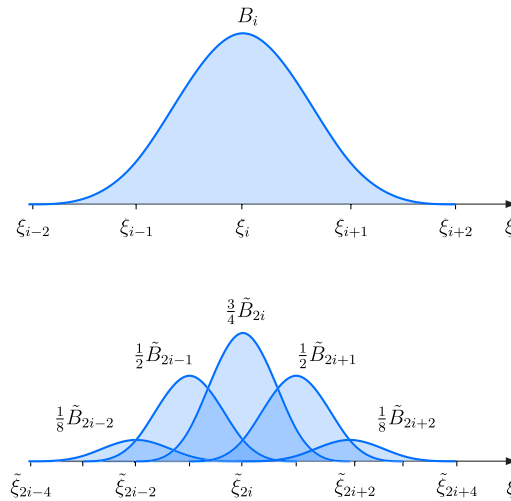
Fig. 4. Refinement relation for cubic b-splines.

represent the coarse b-splines as a linear combination of the fine b-splines

$$B_i(\xi) = \sum_j S_{ij} \tilde{B}_j(\xi), \tag{28}$$

where S_{ij} is the subdivision matrix with the components

$$S_{ij} = \begin{bmatrix} \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \frac{1}{2} & \frac{1}{8} & 0 & 0 & 0 & 0 & 0 & \cdots \\ \cdots & \frac{1}{2} & \frac{3}{4} & \frac{1}{2} & \frac{1}{8} & 0 & 0 & 0 & \cdots \\ \cdots & 0 & \frac{1}{8} & \frac{1}{2} & \frac{3}{4} & \frac{1}{2} & \frac{1}{8} & 0 & \cdots \\ \cdots & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{2} & \frac{3}{4} & \frac{1}{2} & \cdots \\ \cdots & 0 & 0 & 0 & 0 & 0 & \frac{1}{8} & \frac{1}{2} & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \end{bmatrix}. \tag{29}$$

Each row of this banded matrix has the same five non-zero components, always shifted by two columns relative to adjacent rows. As shown in Fig. 4 each row expresses how a coarse b-spline can be obtained as the weighted sum of fine b-splines. It is evident that the exact structure and components of the matrix (29) depend on the degree of the considered b-splines. The components of the subdivision matrix for different polynomial degrees can be found, e.g., in [29].

Next, we consider a spline curve defined in terms of the coarse b-splines and the corresponding control vertices with the coordinates x_i, i.e.,

$$x_h(\xi) = \sum_i B_i(\xi) x_i. \tag{30}$$

The control polygon of the spline is obtained by linearly connecting the control points x_i. Because of the refinability property (28), the spline curve (30) can also be represented with the fine b-splines $\tilde{B}_j(\xi)$. To show this, we introduce
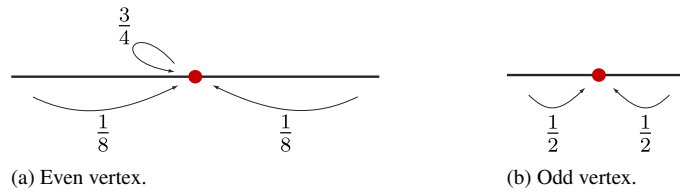
(a) Even vertex.   (b) Odd vertex.

Fig. 5. Subdivision stencils for cubic b-splines.

the refinement relation (28) into (30)

$$\boldsymbol{x}_h(\xi) = \sum_i \sum_j S_{ij} \tilde{B}_j(\xi)\boldsymbol{x}_i = \sum_j \tilde{B}_j(\xi) \left(\sum_i S_{ij}\boldsymbol{x}_i \right). \tag{31}$$

Hence, choosing the fine control vertex coordinates with

$$\tilde{\boldsymbol{x}}_j = \sum_i S_{ij}\boldsymbol{x}_i \tag{32}$$

ensures that both the coarse and fine b-splines represent the same identical spline curve. Further inspection of (32) and subdivision matrix (29) reveals, that the fine control vertices are the weighted averages of coarse control vertices, with the *columns* of the subdivision matrix representing the weights. There are two different sets of weights corresponding to the two different types of columns of the subdivision matrix. Based on the numbering scheme implied in Fig. 4 and the structure of the subdivision matrix, it is easy to deduce that one set of weights applies to control vertices with even indices

$$\tilde{\boldsymbol{x}}_{2i} = \frac{1}{8}\boldsymbol{x}_{i-1} + \frac{3}{4}\boldsymbol{x}_i + \frac{1}{8}\boldsymbol{x}_{i+2} \tag{33}$$

and the other set of weights applies to control vertices with odd indices

$$\tilde{\boldsymbol{x}}_{2i+1} = \frac{1}{2}\boldsymbol{x}_i + \frac{1}{2}\boldsymbol{x}_{i+1}. \tag{34}$$

Hence, in terms of computer implementation, for a given coarse polygon a corresponding fine polygon is obtained by first splitting each edge into two edges and subsequently computing control point coordinates according to (33) and (34). In computer graphics literature the weights in (33) and (34) are usually given in the form of stencils shown in Fig. 5.

In subdivision schemes the foregoing described approach for obtaining a refined control polygon is applied successively leading to finer and finer polygons. From the properties of the b-splines underlying (33) and (34), it is clear that the control points converge to a cubic b-spline.

4.2. Catmull–Clark subdivision surfaces

Two-dimensional b-splines can be generated as the tensor products of two one-dimensional b-splines. Similarly, the tensor product of two one-dimensional subdivision stencils yields the two-dimensional subdivision stencils shown in Fig. 6. The three different stencils correspond to the three different type of vertices which occur during the splitting of each face into four faces. It can be easily verified that the weights given in Fig. 6 are the tensor-products of the one-dimensional weights for cubic b-splines given in Fig. 5. Hence, successively refining a control mesh and computing the control vertex coordinates with the stencils given in Fig. 6 will lead in the limit to a cubic spline surface.

It is evident that the tensor-product stencils only apply to meshes in which each vertex within the domain is connected to four faces. The number of the faces connected to a vertex is referred to as the valence of that vertex and is denoted with v. For the sake of brevity, we refer to [16] for the discussion of regularity of vertices on the boundaries and corners. The domain vertices with a valence other than four are known as extraordinary vertices or star-vertices. As originally proposed by Catmull and Clark [20], the key idea in subdivision surfaces is to apply the modified stencil shown in Fig. 7 at the extraordinary vertices.
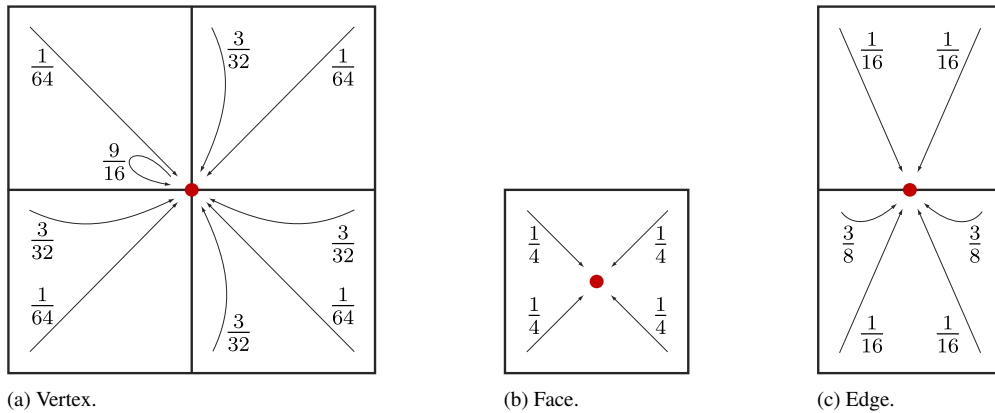
Fig. 6. Subdivision stencils for the Catmull–Clark scheme. Each of the stencils are used for computing the coordinates of vertices of the type indicated by red dot.
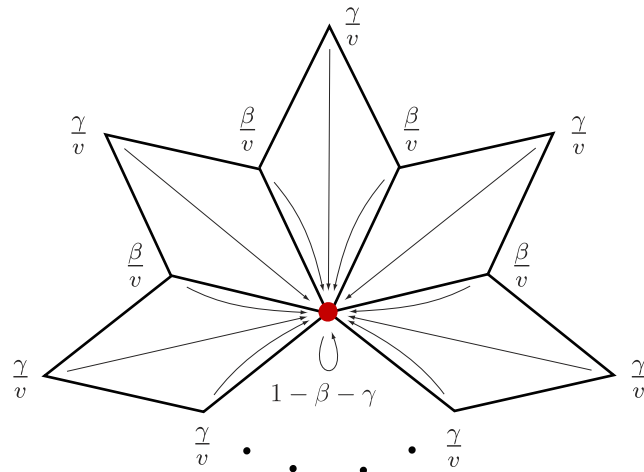

Fig. 7. Subdivision stencil for an irregular vertex with valence v with the weights $\beta = \frac{3}{2v}$ and $\gamma = \frac{1}{4v}$, see [20].

To summarise, in each subdivision refinement step each face of the control mesh is split into four faces and the coordinates of the control vertices are computed with the weights given in Figs. 6 and 7 depending on the local connectivity structure of the vertex. There is mathematical theory which shows that the resulting surface is C^2 continuous almost everywhere except at the extraordinary vertices where it is only C^1 continuous [17].

In addition to the stencils shown in Figs. 6 and 7 there are also extended subdivision stencils for vertices on edges, creases and corners, see, e.g., [40]. In this context, a crease is a line on the surface across which the surface is only C^0 continuous. As an illustrative example, Fig. 8 shows the subdivision refinement of a control mesh for a T-junction geometry with extraordinary vertices and prescribed crease edges.

For later reference, we write the subdivision process as a linear mapping that maps a coarse control mesh at level ℓ to a finer control mesh at level $\ell + 1$

$$x^{\ell+1} = Sx^{\ell}, \tag{35}$$

where $x^{\ell+1}$ and x^{ℓ} are two matrices containing the coordinates of all the vertices at level ℓ and $\ell + 1$, respectively. By definition the initial coarse control mesh has the level $\ell = 0$. The number of columns of $x^{\ell+1}$ and x^{ℓ} is equal to the space dimension and their number of rows is equal to the number of all vertices in the mesh. The subdivision matrix S contains the weights given by the subdivision stencils and its dimension depends on the subdivision level ℓ considered. Lastly, according to (35) the subdivision process can be interpreted as the chain of linear mappings for

(a) Coarse control mesh ($x^{\ell=0}$).   (b) Once subdivided control mesh ($x^{\ell=1}$).   (c) Limit surface ($x^{\ell=\infty}$).
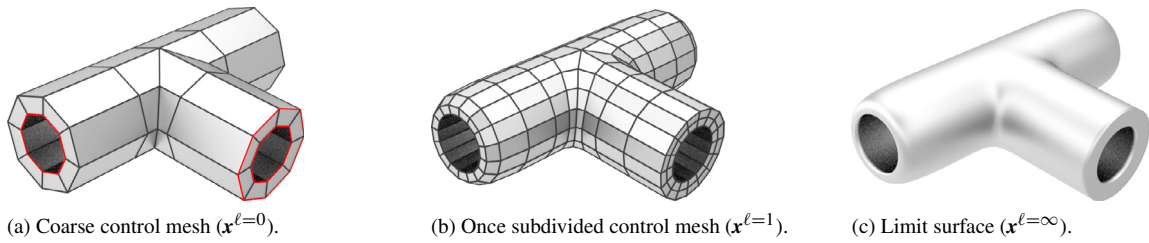
Fig. 8. Subdivision refinement of a T-junction geometry. On the coarse control mesh the edges in red are tagged as crease edges. The control mesh at the centre is obtained after one step of subdivision refinement. The geometry on the right is a rendering of the limit surface. Notice that the limit surface is not smooth on the crease edges. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

obtaining increasingly finer control meshes, i.e.,

$$x^0 \xrightarrow{\ S\ } x^1 \xrightarrow{\ S\ } x^2 \xrightarrow{\ S\ } \cdots \xrightarrow{\ S\ } x^{\ell-1} \xrightarrow{\ S\ } x^\ell. \tag{36}$$

4.3. Multiresolution surface editing

The sequence of control meshes generated during subdivision refinement readily lends itself for multiresolution editing of geometries [21,22,41,42]. As discussed, Catmull–Clark subdivision surfaces are based on cubic b-splines. Hence, the support size of each subdivision basis function consists of a two-ring of faces, cf. Fig. 4. With increasing refinement level the physical support size of basis functions becomes smaller. Accordingly, depending on the refinement level the editing of control vertex positions leads to changes with different spatial extent on the limit surface. As an illustrative example the T-junction geometry introduced earlier is considered in Fig. 9. In the middle column the coordinates of selected vertices at the levels $\ell = 0$, $\ell = 1$ and $\ell = 2$ are modified. As can be seen, in the last column of pictures this leads to changes in the limit surface in the vicinity of the edited vertices and the spatial extent of the changes is correlated with the refinement level.

The subdivision surfaces by itself do not provide the possibility to simultaneously edit coarse and fine control meshes. For instance, after a fine control mesh is edited it is not possible anymore to edit a coarser level in order to apply large scale changes to the geometry. Simultaneous editing of different levels can be achieved with a wavelet-like multiresolution decomposition of the control meshes, as will be discussed further below.

Before considering the multiresolution decomposition of control meshes, we introduce the coarsening of control meshes that were obtained with subdivision. The linear coarsening matrix $\boldsymbol{R}$ maps the given control points at level $\ell + 1$ to the control points at the coarser level ℓ,

$$x^\ell = \boldsymbol{R} x^{\ell+1}. \tag{37}$$

The coarsening matrix $\boldsymbol{R}$ is not unique and different choices are possible. Essentially, the control mesh at level $\ell + 1$ has more control vertices than the one at level ℓ and may contain more geometric information. In our implementation we obtain the coarsening matrix $\boldsymbol{R}$ from a least squares fit of the subdivided coarse control vertices $\boldsymbol{S} x^\ell$ to the fine control vertices $x^{\ell+1}$, i.e.,

$$x^\ell = \underset{y^\ell}{\operatorname{argmin}} \|x^{\ell+1} - \boldsymbol{S} y^\ell\|^2, \tag{38}$$

which leads to

$$x^\ell = \boldsymbol{R} x^{\ell+1} \quad \text{with } \boldsymbol{R} = (\boldsymbol{S}^\mathsf{T} \boldsymbol{S})^{-1} \boldsymbol{S}^\mathsf{T}. \tag{39}$$

By comparing with (35) we can identify $\boldsymbol{R}$ as the pseudo-inverse of the subdivision matrix $\boldsymbol{S}$ so that

$$x^\ell = \boldsymbol{R} \left(\boldsymbol{S} x^\ell \right) = x^\ell. \tag{40}$$

In words, subdivision refinement of a control mesh (without editing) followed by coarsening yields the original control mesh. Instead of least squares fitting, the coarsening matrix $\boldsymbol{R}$ can also be defined based on quasi-interpolation
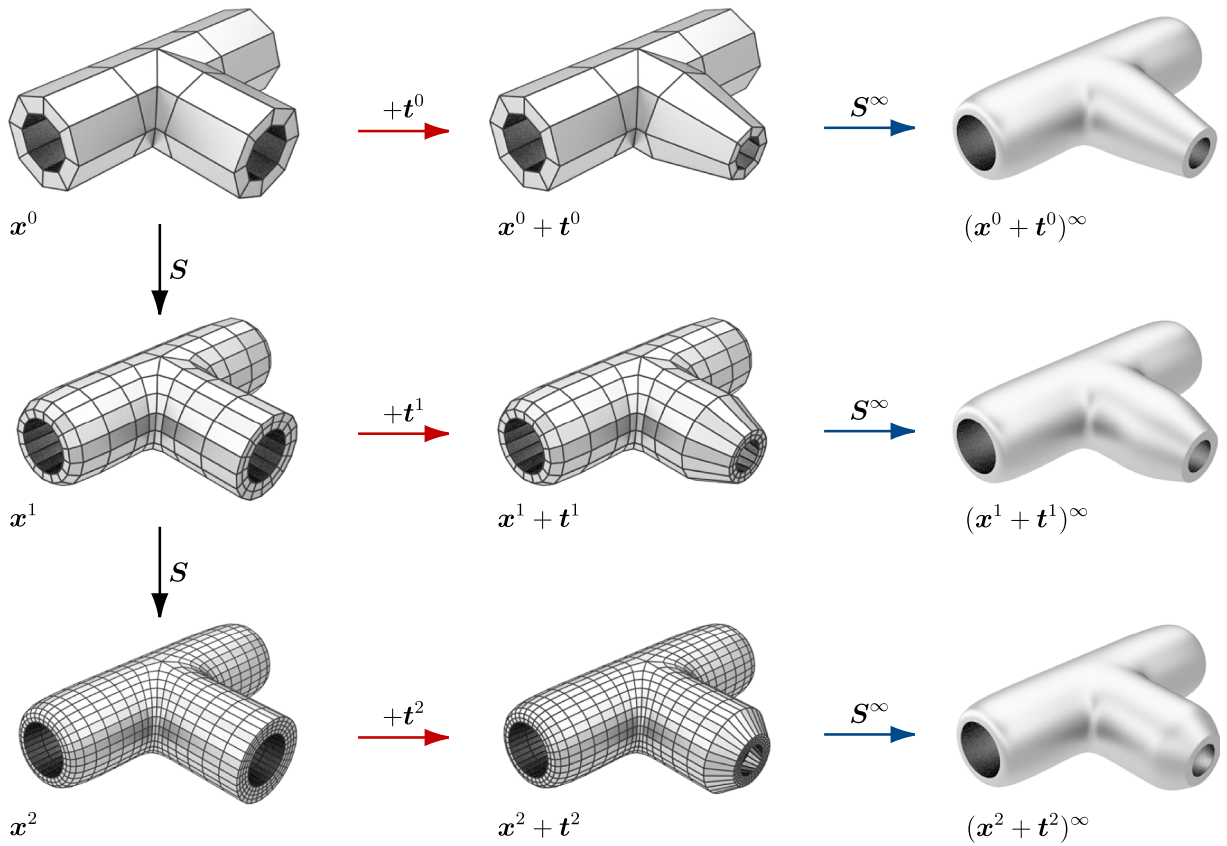
Fig. 9. Multiresolution editing of the T-junction geometry introduced in Fig. 8. The aim is to modify the shape and diameter of the protruding tube. To achieve this the coordinates of selected vertices at either level $\ell = 0$, 1 or 2 are modified as shown on the middle column, where the vectors t^0, t^1 and t^2 represent the modifications at the respective level. Notice the effect of the modification level on the limit surface (last column).

[43,44] or smoothing [21]. On the other hand, coarsening by simply subsampling of the fine control mesh usually leads to artefacts in the form of oscillations in the coarse control mesh. The proposed least squares fit approach is not very common in computer graphics because of the need for interactivity and fast processing times. Although the least squares matrix in (39) is sparse its solution cannot be found at interactive rates. Notice also that (39) represents a system of equations with d right hand side vectors for each of the d coordinate directions.

Similar to subdivision refinement the coarsening matrix can be successively applied in order to obtain coarser representations of the geometry, i.e.,

$$x^0 \xleftarrow{\;R\;} x^1 \xleftarrow{\;R\;} x^2 \xleftarrow{\;R\;} \cdots \xleftarrow{\;R\;} x^{\ell-1} \xleftarrow{\;R\;} x^{\ell}. \tag{41}$$

The dimension of the matrix R depends on the considered level ℓ. As mentioned during the coarsening process each of the coordinate directions are considered individually. Fig. 10 shows the coarsening of a subdivision surface with the described approach. From the shown limit surfaces it is visible that the coarsening process leads to a smoothing of the geometry; and the overall geometry is faithfully represented by the coarser representations. For this reason, the coarsening process is sometimes also referred to as a smoothing process.

With the introduced subdivision and coarsening operations it is now possible to devise a wavelet-like multiresolution decomposition of a control mesh and the associated geometry. The aim of this decomposition is to enable the simultaneous editing of different levels of the subdivision surface. This is achieved by storing a coarse control mesh and the differences between successive control meshes called *details*. Subsequently it is possible to first edit any control mesh level and then to add the stored details as necessary. For a given fine control mesh at level ℓ the

(a) Fine control mesh ($x^{\ell=3}$).   (b) Twice coarsened control mesh ($x^{\ell=1}$).   (c) Coarse control mesh ($x^{\ell=0}$).
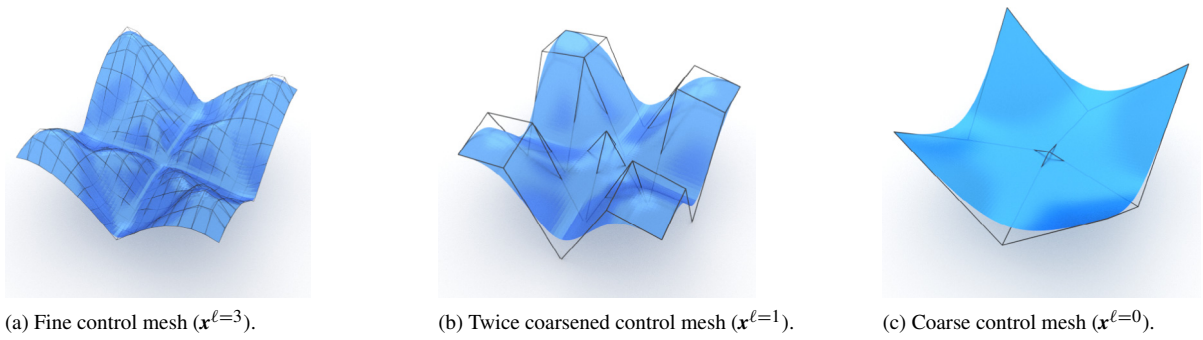
Fig. 10. Subdivision coarsening of a fine control mesh. Successive application of the coarsening matrix R leads to increasingly coarser control meshes.

detail vector $d^{\ell-1}$ is computed using the subdivision and coarsening processes as follows

$$d^{\ell-1} = x^\ell - S\left(Rx^\ell\right) = (I - SR)x^\ell. \tag{42}$$

In turn, when a control mesh at level $\ell - 1$ and the detail $d^{\ell-1}$ are given the geometry at level ℓ can be recovered according to (42) and (39) with

$$x^\ell = Sx^{\ell-1} + d^{\ell-1}. \tag{43}$$

The global detail vector $d^{\ell-1}$ is composed of local vertex detail vectors, which can be conveniently stored at the vertices. In our actual implementation, we express each local detail vector in a local tangential coordinate system at its vertex. As known in computer graphics, this is necessary so that in (43) any modifications to the geometry $x^{\ell-1}$ should have an intuitive effect on the details contained in $d^{\ell-1}$, see, e.g., [21,41,45].

Finally, the two-level decomposition given by (42) and (43) can be successively applied leading to a wavelet-like multiresolution decomposition of the surface. The process of obtaining the details for a given fine geometry is referred to as the *analysis step*

$$
\begin{array}{c}
x^0 \xleftarrow{R} x^1 \xleftarrow{R} x^2 \xleftarrow{R} \cdots \xleftarrow{R} x^{\ell-1} \xleftarrow{R} x^\ell \\
\swarrow_{I-SR} \quad \swarrow_{I-SR} \quad \swarrow_{I-SR} \quad \swarrow_{I-SR} \quad \swarrow_{I-SR} \\
d^0 \qquad d^1 \qquad \cdots \qquad d^{\ell-2} \qquad d^{\ell-1}
\end{array}
\tag{44}
$$

The corresponding *synthesis step* takes the form

$$
\begin{array}{c}
x^0 \xrightarrow{S} x^1 \xrightarrow{S} x^2 \xrightarrow{S} \cdots \xrightarrow{S} x^{\ell-1} \xrightarrow{S} x^\ell \\
\nearrow \qquad \nearrow \qquad \nearrow \qquad \nearrow \qquad \nearrow \\
d^0 \qquad d^1 \qquad \cdots \qquad d^{\ell-2} \qquad d^{\ell-1}
\end{array}
\tag{45}
$$

For an efficient implementation of the analysis and synthesis steps and the related data structures we refer to Zorin et al. [16].

A typical workflow during multiresolution editing of a cylindrical component with few small bumps is shown in Fig. 11. First the geometry is created starting from a coarse control mesh and adding the bumps on the two times subdivided control mesh. This is achieved by displacing few selected vertices on the fine control mesh and yields the leftmost picture Fig. 11. After this step, in order to apply large scale changes it is necessary to employ a multiresolution decomposition of the geometry. More specifically, the control mesh $x^{\ell=2}$ is decomposed into the detail vectors d^0 and d^1, and the original control mesh x^0. For this specific geometry the detail vector d^0 is zero. After this decomposition it is possible to change the original control mesh into, for instance, a cone and subsequently to subdivide and to automatically add the stored details leading to the shown cone geometry with bumps.

4.4. Multiresolution shape optimisation

The introduced subdivision multiresolution editing technique enables the use of two different resolutions of the same geometry for optimisation and analysis. The two resolutions correspond to different refinement levels in a multiresolution hierarchy. In shape optimisation it is usually necessary to use a coarse control mesh for geometry updating and a relatively fine control mesh for analysis. As is known, unwanted geometry oscillations may appear when the analysis and geometry representations have similar resolutions [2,10,11]. These geometry oscillations are usually a numerical artefact or a result from the ill-posedness of the considered optimisation problem. Moreover, in practical applications it might be desirable to optimise only a very coarse representation out of aesthetic or manufacturability reasons.

Algorithm 1 Multiresolution shape optimisation

    // choose maximum optimisation level $\ell_{o,max}$ and computational level ℓ_c
    // read input control mesh $x^{\ell_{inp}}$
1: **if** $\ell_{inp} = 0$ **then**
    // initialise all detail vectors ($d^\ell = 0$)
2: **else if** $\ell_{inp} > 0$ **then**
3:     **for** $\ell \leftarrow \ell_{inp}$ to 0 **do**
4:       $d^\ell = x^\ell - S(Rx^\ell)$
    // initialise optimisation level
5: $\ell_o = 0$
    // initialise cost function
6: $J = \infty$
    // iterate over optimisation levels
7: **while** $\ell_o \leq \ell_{o,max}$ **do**
    // update vertex coordinates x^{ℓ_o} while the cost function decreases
8:    **repeat**
    // subdivide optimisation level ℓ_o up to analysis level ℓ_c
9:     **for** $\ell \leftarrow \ell_o$ to ℓ_c **do**
10:      $x^\ell \leftarrow Sx^\ell + d^\ell$
    // compute cost function $J = J(x^{\ell_c}, u(x^{\ell_c}))$ and shape kernel field $g(x) = g(u(x^{\ell_c}))$
    // compute maximum ascent direction at the vertices x^{ℓ_c} with the outer normals $n(x^{\ell_c})$
11:     $g^{\ell_c} = g(x^{\ell_c})n(x^{\ell_c})$
    // project shape derivative to optimisation level
12:     **for** $\ell \leftarrow \ell_c$ to ℓ_o **do**
13:      $g^\ell \leftarrow Rg^\ell$
    // update vertex coordinates of the optimisation level
14:     $x^{\ell_o} \leftarrow (x^{\ell_o} - \alpha g^{\ell_o})$ with $\alpha \geq 0$
15:    **until** $(J_{\text{previous}} - J) <$ tolerance
    // increment optimisation level
16:    $\ell_o \leftarrow (\ell_o + 1)$
17:    $x^{\ell_o} \leftarrow Sx^{\ell_o} + d^{\ell_o}$

Algorithm 1 describes the proposed multiresolution shape optimisation approach. The fixed computational level ℓ_c and the maximum optimisation level $\ell_{o,max}$ are prescribed by the user. The level ℓ_c has to be large enough such that the numerical solution is accurate enough for practical purposes. The maximum optimisation level has to be chosen $\ell_{o,max} \leq \ell_c$ and determines the smallest geometric feature size on the optimised geometry. The input control mesh $x^{\ell_{inp}}$ can be a coarse mesh with $\ell_{inp} = 0$ or an already edited fine multiresolution mesh with $\ell_{inp} > 0$. For control meshes with $\ell_{inp} > 0$ first a multiresolution decomposition as indicated in (44) is performed. Throughout the algorithm the optimisation control mesh and its level are denoted with x^{ℓ_o} and ℓ_o, respectively. For the immersed finite element analysis the geometry corresponding to a control mesh x^{ℓ_c} with $\ell_c \geq \ell_o$ is used. The analysis level ℓ_c is usually fixed and the control mesh has elements of similar size like the cells of the immersed finite element grid. In order
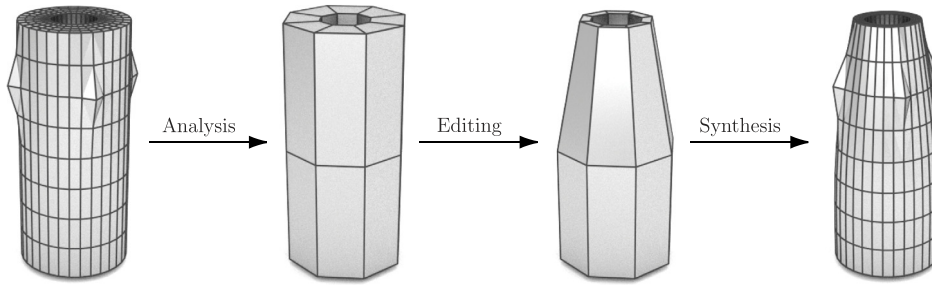
Fig. 11. Multiresolution editing of a cylindrical component. The fine resolution mesh shown on the left at level $\ell = 2$ is given. In the analysis step the geometry is decomposed into the coarse level $\ell = 0$ (second from the left) and details $\boldsymbol{d}^0$ and $\boldsymbol{d}^1$, cf. (44). As shown on the third from the left, the coarse level can be edited irrespective of the details. In the subsequent synthesis step (45) adding the precomputed details to the coarse level yields the control mesh shown on the right.

to obtain the analysis control mesh $\boldsymbol{x}^{\ell_c}$ from the optimisation control mesh $\boldsymbol{x}^{\ell_o}$ we use the introduced multiresolution refinement technique. The immersed finite element analysis yields the cost function $J(\boldsymbol{x}^{\ell_c}, \boldsymbol{u}(\boldsymbol{x}^{\ell_c}))$ and the shape kernel $g(\boldsymbol{x}^{\ell_c})$, see (22). To compute the shape gradient at the vertices, the surface normal vector $\boldsymbol{n}(\boldsymbol{x}^{\ell_c})$ is required, which is easily computed with the known subdivision stencils for tangent vectors, see e.g. [16]. Vertex-wise multiplication of the shape kernel with the normal vector yields the shape gradient vector $\boldsymbol{g}^{\ell_c}$, which is subsequently projected to the optimisation level vector $\boldsymbol{g}^{\ell_o}$ by successive coarsening with $\boldsymbol{R}$. For the sake of brevity, in Algorithm 1 the geometry is updated with a steepest descent technique and no additional constraints are present. In applications it is common to have additional constraints, such as perimeter, area or volume constraints. In our actual implementation we optimise the constrained discrete problem with the method of moving asymptotes (MMA) proposed by Svanberg [46,47] using the implementation in the NLopt library [48]. Finally, note that in Algorithm 1 the optimisation level ℓ_o is not fixed, it is incremented each time a minimum is reached whilst $\ell_o \leq \ell_{o,max}$.

5. Examples

In this section, we present several examples to demonstrate the robustness and versatility of the proposed multiresolution shape optimisation technique. In all the examples the domain is described either by a cubic b-spline curve (in 2D) or a Catmull–Clark subdivision surface (in 3D). The objective of the optimisation is to minimise the structural compliance (5), which is equivalent to maximising the structural stiffness. The corresponding adjoint problem (10) has (up to the sign) the same right hand side as the primal problem (1). Hence, it is sufficient to consider only the primal problem, which is solved with the immersed finite element technique using quadratic b-spline basis functions. The resulting smooth stress field in combination with unique boundary normals at the vertices of the control mesh leads to smooth shape gradients. The optimised boundary curves or surfaces have in general no corners or sharp edges so that there is always a unique normal.

Initially, in Section 5.1 we consider only shape optimisation examples. Subsequently, in Section 5.2 also topology modification is considered. To this end, we make use of the topology derivative, see e.g. [49,50], for introducing new holes inside the domain. In our current implementation the merging or removing of holes is not considered. In all the two-dimensional examples we use a plane strain formulation unless otherwise indicated.

5.1. Shape optimisation

5.1.1. Simply supported plate with a hole

This introductory example aims to highlight the advantages of multiresolution optimisation over classical approaches using only one or two representation levels. The problem consists of a square plate with an edge length $L = 2$ and a circular hole with diameter $D = 1$, see Fig. 12. The plate is loaded with a line load of length 1. Young's modulus and Poisson's ratio of the plate are $E = 100$ and $v = 0.4$, respectively.

During optimisation the shape of the hole is to be modified so that the structural compliance of the plate is minimised. The area of the hole is constrained with

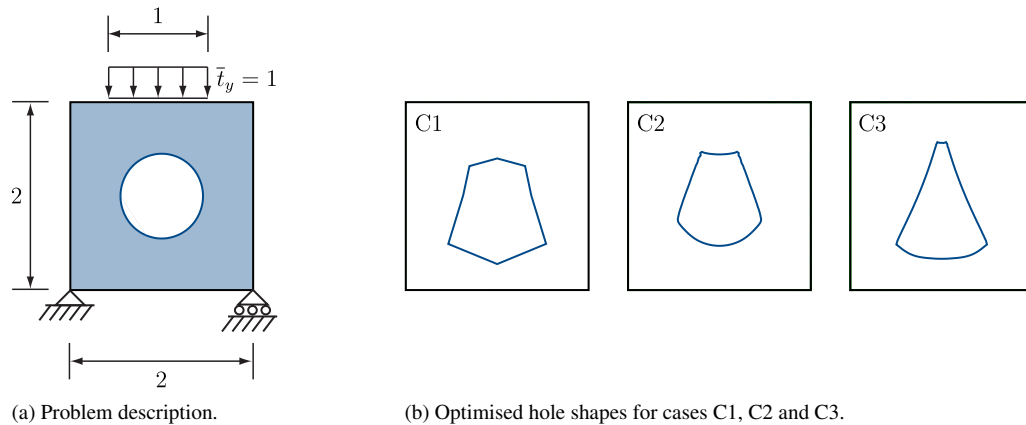$$A(\boldsymbol{x}^{\ell_c}) \geq \frac{\pi}{4}, \tag{46}$$

(a) Problem description.   (b) Optimised hole shapes for cases C1, C2 and C3.

Fig. 12. Simply supported plate with a hole.

where x^{ℓ_c} are the vertex coordinates of the analysis control mesh at level ℓ_c. This constraint is necessary since the stiffest plate is the one with a zero hole diameter. The area of the hole is computed by integrating over its boundary

$$A(x^{\ell_c}) = \frac{1}{2} \int_\Gamma x^{\ell_c} \cdot n(x^{\ell_c}) \, d\,\Gamma, \tag{47}$$

where $n(x^{\ell_c})$ is the normal to the boundary curve Γ. Recall that we represent here the boundary curve with cubic b-splines. In computations, (47) is evaluated using one quadrature point per element of the control polygon. During the optimisation also the gradient of the area constraint is required, which is computed by differentiating the discretised version of (47) with respect to the vertex positions of the analysis control mesh x^{ℓ_c}. Alternatively, it would be possible to use the analytic shape derivative equivalent to (17). The vertex-wise gradient of the area constraint on the analysis level ℓ_c is projected to the optimisation level ℓ_o in the same way as the shape gradient vector g^{ℓ_c}. The detail vectors created during the decomposition (44) are discarded.

Initially, at level $\ell = 0$ the hole is represented with a cubic spline with 8 control points. The immersed finite element grid has 100×100 cells of uniform size. Three cases referred to as C1, C2 and C3 with different geometry and analysis resolutions are studied:

- In C1 only one level with $\ell_o = \ell_c = 0$ is used for analysis and optimisation.
- In C2 a four times subdivided control mesh at refinement level $\ell_o = \ell_c = 4$ is used for analysis and optimisation.
- In C3 the optimisation level starts with $\ell_o = 0$ and increases until $\ell_o = \ell_c = 4$ is reached. Throughout the computations the analysis level is fixed to $\ell_c = 4$.

In case C1 the control mesh that is visible by the immersed finite element grid contains 8 elements and in cases C2 and C3 it contains 128 elements. It is clear that in case C1 the hole geometry is poorly resolved on the immersed finite element grid.

In Fig. 12 the optimised final hole shapes for the three cases are shown. In particular, the difference in optimal shapes for cases C2 and C3 which use the same analysis level $\ell_c = 4$ is striking. The case C1 is different from the other two cases because of the mentioned inadequately coarse analysis control mesh with $\ell_c = 0$. As indicated in Fig. 13, during optimisation only for case C3 the optimisation level ℓ_o is successively increased. The optimisation level is always incremented when a minimum is reached, cf. Algorithm 1. For the three cases the reduction of the relative cost function over the number of iterations is shown in Fig. 14. The case C2 with fixed fine resolution achieves the smallest cost reduction while the case C3 with multiresolution achieves the largest cost reduction. The strong dependence of the optimisation results on geometry parameterisation is well known in structural optimisation and is often associated with the non-convexity of the considered optimisation problem. We conjecture that by initially using a coarse control mesh for optimisation the possible number of local minima is significantly reduced which reduces the possibility of landing in a non-optimal local minimum. It appears that in case C2 the optimisation problem is caught in a local minimum which is significantly higher than the global minimum.
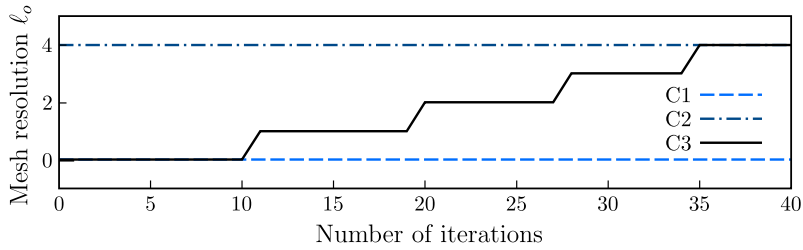
Fig. 13. Simply supported plate with a hole. Variation of the optimisation level ℓ_o over the number of optimisation iterations.
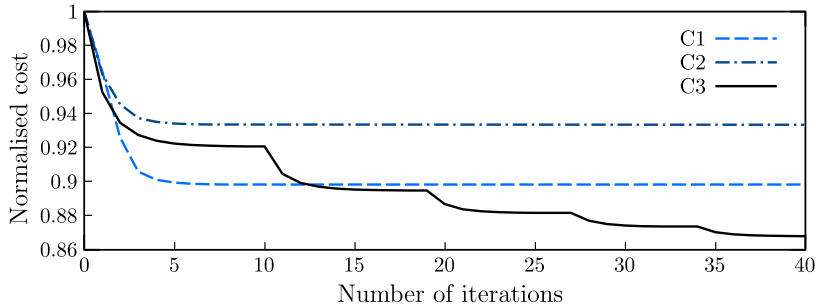

Fig. 14. Simply supported plate with a hole. Reduction of the normalised cost over the number of optimisation iterations. The initial cost for case C1 is 0.073 and for cases C2 and C3 is 0.065.
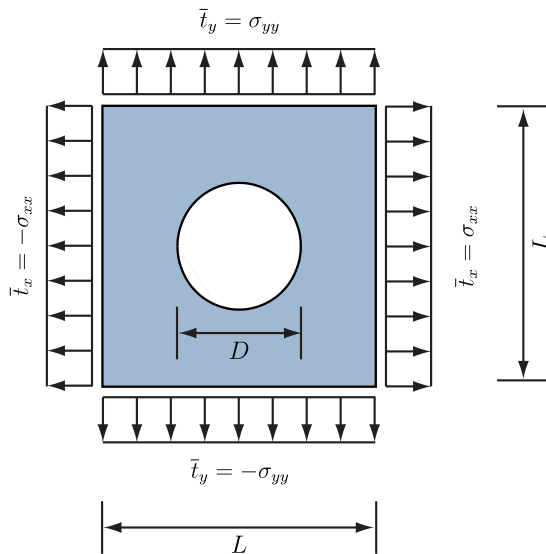

Fig. 15. Optimal hole shapes in a two-dimensional domain. Problem description.

5.1.2. Optimal hole shapes in a two-dimensional domain

In this prototypical example we study the optimal hole shapes in an elastic domain subjected to bi-axial stress. The problem setup is shown in Fig. 15. As in the previous example the area of the hole is constrained to be constant during optimisation. Young's modulus and Poisson's ratio are chosen with $E = 100$ and $\nu = 0.4$, respectively. According to analytical results for infinite plates the optimal hole shape depends on the ratio and sign of the far-field stress [51]. When the two components of the far-field stress are of the same sign the optimal hole shape is an ellipse with an aspect ratio r_x/r_y equal to the bi-axial stress ratio σ_{xx}/σ_{yy}. In contrast, when the two far-field stress components are of opposite sign the optimal hole shape is a quadrilateral with smooth corners. The case with far-field stresses of the same sign has been widely studied in the literature, see, e.g., [30,52,53].
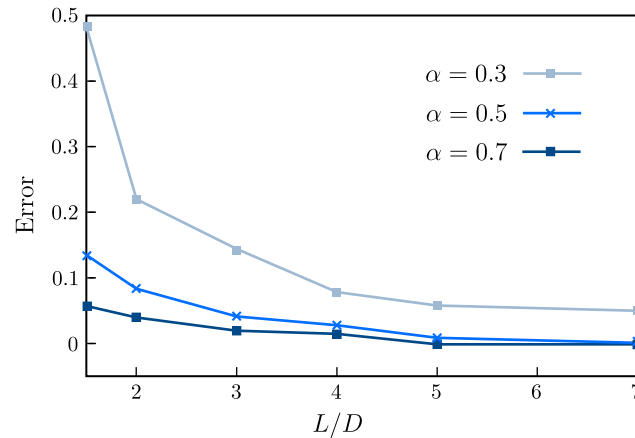
Fig. 16. Optimal hole shapes in a two-dimensional domain. The relative error in the computationally obtained aspect ratio of the ellipse for different stress ratios α and plate sizes L/D, with the hole diameter $D = 1$. The error is defined as the difference between the computationally and analytically obtained aspect ratios.

In our computations the initial hole geometry at level $\ell = 0$ is modelled with a cubic spline with 8 control vertices. The position of the vertices is chosen such that the resulting spline curve represents approximately a circle with diameter $D = 1$. The three times subdivided control mesh with 64 vertices serves as the computational mesh for describing the hole geometry on the immersed finite element grid. The optimisation starts with $\ell_o = 0$ and is incremented until $\ell_o = \ell_c = 3$ is reached, cf. Algorithm 1.

In a first set of computations we quantify the effect of computing with a finite size domain as opposed to an infinite domain underlying the analytical results. To this end, the length to diameter ratio L/D is varied between $1.5 \leq L/D \leq 7$ while the hole diameter is fixed with $D = 1$. In all computations the element size on the immersed finite element grid is kept fixed with $1/25$. We compute the optimised hole shapes for three different bi-axial stress ratios $\alpha = \sigma_{xx}/\sigma_{yy} \in \{0.3, 0.5, 0.7\}$ and seven different length to diameter ratios $L/D \in \{1.5, 2, 3, 4, 5, 7\}$. As mentioned, for bi-axial stress components with the same sign the analytically obtained optimal hole shape is an ellipse with an aspect ratio equal to the stress ratio α. Fig. 16 shows the error in the computationally obtained ellipse aspect ratios for different α and L/D. The error is due to the finite size of the domain and the discretisation errors. As can be seen in Fig. 16 for stress ratios $\alpha \in \{0.5, 0.7\}$ the computationally obtained aspect ratio converges to the analytic result for sufficiently large domains. However, for $\alpha = 0.3$ the obtained aspect ratio does not converge to the analytic result. This is due to the large discretisation errors close to the two apexes of the relatively tall ellipse. This error could be reduced by increasing the computational level ℓ_c and decreasing the cell size of the immersed finite element grid.

In the following set of computations the domain size and the initial hole diameter are chosen with $L = 4$ and $D = 1$. The cell size of the immersed finite element grid is chosen with $1/100$. According to Fig. 16 this setup appears to provide sufficient accuracy while keeping the computation time manageable. The geometric description of the hole remains the same as in the previous set of computations. With the described setup we compute the optimised hole shapes for ten different stress ratios

$$\alpha \in \{-1.0, -0.7, -0.5, -0.3, -0.1, 0.1, 0.3, 0.5, 0.7, 1.0\}.$$

Cherkaev et al. [51] have shown that for negative stress ratios having several holes gives a lower compliance than one single hole. Therefore, it is necessary to prevent the appearance of multiple holes which cannot be obtained with shape optimisation starting with a single hole. To regularise the optimisation problem we add to the compliance cost function (5) an integral penalising perimeter change, i.e.,

$$J(\Omega, \boldsymbol{u}) = \int_{\Omega} \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\epsilon}(\boldsymbol{u}) \, \mathrm{d}\Omega + \rho_L \left(\int_{\Gamma} d\Gamma \right)^2 \tag{48}$$

where ρ_L is a prescribed penalty parameter. The parameter ρ_L controls how much regularisation is applied to penalise the formation of multiple holes. Without this regularisation the solution of the discretised optimisation problem with
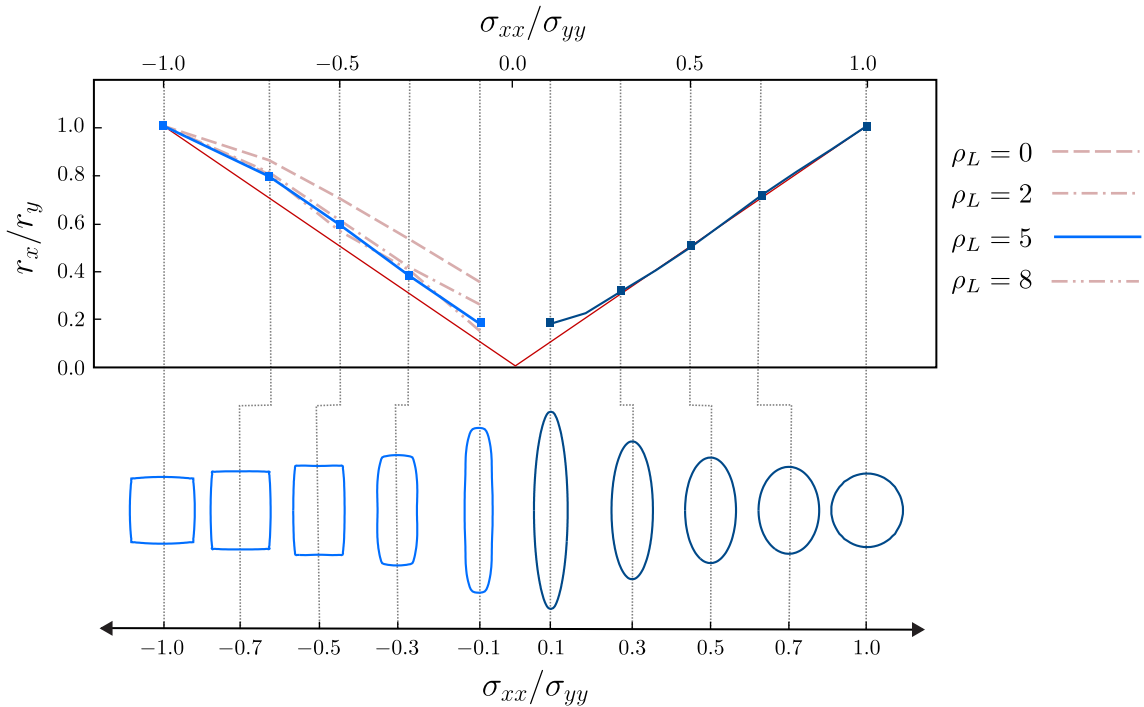
Fig. 17. Optimal hole shapes in a two-dimensional domain. Computationally obtained hole shapes and their aspect ratios r_x/r_y for different stress ratios $\alpha = \sigma_{xx}/\sigma_{yy}$. The analytically obtained aspect ratio is shown in red. The multiple curves for $\sigma_{xx}/\sigma_{yy} < 0$ are computed using different penalty values ρ_L, cf. (48). No penalty is applied when $\sigma_{xx}/\sigma_{yy} > 0$. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

the method of moving asymptotes (MMA) does not converge. In computations we integrate the second term in (48) numerically using the control mesh on the computational level ℓ_c. The required gradient information is obtained by differentiating the resulting discrete equations with respect to the vertex positions. The obtained gradient vector is added to the shape gradient vector $\boldsymbol{g}^{\ell_c}$.

With cost function (48) it is possible to compute the optimal hole shapes for positive as well as negative stress ratios. The obtained hole shapes and aspect ratios are shown in Fig. 17. For positive stress ratios the hole is of elliptical shape and for negative ratios it is a smoothed quadrilateral. The area of all the holes is equal due to the prescribed area constraint. As can be seen in Fig. 17, the obtained aspect ratios are in very good agreement with the analytical result indicated by the solid red line. The discrepancy for very small stress ratios is due to the finite size of the domain and the appearance of tall holes with crack-like stress concentrations requiring a finer discretisation. Finally, the effect of the penalty parameter ρ_L on the obtained aspect ratio is relatively mild.

5.1.3. Optimal hole shapes in a three-dimensional domain

The considered computational domain is a cube with a side length of $L = 4$ and is discretised with cells of size $1/20$. The initial geometry of the hole is a sphere with diameter $D = 1$ and is at level $\ell = 0$ approximated with a control mesh of 26 nodes. The optimisation level starts with $\ell_o = 0$ and is incremented until $\ell_o = \ell_c = 3$ is reached. The volume of the hole is constrained to remain constant during optimisation. This is achieved by computing the volume and its gradient with the three-dimensional extension of (47).

First we choose the two stress components $\sigma_{xx} = \sigma_{yy}$ as equal and only modify the σ_{zz} stress component such that

$$\sigma_{xx}/\sigma_{zz} \in \{0.3, 0.5, 0.8, 1.0\}.$$

According to analytical results the optimised hole shapes are ellipsoids with semi-axis radius $r_x = r_y$ and have the aspect ratio $r_x/r_z = \sigma_{xx}/\sigma_{zz}$. Fig. 18 shows the computationally obtained ellipsoidal hole shapes and their aspect ratios. The computational and analytical results agree very well especially for large stress ratios. For smaller stress
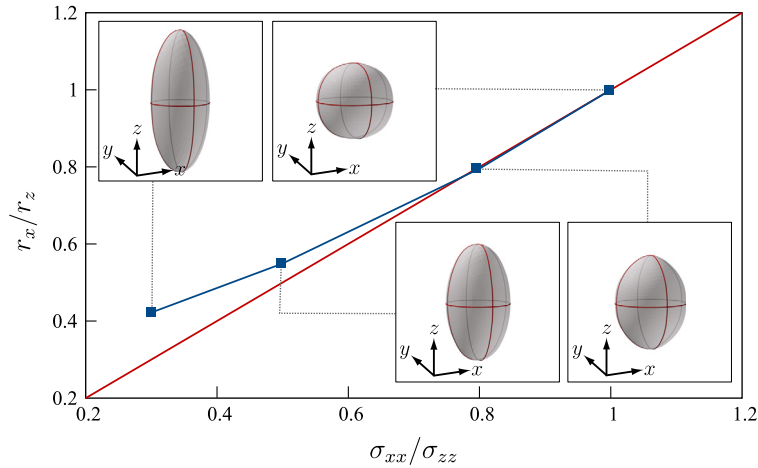
Fig. 18. Optimal hole shapes in a three-dimensional domain. Computationally obtained ellipsoidal hole shapes and their aspect ratios r_x/r_z for different stress ratios σ_{xx}/σ_{zz}. Note that $\sigma_{xx} = \sigma_{yy}$ and $r_x = r_y$ (up to discretisation errors). The analytically obtained aspect ratio is shown in red. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

ratios the discrepancy is due to the discretisation errors in resolving the more pronounced stress concentrations caused by taller holes.

Next, we keep the stress component σ_{zz} fixed and independently vary the two stress components σ_{xx} and σ_{yy}. To reduce the effect of domain size on the computationally obtained hole shapes we choose the domain dimensions in dependence of the stress ratios $\alpha_x = \sigma_{xx}/\sigma_{zz}$ and $\alpha_y = \sigma_{yy}/\sigma_{zz}$ such that

$$L_x = 3|\alpha_x|, \qquad L_y = 3|\alpha_y| \quad \text{and} \quad L_z = 3.$$

In all computations the cell size is constant ($5 \times 5 \times 5$ cells per unit volume) and is independent of the domain size.

As in the case of the preceding two-dimensional example in Section 5.1.2, for negative stress ratios the compliance cost function (5) is augmented with an integral penalising surface area changes, cf. (48). The penalty modified cost function prohibits the formation of multiple holes, which cannot be obtained with shape optimisation. The obtained hole shapes are shown in Fig. 19. The applied positive stress ratios result in ellipsoidal hole shapes with semi-axis ratios proportional to the stress ratios (up to discretisation errors). On the other hand, the applied negative stress ratios result in hole shapes where the cross-section in the x–y plane is an ellipse and in the x–z and y–z planes are smoothed quadrilaterals.

5.2. Combined shape optimisation and topology modification

This section introduces several examples in which shape optimisation is combined with a hole insertion technique. First, holes are introduced using the topology derivative. Subsequently, the hole shapes are optimised with the proposed multiresolution optimisation technique. This approach is in spirit similar to the classical *bubble method* by Eschenauer et al. [54]. In our present implementation we do not consider the merging or removing of holes, hence our approach is more restrictive than most newer topology optimisation methods. For excellent recent reviews on various topology optimisation methods see [35,55,56].

Without going into details, the topology derivative at a point gives the change of the cost function when a small hole is introduced at that point, see e.g. [49,50,57]. To this end, in addition to the original domain Ω a modified domain Ω_r containing a hole of radius of r is considered. The hole shape is either a circle or sphere depending on the dimension of the domain. The cost function $J(\Omega_r, \boldsymbol{u}_r)$ of the problem defined on Ω_r can be expressed with a series expansion

$$J(\Omega_r, \boldsymbol{u}_r) = J(\Omega, \boldsymbol{u}) + f(r) D_T J(\Omega, \boldsymbol{u}) + \mathrm{O}(r^d) \tag{49}$$

where $f(r)$ is the size of the hole and $D_T J(\Omega, \boldsymbol{u})$ is the topology derivative. The hole size in 2D is $f(r) = r^2\pi$ and in 3D it is $f(r) = 4\pi r^3/3$. The topology derivative can be related to the shape derivative by considering the expansion of an infinitesimally small hole [50]. In our implementation we use the expressions given in [58,59] for

(a) Positive stress ratios $\sigma_{xx}/\sigma_{zz} > 0$ and $\sigma_{yy}/\sigma_{zz} > 0$.

(b) Negative stress ratios $\sigma_{xx}/\sigma_{zz} < 0$ and $\sigma_{yy}/\sigma_{zz} < 0$.
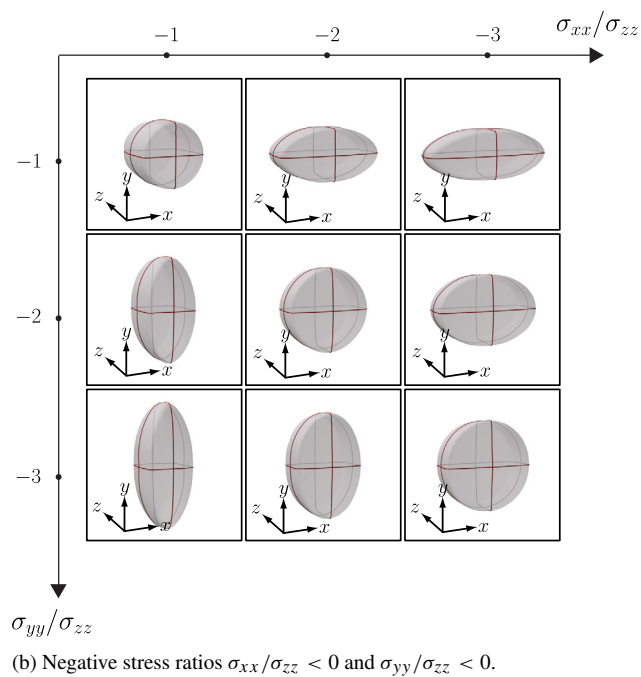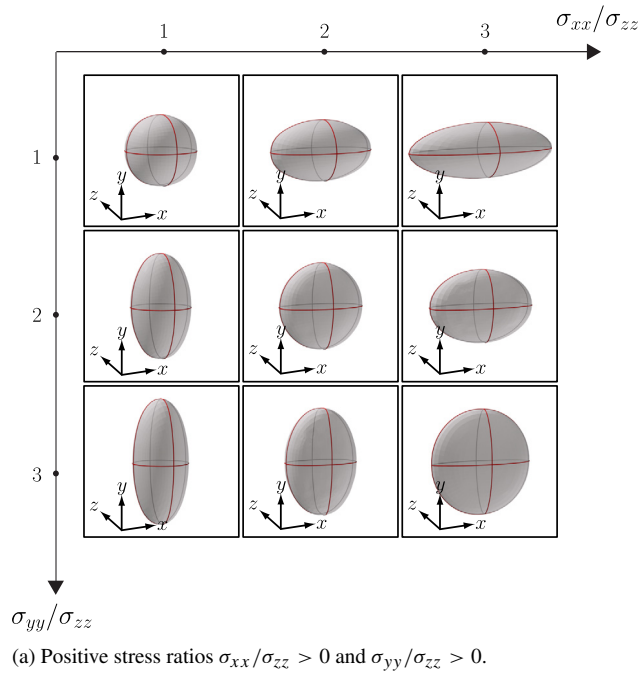
Fig. 19. Optimal hole shapes in a three-dimensional domain. Computationally obtained hole shapes for different stress ratios.

the topology derivative of the compliance cost function. Depending on the dimension of the domain we obtain the topology derivative for a material with Poisson's ratio ν with the following equations:

- two-dimensional elasticity, plane-stress

$$D_T J(\Omega, \boldsymbol{u}) = \frac{4}{1 + \nu} \boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\epsilon}(\boldsymbol{u}) - \frac{1 - 3\nu}{1 - \nu^2} \operatorname{Tr} \boldsymbol{\sigma}(\boldsymbol{u}) \operatorname{Tr} \boldsymbol{\epsilon}(\boldsymbol{u})$$

(50)

- two-dimensional elasticity, plane-strain

$$D_T J(\Omega, \boldsymbol{u}) = 4(1 - \nu)\boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\epsilon}(\boldsymbol{u}) - \frac{(1 - 4\nu)(1 - \nu)}{1 - 2\nu} \operatorname{Tr}\boldsymbol{\sigma}(\boldsymbol{u}) \operatorname{Tr}\boldsymbol{\epsilon}(\boldsymbol{u}) \tag{51}$$

- three-dimensional elasticity

$$D_T J(\Omega, \boldsymbol{u}) = \frac{3}{4} \frac{1 - \nu}{7 - 5\nu} \left[10\boldsymbol{\sigma}(\boldsymbol{u}) : \boldsymbol{\epsilon}(\boldsymbol{u}) - \frac{1 - 5\nu}{1 - 2\nu} \operatorname{Tr}\boldsymbol{\sigma}(\boldsymbol{u}) \operatorname{Tr}\boldsymbol{\epsilon}(\boldsymbol{u}) \right]. \tag{52}$$

We use the isocontours of the topology derivative $D_T J(\Omega, \boldsymbol{u})$ to determine the location and shape of the hole to be introduced. To this end, we introduce a control mesh which approximates the boundary of the region to be removed from the domain. Although this step is presently performed manually, it is feasible to automate it. This is particularly straightforward in case of two-dimensional domains with a polygon as the boundary of the hole. For three-dimensional problems the isocontour of the topology derivative can be first extracted with a marching-cube algorithm and subsequently remeshed in order to obtain a uniform control mesh with mostly regular vertices, see e.g. [60].

After a hole is generated, it is subjected to shape optimisation. During the shape optimisation the size of the hole is allowed to increase by a prescribed amount. For instance, the area constraint (46) is modified as follows

$$A(\boldsymbol{x}^{\ell_c}) \geq \rho_A A_0, \tag{53}$$

where ρ_A is a user prescribed scalar and A_0 is the area of the initial hole. In this context the boundary of each hole is treated as a separate multiresolution curve or surface.

In passing we note that instead of the topology derivative it would also be possible to determine the location and shape of the introduced holes with the density based SIMP technique widely used in topology optimisation [61,62].

5.2.1. Cantilever

The cantilever shown in Fig. 20(a) is a widely studied benchmark example in topology and shape optimisation. Most relevant to our study are the results obtained by Eschenauer et al. [54] using the bubble method, which as previously mentioned is similar to our approach. In our computations the rectangular domain is loaded with a distributed force $\bar{t}_y = 10$ close to the lower right corner and Young's modulus and Poisson's ratio are chosen with $E = 100$ and $\nu = 0.4$, respectively. The immersed finite element grid contains 100×100 cells of uniform size.

The final topology and shape optimised geometry is shown in Fig. 20(c). This geometry is similar to the ones presented in [54,63] and has been obtained in three steps.

1. The first step is a shape optimisation step. During optimisation only the top boundary of the plate is allowed to move based on the computed shape gradients. In the optimisation level $\ell_o = 0$ the top boundary is represented with a control polygon using 2 elements and 3 vertices. In the twice subdivided computational level $\ell_c = 2$ the control polygon contains 16 elements. During subdivision the two end nodes are tagged as corner so that they do not move horizontally. In addition, during iterative shape optimisation the right corner node is allowed to move only vertically. Moreover, we apply area constraints of the form (53) so that the domain size is reduced. The geometry obtained with shape optimisation is shown in Fig. 20(b).
2. The second step is a topology modification step. After computing the topology derivative we manually introduce two holes at locations with minimum topology derivative, see Fig. 20(b). The first hole is triangular shaped and splits the clamped boundary into an upper and a lower part. The second hole is square shaped and is located inside the domain. The holes have to be large enough so that they can be represented on the immersed finite element grid. According to [36] the minimum hole size has to be larger than

$$2\sqrt{d}(\alpha + 1)h, \tag{54}$$

where $d = 2$ is the dimensionality of the domain, α is the polynomial degree of the b-spline basis functions and h is the cell size.
3. The last step is again a shape optimisation step. The control polygons belonging to the previously introduced two holes with three and four nodes, respectively, are subdivided twice to obtain the computational control polygon at level $\ell_c = 2$. Subsequently the geometry of the two holes is iteratively optimised using shape optimisation. The optimisation is terminated before the two holes start to merge.
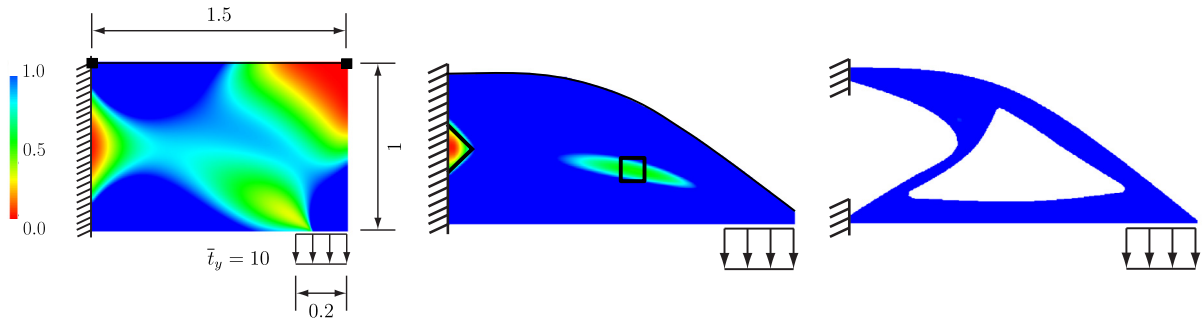
(a) Problem description. The top boundary is first optimised with constraints on the end nodes (solid squares) to prevent horizontal movement.

(b) Two holes are introduced at locations of minimum topology derivative.

(c) Final optimised geometry.

Fig. 20. Shape and topology optimisation of a cantilever truss. The isocontours indicate the topology derivative.
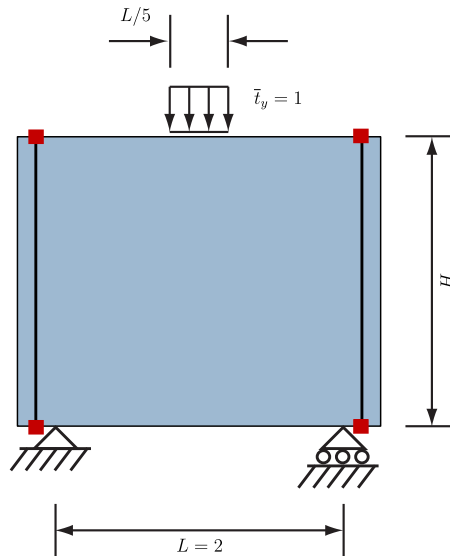

Fig. 21. Simply supported plate with aspect ratio H/L. Each of the two supports have a width of $0.05L$. The two bold vertical lines indicate the free boundaries which are allowed to move during shape optimisation. The two ends of each of the vertical lines (squares) are constrained to move only horizontally.

5.2.2. Simply supported plates with different aspect ratios

We consider three simply supported plates with different aspect ratios H/L, see Fig. 21. It is known from structural analysis that for slender plates a truss-like system and for stocky plates an arch-like system is more efficient. The plate is loaded with a symmetrically placed distributed vertical force $\bar{t}_y = 1$ of length $L/5$ on its top edge. Young's modulus and Poisson's ratio are chosen with $E = 100$ and $\nu = 0.4$, respectively. We consider three different aspect ratios $H/L \in \{0.25, 0.5, 1\}$ with the corresponding immersed finite element grids containing 80×200, 200×100 and 150×150 cells, respectively. In the computations the height is fixed to $H = 1$ and only the length L is varied.

As in the cantilever example, cf. Section 5.2.1, the final geometry is obtained in several steps, namely an initial shape optimisation step is followed by several topology modification and shape optimisation steps. In the initial shape optimisation step the two vertical boundaries of the plate are optimised while the domain area is allowed to reduce. At the coarse optimisation level $\ell_o = 0$ each edge is represented with two elements, which are three times subdivided to obtain the computational control polygon at level $\ell_c = 3$. In the subsequent topology modification step we semi-manually introduce triangle and square shaped holes, see Fig. 22 middle column. For each of the polygons the optimisation and computation levels are chosen with $\ell_o = 0$ and $\ell_c = 3$, respectively. The topology modification
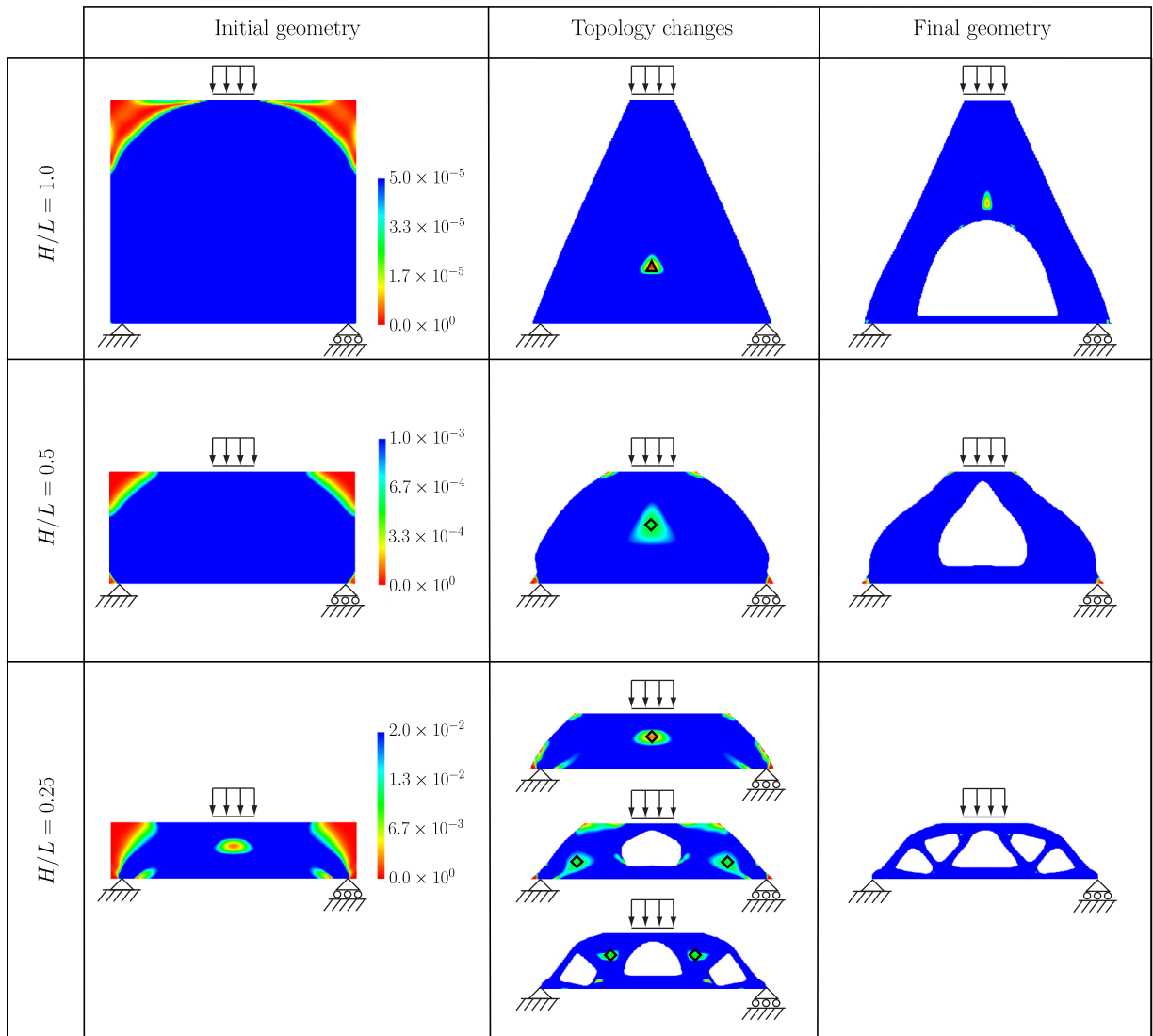
Fig. 22. Simply supported plates with different aspect ratios. In all snapshots the isocontours indicate the topology derivative. In the middle column the small triangular and square shaped holes introduced during the topology modification step are shown.

is followed by the shape optimisation of all the domain boundaries. The movement of the vertices under the distributed force and supports is constrained to remain fixed. In case of the plate with small $H/L = 0.25$ several more topology modification and shape optimisation steps are performed, see Fig. 22 middle column. The obtained geometries are shown in Fig. 22 right column. The optimisation is always terminated before any holes start to merge. As expected, we obtain for a $H/L = 1.0$ an arch-like structure and for $H/L = 0.25$ a truss-like structure.

5.2.3. Three-dimensional stool

In this last example we present the combined shape optimisation and topology modification of a three-dimensional solid, see Fig. 23. The initial domain is a truncated pyramid and is at its top loaded with a uniform distributed load $\bar{t}_z = 10$. At its bottom it is supported by four distributed roller supports each of size 0.2×0.2. Young's modulus and Poisson's ratio are chosen with $E = 100$ and $\nu = 0.4$, respectively. In the optimisation study only one quarter of the domain is considered and appropriate bounds and geometry tags are applied at the two planes of symmetry. The corresponding immersed finite element grid is of size $0.7 \times 0.7 \times 1$ and consists of $30 \times 30 \times 30$ cells.
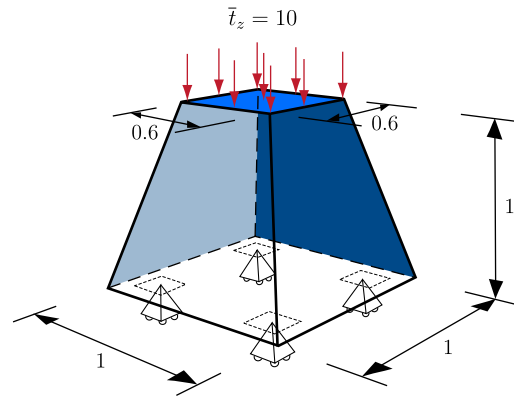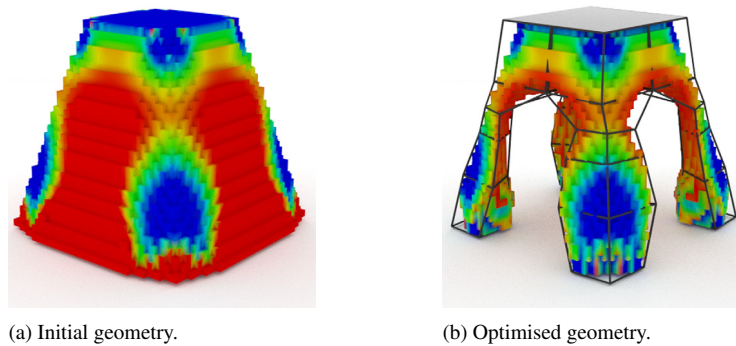
Fig. 23. Three-dimensional stool. Problem description. Roller supports are applied to all finite element nodes inside the regions of size 0.2×0.2 marked by dashed squares.

(a) Initial geometry.

(b) Optimised geometry.

Fig. 24. Three-dimensional stool. First topology modification step. The isocontours indicate the topology derivative. The wireframe represents the coarse control mesh.

The sequence of the performed topology modification and shape optimisation steps are shown in Figs. 24–27. In total two topology modification and two shape optimisation steps are performed. In each topology modification step we remove in one go a relatively large amount of material by deleting computational cells with topology derivative below a threshold. In the first topology modification step illustrated in Fig. 24 all cells with topology derivative $D_T J(\Omega, \boldsymbol{u}) \leq 0.025$ are removed. Subsequently, we semi-manually generate the coarse resolution subdivision control mesh depicted in Fig. 24(b) for representing the new topology. In the following shape optimisation step, see Fig. 25, the generated control mesh serves as the optimisation level $\ell_o = 0$ and the computation level is chosen with $\ell_c = 2$. During the shape optimisation the volume of the domain is constrained to remain constant. In the second topology modification step shown in Fig. 26 all cells with topology derivative $D_T J(\Omega, \boldsymbol{u}) \leq 0.04$ are removed. This is followed by a semi-manual control mesh generation, see Fig. 26(b), and the final shape optimisation step shown Fig. 27.

6. Summary and conclusions

A multiresolution optimisation technique based on subdivision surfaces was introduced. The domain boundaries are described with subdivision surfaces and the domain boundary value problem is discretised with an immersed finite element technique. The wavelet-like multiresolution decomposition of the domain boundary yields a low resolution control mesh and a sequence of detail vectors. The control mesh at any specific refinement level can be reconstructed on-the-fly with the introduced synthesis operator. Editing the coarse levels leads to large-scale geometry changes while editing fine levels leads to small-scale geometry changes. In addition, the multiresolution editing semantics allows the decoupling of the choice of the editing level from the size of the geometric features present in the geometry. In the proposed approach we start optimising the coarsest control mesh and successively increase the optimisation level each time a minimum is reached. The domain geometry is always described with a fine control mesh on the immersed finite
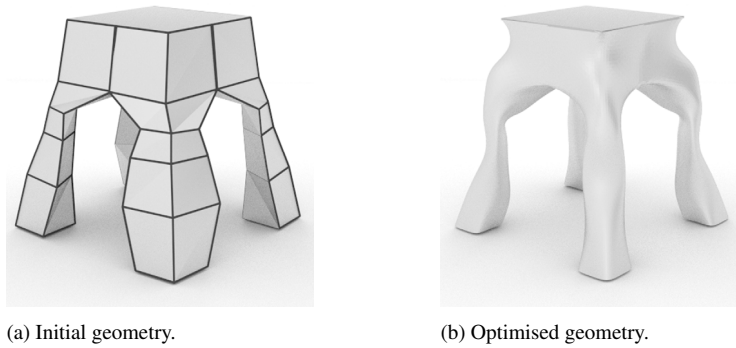
(a) Initial geometry.

(b) Optimised geometry.

Fig. 25. Three-dimensional stool. First shape optimisation step. The wireframe represents the coarse resolution mesh used for optimisation.
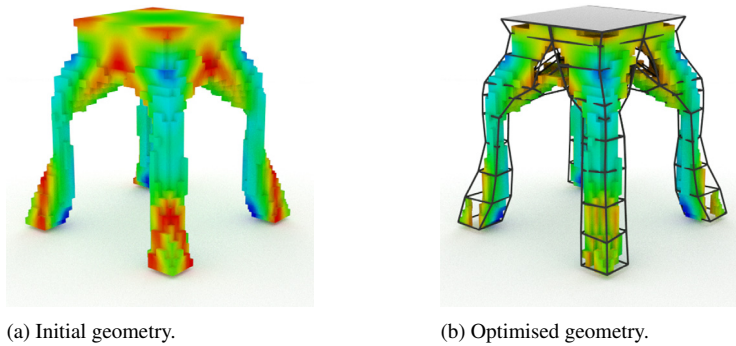

(a) Initial geometry.

(b) Optimised geometry.

Fig. 26. Three-dimensional stool. Second topology modification step. The isocontours indicate the topology derivative. The wireframe represents the coarse control mesh.

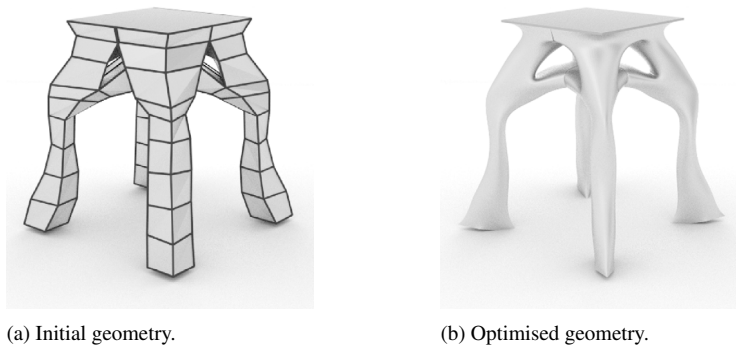(a) Initial geometry.

(b) Optimised geometry.

Fig. 27. Three-dimensional stool. Second shape optimisation step. The wireframe represents the coarse resolution mesh used for optimisation.

element grid, independently from the control mesh level used for optimisation. Hence, any fine scale geometric details, like fillets or surface undulations, are faithfully represented on the discretisation grid. As our examples demonstrate multiresolution shape optimisation enables us to find better optima and is exceedingly robust, partly due to the use of the immersed finite element technique.

The multiresolution editing semantics appears to be particularly appealing for isogeometric analysis because it enables the full decoupling of the geometry and the analysis representations of the same geometry. It allows to seamlessly map variables and fields between the two representations irrespective of their resolutions. Beyond optimisation this decoupling can be useful in a number of applications, such as for computing fast approximate solutions or multigrid and multilevel preconditioners [32]. The presented multiresolution techniques can also be extended to non-uniform rational b-splines (NURBS), which are the more commonly used basis functions in industrial software. It is straight-forward to include rational b-splines through the use of homogenised coordinates, see, e.g., [64]. In order to consider

non-uniform b-splines it is instructive to consult previous works on non-uniform subdivision [65,66]. Although we focused in the present paper on uniform subdivision refinement and coarsening, it is conceivable (and desirable) to develop adaptive multiresolution algorithms in the spirit of hierarchical b-splines [67–69]. The utility of adaptive geometry representations in shape optimisation has already been demonstrated, for instance, with b-spline surfaces refined by knot insertion [8] and reparameterisation of Bezier surfaces [70]. Furthermore, in the present paper holes were introduced by manually fitting control meshes to the isocontours of the topology derivative. This process can be automated using techniques for extracting subdivision control meshes from isocontours [71]. The related issue of merging of holes can be achieved with approximate Boolean operations for subdivision surfaces [72]. In closing, it is worth emphasising that basic subdivision techniques recently became available in a number of engineering design software, including Autodesk Fusion 360, PTC Creo and CATIA, which most likely will increase their use in future engineering practice.

Acknowledgements

References

[1] T. Hughes, J. Cottrell, Y. Bazilevs, Isogeometric analysis: CAD, finite elements, NURBS, exact geometry and mesh refinement, Comput. Methods Appl. Mech. Engrg. 194 (2005) 4135–4195.
[2] V. Braibant, C. Fleury, Shape optimal design using b-splines, Comput. Methods Appl. Mech. Engrg. 44 (1984) 247–267.
[3] R.T. Haftka, R.V. Grandhi, Structural shape optimization—a survey, Comput. Methods Appl. Mech. Engrg. 57 (1986) 91–106.
[4] K.-U. Bletzinger, S. Kimmich, E. Ramm, Efficient modeling in shape optimal design, Comput. Syst. Eng. 2 (1991) 483–495.
[5] N. Olhoff, M.P. Bendsøe, J. Rasmussen, On CAD-integrated structural topology and design optimization, Comput. Methods Appl. Mech. Engrg. 89 (1) (1991) 259–279.
[6] F. Beux, A. Dervieux, A hierarchical approach for shape optimization, Eng. Comput. 11 (1) (1994) 25–48.
[7] B.A. El Majd, J.-A. Désidéri, R. Duvigneau, Multilevel strategies for parametric shape optimization in aerodynamics, Eur. J. Comput. Mech./Revue Européenne de Mécanique Numérique 17 (1–2) (2008) 149–168.
[8] X. Han, D.W. Zingg, An adaptive geometry parametrization for aerodynamic shape optimization, Optim. Eng. 15 (1) (2014) 69–91.
[9] K.-U. Bletzinger, M. Firl, J. Linhard, W. R., Optimal shapes of mechanically motivated surfaces, Comput. Methods Appl. Mech. Engrg. 199 (2010) 324–333.
[10] C. Le, T. Bruns, D. Tortorelli, A gradient-based, parameter-free approach to shape optimization, Comput. Methods Appl. Mech. Engrg. 200 (9) (2011) 985–996.
[11] K.-U. Bletzinger, A consistent frame for sensitivity filtering and the vertex assigned morphing of optimal shape, Struct. Multidiscip. Optim. 49 (6) (2014) 873–895.
[12] F. Cirak, M. Scott, E. Antonsson, M. Ortiz, P. Schröder, Integrated modeling, finite-element analysis, and engineering design for thin-shell structures using subdivision, Comput. Aided Des. 34 (2002) 137–148.
[13] J. Kiendl, R. Schmidt, R. Wüchner, K.-U. Bletzinger, Isogeometric shape optimization of shells using semi-analytical sensitivity analysis and sensitivity weighting, Comput. Methods Appl. Mech. Engrg. 274 (2014) 148–167.
[14] W. Wall, M. Frenzel, C. Cyron, Isogeometric structural shape optimisation, Comput. Methods Appl. Mech. Engrg. 197 (2008) 2976–2988.
[15] K. Bandara, F. Cirak, G. Of, O. Steinbach, J. Zapletal, Boundary element based multiresolution shape optimisation in electrostatics, J. Comput. Phys. 297 (2015) 584–598.
[16] D. Zorin, P. Schröder, Subdivision for Modeling and Animation, SIGGRAPH 2000 Course Notes, 2000.
[17] J. Peters, U. Reif, Subdivision surfaces, in: Springer Series in Geometry and Computing, Springer, 2008.
[18] J.M. Lane, R.F. Riesenfeld, A theoretical development for the computer generation and display of piecewise polynomial surfaces, IEEE Trans. Pattern Anal. Mach. Intell. PAMI-2 (1980) 35–46.
[19] D. Doo, M. Sabin, Behavior of recursive division surfaces near extraordinary points, Comput. Aided Des. 10 (1978) 356–360.
[20] E. Catmull, J. Clark, Recursively generated b-spline surfaces on arbitrary topological meshes, Comput. Aided Des. 10 (1978) 350–355.
[21] D. Zorin, P. Schröder, W. Sweldens, Interactive multiresolution mesh editing, in: SIGGRAPH 1997 Conference Proceedings, ACM Press/Addison-Wesley Publishing Co., Los Angeles, CA, 1997, pp. 259–268.
[22] M. Lounsbery, T.D. DeRose, J. Warren, Multiresolution analysis for surfaces of arbitrary topological type, ACM Trans. Graph. 16 (1997) 34–73.
[23] E.J. Haug, K.K. Choi, V. Komkov, Design Sensitivity Analysis of Structural Systems, Academic Press, 1986.
[24] J. Sokolowski, J.P. Zolesio, Introduction to Shape Optimization: Shape Sensitivity Analysis, Springer, 1992.
[25] J. Haslinger, R.A.E. Mäkinen, Introduction to Shape Optimization: Theory, Approximation, and Computation, SIAM, Philadelphia, 2003.
[26] T. Rüberg, F. Cirak, An immersed finite element method with integral equation correction, Int. J. Numer. Methods Eng. 86 (2011) 93–114.
[27] R. Sanches, P. Bornemann, F. Cirak, Immersed b-spline (i-spline) finite element method for geometrically complex domains, Comput. Methods Appl. Mech. Engrg. 200 (2011) 1432–1445.

[28] D. Schillinger, L. Dede, M.A. Scott, J.A. Evans, B.M. J., E. Rank, T.J.R. Hughes, An isogeometric design-through-analysis methodology based on adaptive hierarchical refinement of NURBS, immersed boundary methods, and T-spline CAD surfaces, Comput. Methods Appl. Mech. Engrg. 249–253 (2012) 116–150.

[29] T. Rüberg, F. Cirak, Subdivision-stabilised immersed b-spline finite elements for moving boundary flows, Comput. Methods Appl. Mech. Engrg. 209–212 (2012) 266–283.

[30] J. Norato, R. Haber, D. Tortorelli, M. Bendsoe, A geometry projection method for shape optimization, Int. J. Numer. Methods Eng. 60 (2004) 2289–2312.

[31] G. Allaire, F. Jouve, A.-M. Toader, Structural optimization using sensitivity analysis and a level-set method, J. Comput. Phys. 194 (2004) 363–393.

[32] P.F. Antonietti, A. Borzì, M. Verani, Multigrid shape optimization governed by elliptic PDEs, SIAM J. Control Optim. (ISSN: 0363-0129) 51 (2) (2013) 1417–1440.

[33] M. Delfour, J.-P. Zolésio, Shapes and geometries: analysis, differential calculus, and optimization, in: Advances in Design and Control, SIAM, Philadelphia, 2001.

[34] G. Holzapfel, Nonlinear Solid Mechanics: A Continuum Approach for Engineering, Wiley-Blackwell, 2000.

[35] N.P. van Dijk, K. Maute, M. Langelaar, F. Van Keulen, Level-set methods for structural topology optimization: a review, Struct. Multidiscip. Optim. 48 (3) (2013) 437–472.

[36] T. Rüberg, F. Cirak, A fixed-grid b-spline finite element technique for fluid–structure interaction, Internat. J. Numer. Methods Fluids 74 (9) (2014) 623–660.

[37] J. Nitsche, Über ein Variationsprinzip zur Lösung von Dirichlet-Problemen bei Verwendung von Teilräumen, die keinen Randbedingungen unterworfen sind, Abh. Math. Semin. Univ. Hambg. 36 (1971) 9–15.

[38] K. Höllig, Finite Element Methods with B-splines, SIAM Frontiers in Applied Mathematics, 2003.

[39] H. Prautzsch, W. Boehm, M. Paluszny, Bezier and B-Spline Techniques, Springer Verlag, 2002.

[40] L. Ying, D. Zorin, Nonmanifold Subdivision, in: IEEE Visualization 2001, San Diego, CA, 2001, pp. 325–332.

[41] A. Finkelstein, D.H. Salesin, Multiresolution curves, in: Proceedings of the 21st Annual Conference on Computer Graphics and Interactive Techniques, ACM, 1994, pp. 261–268.

[42] S. Gortler, M. Cohen, Hierarchical and variational geometric modeling with wavelets, in: Proceedings of the 1995 Symposium on Interactive 3D Graphics, 1995, pp. 35–42.

[43] C. de Boor, The quasi-interpolant as a tool in elementary polynomial spline theory, Approx. Theory (1973) 269–276.

[44] N. Litke, A. Levin, P. Schröder, Fitting subdivision surfaces, in: IEEE Visualization 2001, San Diego, CA, 2001, pp. 319–324.

[45] O. Sorkine, D. Cohen-Or, Y. Lipman, M. Alexa, C. Rössl, H.-P. Seidel, Laplacian surface editing, in: Proceedings of the 2004 Eurographics/ACM SIGGRAPH Symposium on Geometry Processing, ACM, 2004, pp. 175–184.

[46] K. Svanberg, The method of moving asymptotes—a new method for structural optimization, Int. J. Numer. Methods Eng. 24 (1987) 359–373.

[47] K. Svanberg, A class of globally convergent optimization methods based on conservative convex separable approximations, SIAM J. Optim. 12 (2002) 555–573.

[48] S.G. Johnson, The NLopt nonlinear-optimization package, http://ab-initio.mit.edu/nlopt.

[49] H.A. Eschenauer, N. Olhoff, Topology optimization of continuum structures: a review, Appl. Mech. Rev. 54 (4) (2001) 331–390.

[50] A.A. Novotny, R. Feijóo, E. Taroco, C. Padra, Topological sensitivity analysis, Comput. Methods Appl. Mech. Engrg. 192 (7) (2003) 803–829.

[51] A. Cherkaev, Y. Grabovsky, A. Movchan, S. Serkov, The cavity of the optimal shape under the shear stresses, Int. J. Solids Struct. 35 (33) (1998) 4391–4410.

[52] E.S. Kristensen, N.F. Madsen, On the optimum shape of fillets in plates subjected to multiple in-plane loading cases, Int. J. Numer. Methods Eng. 10 (5) (1976) 1007–1019.

[53] E. Cervera, J. Trevelyan, Evolutionary structural optimisation based on boundary representation of NURBS. Part I: 2D algorithms, Comput. Struct. 83 (23) (2005) 1902–1916.

[54] H.A. Eschenauer, V.V. Kobelev, A. Schumacher, Bubble method for topology and shape optimization of structures, Struct. Optim. 8 (1) (1994) 42–51.

[55] O. Sigmund, K. Maute, Topology optimization approaches, Struct. Multidiscip. Optim. 48 (6) (2013) 1031–1055.

[56] J.D. Deaton, R.V. Grandhi, A survey of structural and multidisciplinary continuum topology optimization: post 2000, Struct. Multidiscip. Optim. 49 (1) (2014) 1–38.

[57] S. Garreau, P. Guillaume, M. Masmoudi, The topological asymptotic for PDE systems: the elasticity case, SIAM J. Control Optim. 39 (6) (2001) 1756–1778.

[58] A. Novotny, R. Feijóo, E. Taroco, C. Padra, Topological-shape sensitivity method: theory and applications, in: IUTAM Symposium on Topological Design Optimization of Structures, Machines and Materials, Springer, 2006, pp. 469–478.

[59] A. Novotny, R. Feijóo, E. Taroco, C. Padra, Topological sensitivity analysis for three-dimensional linear elasticity problem, Comput. Methods Appl. Mech. Engrg. 196 (41) (2007) 4354–4364.

[60] M. Botsch, L. Kobbelt, M. Pauly, P. Alliez, B. Lévy, Polygon Mesh Processing, CRC press, 2010.

[61] M.P. Bendsoe, Optimal shape design as a material distribution problem, Struct. Optim. 1 (4) (1989) 193–202.

[62] M.P. Bendsoe, O. Sigmund, Topology Optimization: Theory, Methods and Applications, Springer, 2003.

[63] M.Y. Wang, X. Wang, D. Guo, A level set method for structural topology optimization, Comput. Methods Appl. Mech. Engrg. 192 (1) (2003) 227–246.

[64] F. Cirak, Q. Long, Subdivision shells with exact boundary control and non-manifold geometry, Int. J. Numer. Methods Eng. 88 (2011) 897–923.

[65] T. Sederberg, J. Zheng, A. Bakenov, A. Nasri, T-splines and T-NURCCs, in: SIGGRAPH 2003 Conference Proceedings, San Diego, CA, 2003, pp. 477–484.

K. Bandara et al. / Comput. Methods Appl. Mech. Engrg. 300 (2016) 510–539

539

[66] T. Cashman, U. Augsdörfer, N. Dodgson, M. Sabin, NURBS with extraordinary points: high-degree, non-uniform, rational subdivision schemes, in: SIGGRAPH 2009 Conference Proceedings, New Orleans, LA, 2009, pp. 46:1–46:9.

[67] E. Grinspun, P. Krysl, P. Schröder, CHARMS: A simple framework for adaptive simulation, in: SIGGRAPH 2002 Conference Proceedings, San Antonio, TX, 2002, pp. 281–290.

[68] P.B. Bornemann, F. Cirak, A subdivision-based implementation of the hierarchical b-spline finite element method, Comput. Methods Appl. Mech. Engrg. 253 (2013) 584–598.

[69] X. Wei, Y. Zhang, T. J. Hughes, M.A. Scott, Truncated hierarchical Catmull–Clark subdivision with local refinement, Comput. Methods Appl. Mech. Engrg. 291 (2015) 1–20.

[70] J.-A. Désidéri, B.A. El Majd, A. Janka, Nested and self-adaptive bézier parameterizations for shape optimization, J. Comput. Phys. 224 (1) (2007) 117–131.

[71] D. Bommes, B. Lévy, N. Pietroni, E. Puppo, C. Silva, M. Tarini, D. Zorin, Quad-mesh generation and processing: a survey, Comput. Graph. Forum 32 (6) (2013) 51–76.

[72] H. Biermann, D. Kristjansson, D. Zorin, Approximate boolean operations on free-form solids, in: Proceedings of the 28th Annual Conference on Computer Graphics and Interactive Techniques, 2001, pp. 185–194.