Automatically imposing incremental boundary displacements for valid mesh morphing and curving

Eloi Ruiz-Gironés[a], Abel Gargallo-Peiró[a], Josep Sarrate[b], Xevi Roca[a,*]

[a] *Computer Applications in Science and Engineering, Barcelona Supercomputing Center, 08034 Barcelona, Spain.*
[b] *Laboratori de Càlcul Numèric (LaCàN), Departament d'Enginyeria Civil i Ambiental, Universitat Politècnica de Catalunya (UPC), Campus Nord UPC, 08034 Barcelona, Spain.*

Abstract

We present a new incremental mesh morphing method obtained by proposing and discretizing a solution procedure for the continuous morphing problem. Our method seeks a diffeomorphism that transforms an initial domain to a final domain by only prescribing the boundary displacement. To this end, we propose to minimize the distortion of the morphing mapping constrained to satisfy the imposed boundary displacement. To solve this problem, we consider an augmented Lagrangian method in Hilbert spaces that incorporates the boundary condition in the objective function using the Lagrange multipliers and a penalty parameter. The distortion is devised to penalize the appearance of non-invertible mappings and therefore, we do not need to equip our discrete implementation with untangling capabilities. Moreover, we introduce a weight function to improve the quality of the deformation and thus, the robustness of the non-linear solver. The discretization of the continuous augmented Lagrangian method leads to a mesh morphing method suitable for large displacements and rotations of meshes with non-uniform sizing, and mesh curving of highly stretched high-order meshes.

Keywords: Mesh morphing, mesh moving, mesh curving, smoothing, boundary displacements, augmented Lagrangian

1. Introduction

In the last decades, mesh morphing methods [1, 2] have attracted considerable attention from the computational methods and computer graphics communities. This attention has been prompted by the ability of these methods to deform an initial mesh to a target mesh without introducing inverted elements

*Corresponding author
Email addresses: eloi.ruizgirones@bsc.es (Eloi Ruiz-Gironés), abel.gargallo@bsc.es (Abel Gargallo-Peiró), jose.sarrate@upc.edu (Josep Sarrate), xevi.roca@bsc.es (Xevi Roca)

by only prescribing the mesh boundary displacement. This capability is critical both in applications that might require only one target configuration, *e.g.* high-order mesh curving, or several target configurations, *e.g.* moving meshes for fluid-structure interaction.

One key ingredient of mesh morphing methods is how to relocate the inner nodes of the mesh to accommodate the imposed boundary displacement without introducing inverted elements. The relocation of the inner nodes has been extensively addressed in the existing mesh morphing literature and it is mainly performed using solid mechanics analogies [3–9], optimizing mesh quality functionals [10–19], and solving specific-purpose PDE's [20–22]. Ordered from less to more robust approaches, existing methods avoid the insertion of inverted elements by using: non-explicit heuristics, explicit penalization of inverted elements, and untangling techniques to remove existing invalid elements. It is important to point out that untangling techniques usually feature the highest computational cost.

Another key ingredient is how to impose the boundary displacement that determines the target configuration. Unfortunately, this ingredient has not been extensively studied in the existing literature even though it drives the insertion of inverted elements. To illustrate this, we only need to consider a mesh where part of its boundary has been displaced a distance larger than the size of the corresponding boundary elements. These elements now feature an inverted orientation since they have been crossed over by their corresponding boundary nodes.

Existent methods can impose the boundary displacement in one stage or in several incremental stages. To enhance robustness, one stage methods need to feature untangling capabilities [3, 10, 11, 14–17] since large boundary displacements might introduce, as we said before, inverted elements next to the displaced boundary. On the contrary, by incrementally moving the mesh boundary [5, 6, 8] the insertion of inverted elements can be mitigated or even fully avoided.

To use several incremental stages it is necessary to define the trajectory of the boundary nodes and a methodology to decide feasible increments. Specifically, it is standard to use a linear [5, 6, 8] or non-linear [8] homotopy to characterize the trajectory from the initial to the target boundary. Then, a full linear step from the current to the target configuration is considered. While a valid mesh is not obtained, the length of the step along the current linear trajectory is reduced. This procedure is iterated until a valid mesh matching the target boundary configuration is obtained. This approach explicitly penalizes the insertion of inverted elements enhancing the robustness of mesh morphing methodology.

However, the number of intermediate mesh morphing stages might grow with the number of element layers surrounding the moving boundaries. Furthermore, it is required that all the intermediate stages of the boundary nodes along the trajectory determined by the linear homotopy are compatible with a valid configuration of the inner nodes. If an intermediate incompatible or non-valid boundary configuration is obtained, the mesh morphing process might not converge or low-quality or inverted elements might be generated. These issues might

arise when large displacements are prescribed, when morphing highly stretched boundary layer meshes and especially, when a straight-sided high-order mesh is highly curved to match a prescribed curved boundary. This is so since the trajectory between the piecewise straight-sided boundary and the curved boundary could feature invalid curved boundary elements on the intermediate stages.

These issues can be alleviated by iteratively refining the initial straight-sided mesh around the problematic boundary elements, until the mesh morphing procedure is successful. However, this could lead to non-desired additional human interaction. Accordingly, the main goal of this work is to propose a novel incremental mesh morphing formulation that automatically addresses the previous issues. To this end, we consider a continuous morphing problem that consists on transforming an initial domain to a deformed one by means of an invertible mapping. Specifically, we minimize the distortion of such mapping, while constraining the boundary values to the prescribed morphing condition.

To solve this continuous problem, we apply the augmented Lagrangian method in Hilbert spaces, see [23–25]. In this method, the boundary condition is introduced into the target functional in a weak sense by means of a penalty parameter and the corresponding Lagrange multipliers. The main idea of the augmented Lagrangian method is to solve a series of non-linear unconstrained problems with larger penalty parameters and successive approximations of the Lagrange multipliers. In each non-linear problem, the boundary constraint is further approximated, while keeping a valid mapping.

The second contribution is to introduce a weight function in order to increase the quality of the morphing mapping. The weight function assigns an *importance* to the different regions of the domain. Thus, low-quality regions are associated to high weights. Nevertheless, the weight function is not known *a priori*. Hence, we propose an iterative process at each step of the augmented Lagrangian procedure to approximate it. The main idea is to compute the weight function using the evolution of the distortion of the morphing mapping during the optimization process. Thus, while the augmented Lagrangian formulation deforms the domain to accommodate the boundary condition, the weight function is modified in order to improve the quality of the mapping.

The discretization of the continuous augmented Lagrangian formulation leads to a methodology to morph linear and high-order meshes. The proposed methodology has several advantages. The mesh boundary is not fixed during the optimization process since it is driven by the penalty parameter and the Lagrange multipliers. Thus, similarly to other incremental moving methods, the boundary condition is only satisfied at the end of the optimization process. However, it is worth pointing out that in our method we do not impose the trajectory of the boundary nodes during the deformation process since it is automatically computed by the augmented Lagrangian method. Moreover, since the mesh is valid during the whole minimization, it is not necessary to feature untangling capabilities. We propose to use a global non-linear solver, in which the nodes of the mesh are moved at the same time. To this end, we use a backtracking line-search method [26] in which the descent direction is computed using Newton's method and the step length is selected according to the Armijo rule. We point

out that to perform Newton's method we use the analytical derivatives of the objective function.

This work improves in several aspects our previous method proposed in [27]. The mesh morphing problem is now derived from a continuous problem of domain deformation. Accordingly, we use an augmented Lagrangian technique in Hilbert spaces. Moreover, the newly introduced weight function allows obtaining deformed meshes with higher quality and to improve the robustness of the non-linear solver. The main reason is that quality is not only increased in the final mesh, but also in the intermediate steps of the optimization process. Finally, new and more complex examples are considered, that show the application of the proposed methodology in several test cases.

The rest of the paper is structured as follows. In Section 2, we review the existing literature related to the presented work. In Section 3, we formulate the constrained minimization problem using the augmented Lagrangian method. Next, in Section 4 we detail the implementation of the method. In Section 5, we present several examples to show the features of the proposed method. Finally, in Section 6, we detail the conclusions and the future work.

2. Related Work

In several applications, mesh untangling and smoothing can be required to repair invalid elements and improve the overall mesh quality when applying a mesh morphing technique. In these cases, the boundary nodes are displaced to new positions, and it is necessary to recover a new mesh composed of valid elements. There are different formulations to define a mesh morphing technique. For instance, in the solid mechanics approach, the mesh is moved using linear or non-linear elasticity analogies [3–9]. Other approaches consist on minimizing an objective function that measures a quantity of interest of the mesh, such as the mesh distortion [10, 11, 18, 28], the minimum Jacobian of the iso-parametric mapping [16], or by formulating a variational minimization problem [21]. Finally, it is possible to define the mesh morphing process in terms of a PDE [20]. In all these approaches, it is necessary to fix the positions of the boundary nodes in order to satisfy the prescribed mesh morphing displacement. Usually, the morphing displacement is introduced as a boundary condition for the mesh morphing problem. In the most common approach, the positions of the boundary nodes are fixed, and then, the locations of the inner nodes are computed accordingly. However, when the boundary nodes are displaced in the initial step, invalid elements may appear, and they could hinder the convergence of the mesh morphing method, especially when complex geometries or non-uniform element sizes are present.

In reference [29], the authors propose to solve a constrained minimization problem in which the constraint is the final positions of the boundary nodes. To this end, they apply a penalty method to solve a series of optimization problems to enforce the constraint. Other approaches first project the boundary nodes to the CAD model, and then pose an unconstrained minimization problem in which the boundary nodes can slide along the geometric entity they belong

to [15, 16, 30, 31]. In this case, the imposition of the boundary condition is more flexible than prescribing a fixed nodal value. The only requisite is that the boundary nodes are located on the required geometric entity. In [17], the authors combine a mesh curving technique with a geometric accuracy measure into a single functional. The boundary nodes are free to slide along the geometric entities taking into account the geometric accuracy.

In reference [32], the authors introduce a geometric accuracy measure, and it is later combined with a mesh curving technique in [33]. In this approach, all the nodes of the mesh are free to move in the space, while the boundary nodes take into account the geometric error. In the proposed optimization process, the geometric accuracy term is treated as a boundary condition of the mesh curving problem by means of a penalty method. In this approach, the boundary nodes are not restricted to move along the geometric model. Instead, the boundary mesh approximates the CAD model in a weak sense.

References [5, 6, 8, 20, 34] propose an incremental displacement of the boundary nodes from the initial positions to the target ones. In this manner, inverted elements may not appear, and the optimization process is more robust. To this end, the optimization method tries a full step to the target positions. If the optimization process fails, the step is reduced until the optimization converges. This approach is iterated until the final positions of the nodes are reached.

In incremental stepping methods, invalid intermediate boundary configurations might appear hampering the capability to recover a valid volume mesh. This is not the case with our method, since the boundary conditions are weakly imposed, and we penalize the appearance of inverted elements during the optimization process. To avoid the appearance of invalid boundary configurations it is also possible to constrain the boundary nodes to move on top of the CAD entities and incorporate untangling capabilities [15, 17, 35]. On the contrary, in this work, the boundary nodes are not constrained to be on top of the boundaries, and we do not need untangling capabilities. There are other methods, proposed for linear meshes, where the boundary condition is incorporated in the minimization functional [12, 22, 29]. However, these approaches do not use an augmented Lagrangian formulation to incrementally impose the boundary conditions as proposed herein. Furthermore, we have checked our method not only for large displacements, but for non-uniform sizing, highly stretched elements, and curved high-order meshes.

3. Formulation of the Optimization Process

In this section, we first formulate a constrained minimization problem for mesh morphing, Section 3.1. Then, we detail the augmented Lagrangian continuous framework to impose the desired boundary constraint, Section 3.2. Finally, in Section 3.3 we state the optimization framework for a piece-wise polynomial discretization of the domain.

3.1. Continuous framework

Given an initial domain, $\Omega_I \subset \mathbb{R}^n$, we want to characterize a morphed domain, $\Omega_P \subset \mathbb{R}^n$, in terms of a diffeomorphism $\boldsymbol{\phi}^*$ from Ω_I to Ω_P. To this end, we seek a diffeomorphism in the Hilbert space $\left[\mathcal{H}^1(\Omega_I)\right]^n$, the space of square-integrable vector functions with square-integrable gradients. The diffeomorphism $\boldsymbol{\phi}^*$ has to satisfy the morphing boundary condition, and feature optimal point-wise distortion [28]. Herein we consider,

$$\begin{aligned} \boldsymbol{\phi}^* &= \underset{\boldsymbol{\phi}\in[\mathcal{H}^1(\Omega_I)]^n}{\operatorname{argmin}} \|M\boldsymbol{\phi}\|_\omega^2, \\ \boldsymbol{T}(\boldsymbol{\phi}^*) &= \mathbf{g}_D, \qquad \forall \mathbf{y} \in \partial\Omega_I, \end{aligned} \tag{1}$$

where $\mathbf{g}_D$ is a known and fixed Dirichlet boundary condition in the Hilbert space $\left[\mathcal{L}^2(\partial\Omega_I)\right]^n$, and the weighted norm $\|\mathbf{f}\|_\omega$ is

$$\|\mathbf{f}\|_\omega^2 = \int_{\Omega_I} \omega\ \mathbf{f}\cdot\mathbf{f}\ \mathrm{d}\Omega,$$

being ω a positive weight function. Note that $\boldsymbol{\phi}$ and $\mathbf{g}_D$ are mappings that go from the undeformed configuration to the morphed configuration. Thus, the input are the coordinates in the undeformed configuration, Ω_I, and the output are coordinates of the morphed configuration, Ω_M.

The point-wise distortion measure, $M\boldsymbol{\phi}^*$, is defined in terms of the shape distortion measure for linear simplices [10, 36] as

$$M\boldsymbol{\phi}(\mathbf{y}) = \eta(\mathbf{D}\boldsymbol{\phi}(\mathbf{y})) = \frac{|\mathbf{D}\boldsymbol{\phi}(\mathbf{y})|^2}{n\sigma(\mathbf{D}\boldsymbol{\phi}(\mathbf{y}))^{2/n}}, \tag{2}$$

where $\mathbf{D}\boldsymbol{\phi}$ is the Jacobian of $\boldsymbol{\phi}$, $\sigma(\cdot)$ is the determinant, n is the space dimension and $|\cdot| = \sqrt{(\cdot,\cdot)}$ is the Frobenius norm of matrices. The point-wise distortion measure is equal to one when the mapping $\boldsymbol{\phi}$ is locally a rotation, a translation or a scaling, and it is equal to infinity when the determinant of the mapping Jacobian is zero.

Nevertheless, the shape distortion measure presents finite values when the determinant is negative. This would lead to meshes with inverted elements. To solve this issue, we propose to regularize the shape distortion measure as

$$\eta_0(\mathbf{D}\boldsymbol{\phi}(\mathbf{y})) = \frac{|\mathbf{D}\boldsymbol{\phi}(\mathbf{y})|^2}{n\sigma_0(\mathbf{D}\boldsymbol{\phi}(\mathbf{y}))^{2/n}}, \tag{3}$$

where

$$\sigma_0(\mathbf{D}\boldsymbol{\phi}(\mathbf{y})) = \frac{1}{2}\left(\sigma + |\sigma|\right). \tag{4}$$

In this manner, when the determinant is negative, the point-wise distortion takes a value of infinity, and when the determinant is positive, the point-wise distortion takes a finite value.

3.2. Continuous augmented Lagrangian framework

In this section, we deduce an augmented Lagrangian formulation [23–25] to solve the optimization problem (1). First, we rewrite the morphing problem as the following constrained optimization

$$\begin{aligned} &\min_{\boldsymbol{\phi}\in[\mathcal{H}^1(\Omega_I)]^n} E(\boldsymbol{\phi}) = \|M\boldsymbol{\phi}\|^2_\omega \\ &\text{subject to:} \\ &\boldsymbol{T}(\boldsymbol{\phi}) = \mathbf{g}_D, \end{aligned}$$

where $\boldsymbol{T} : \left[\mathcal{H}^1(\Omega_I)\right]^n \to \left[L^2(\partial\Omega_I)\right]^n$ is the trace operator. The objective of the weight function ω is to give more importance to regions of high distortion in order to obtain a high quality mapping. In Section 4.1, we discuss how to select the weight function. We have introduced a merit function that measures the distortion of mapping $\boldsymbol{\phi}$ and thus, the constrained minimization problem consists on minimizing the mapping distortion while fixing the boundary values.

The idea of the augmented Lagrangian method is to define a new functional that takes into account the merit function (distortion) and the constraint (target boundary displacement). Then, a series of optimization problems is solved to enforce the constraint. Specifically, the new functional to minimize is

$$E_{\boldsymbol{\lambda},\mu}(\boldsymbol{\phi}) = \|M\boldsymbol{\phi}\|^2_\omega - (\boldsymbol{\lambda}, \boldsymbol{T}(\boldsymbol{\phi}) - \mathbf{g}_D)_{\partial\Omega_I} + \frac{1}{2}\mu\|\boldsymbol{T}(\boldsymbol{\phi}) - \mathbf{g}_D\|^2_{\partial\Omega_I}, \qquad (5)$$

where $\mu \geq 0$ is a real parameter, and $\boldsymbol{\lambda} \in \left[\mathcal{L}^2(\partial\Omega_I)\right]^n$ is an approximation of the Lagrange multipliers associated to the constraint. Note that $(\cdot,\cdot)_{\partial\Omega_I}$ and $\|\cdot\|_{\partial\Omega_I}$ are the scalar product and the norm of $\left[\mathcal{L}^2(\partial\Omega_I)\right]^n$, respectively, defined as:

$$(\mathbf{f},\mathbf{g})_{\partial\Omega_I} = \int_{\partial\Omega_I} \mathbf{f}\cdot\mathbf{g}\ \mathrm{d}\Gamma, \qquad \|\mathbf{f}\|_{\partial\Omega_I} = (\mathbf{f},\mathbf{f})^{1/2}_{\partial\Omega_I}.$$

The augmented Lagrangian functional in Equation (5) is composed of three terms. The first one is the original target function while the second one is related to the Lagrange multipliers. These two terms define the Lagrangian of the constrained minimization problem. Nevertheless, to solve the constrained minimization problem using the Lagrangian it is necessary to solve a non-linear saddle-point problem taking into account the original unknowns and the Lagrange multipliers. To address this issue, the augmented Lagrangian method introduces an additional penalty term. Note that the Lagrange multipliers, $\boldsymbol{\lambda}$, and the penalty, μ, are parameters of the problem, and no additional unknowns are introduced. Instead of solving one non-linear saddle-point problem with additional unknowns, we solve a series of non-linear problems with the original unknowns. During the optimization process the penalty parameter is increased in order to enforce the boundary condition, and the Lagrange multipliers are successively approximated in order to limit the growth of the penalty parameter.

Algorithm 1 details the proposed augmented Lagrangian method adapted for domain morphing with prescribed boundary condition. The input of the

Algorithm 1 Continuous augmented Lagrangian method.

Input: Ω_I, $\mathbf{g}_d$, δ^* ,ε^*, μ_0, ω_0, ε_0
Output: Mapping $\boldsymbol{\phi}^*$

function `meshOptimization`

▷ Variables initialization
$\boldsymbol{\phi}_0 \leftarrow \mathbf{Id}$
$\boldsymbol{\lambda}_0 \leftarrow \mathbf{0}$
$\omega_0 \leftarrow 1$
$m_0 \leftarrow 10$
$m_1 \leftarrow m_0$, $\mu_1 \leftarrow \mu_0$
$\delta_1 \leftarrow \delta_0$, $\varepsilon_1 \leftarrow \varepsilon_0$

▷ Augmented Lagrangian main loop
while $\|\boldsymbol{T}(\boldsymbol{\phi}_k) - \mathbf{g}_D\|_{\partial\Omega_I} > \varepsilon^*$ **and** $\|\nabla E_{\boldsymbol{\lambda}_k,\mu_k}(\boldsymbol{\phi}_k)\|_{\Omega_I} > \delta^*$ **do**

▷ Optimize Functional (5), see Section 4
$\boldsymbol{\phi}_{k+1} \leftarrow$ `optimizeFunction`$(\boldsymbol{\phi}_k, \boldsymbol{\lambda}_k, \mu_k, \omega_k, \delta_k)$

▷ Update augmented Lagrangian parameters
if $\|\boldsymbol{T}(\boldsymbol{\phi}_{k+1}) - \mathbf{g}_D\|_{\partial\Omega_I} \leq \varepsilon_k$ **then** ▷ Update Lagrange multipliers
$\boldsymbol{\lambda}_{k+1} \leftarrow \boldsymbol{\lambda}_k - \mu_k(\boldsymbol{T}(\boldsymbol{\phi}_{k+1}) - \mathbf{g}_D)$
$m_{k+1} \leftarrow m_k$
$\mu_{k+1} \leftarrow \mu_k$
$\varepsilon_{k+1} \leftarrow \varepsilon_k / m_{k+1}^{0.9}$
$\delta_{k+1} \leftarrow \delta_k / \mu_{k+1}$
else ▷ Update penalty parameter
$\boldsymbol{\lambda}_{k+1} \leftarrow \boldsymbol{\lambda}_k$
$m_{k+1} \leftarrow 100 m_k$
$\mu_{k+1} \leftarrow \mu_0 / (m_0/m_{k+1})$
$\varepsilon_{k+1} \leftarrow \varepsilon_0 / (m_0/m_{k+1})^{0.1}$
$\delta_{k+1} \leftarrow \delta_0 / (m_0/m_{k+1})$
end if
end while
end function

algorithm is an initial domain, Ω_I, the tolerances for optimization of the non-linear objective function and the boundary condition, δ^* and ε^*, respectively, and the parameters of the augmented Lagrangian method. The algorithm stops when a solution is found that satisfies

$$\|\boldsymbol{T}(\boldsymbol{\phi}_k) - \mathbf{g}_D\|_{\partial\Omega_I} \leq \varepsilon^* \qquad \text{and} \qquad \|\nabla E_{\boldsymbol{\lambda}_k,\mu_k}(\boldsymbol{\phi}_k)\|_{\Omega_I} \leq \delta^*.$$

We initialize $\boldsymbol{\phi}_0$ to the identity mapping, **Id**. Note that the identity mapping is optimal with respect to the distortion measure. However, it does not satisfy the boundary constraint. We also set the initial weight function, ω_0 to one. This weight function will be updated to take into account the elements of worse

quality. Then, we initialize the multipliers in Line 3, the penalty parameter in Line 6, and the tolerances to check the evolution of the non-linear problem and the constraint in Line 7. Lines 8–23 define the main loop of the augmented Lagrangian method. This loop is performed until the stopping criteria are satisfied. In Line 9, we optimize functional (5) using the current $\boldsymbol{\lambda}$ and μ parameters, using $\boldsymbol{\phi}_k$ as initial condition, and using the current weights, ω_k to compute the distortion measure of the mapping $\boldsymbol{\phi}_k$. In Section 4.1, we detail the algorithm to optimize functional (5), and to update the weight function. Then, we update the values of $\boldsymbol{\lambda}$ and μ according to the current value of the constraint norm. If the constraint norm is small enough, Line 10, we update the values of $\boldsymbol{\lambda}$, keep the value of μ, and tighten the tolerances. On the contrary, if the constraint norm is too high, Line 16, we increase the value of μ, keep the current value of $\boldsymbol{\lambda}$, and tighten the tolerances accordingly.

Note that the only unknown of functional (5) is the mapping $\boldsymbol{\phi}$, while $\boldsymbol{\lambda}$ and μ are parameters that are selected according to the augmented Lagrangian method, see Lines 10–22. Moreover, the weight function is updated using an explicit formula, see Section 4.1. Thus, when we apply the proposed augmented Lagrangian technique, we have not included additional unknowns to the original constrained minimization problem.

The stopping condition of the augmented Lagrangian method, Line 8, involves two norms. The first one, $\|\boldsymbol{T}(\boldsymbol{\phi}_k) - \mathbf{g}_D\|_{\partial\Omega_I}$, is an $\mathcal{L}^2$ norm of functions. The second one, $\|\nabla E_{\boldsymbol{\lambda}_k,\mu_k}(\boldsymbol{\phi}_k)\|_{\Omega_I}$, is the $\mathcal{H}^1$ norm of the $\mathcal{H}^1$ gradient of the functional $E_{\boldsymbol{\lambda}_k,\mu_k}$ evaluated at $\boldsymbol{\phi}_k$.

In Algorithm 1, the exponents 0.9 and 0.1, Lines 14 and 20, respectively, and the scaling factor in the update of m, Line 18, are chosen according to [24, 26]. In particular, these values are set up to control the increase of the penalty parameter and the decrease of the convergence tolerances for the non-linear solver.

3.3. Problem discretization

The proposed augmented Lagrangian formulation is devised to solve a continuous morphing problem by performing a series of minimization problems of the functional (5). If we want to apply this methodology to the corresponding discrete mesh morphing problem, we need to discretize the functional. To this end, we assume that the initial domain, Ω_I, is approximated by a mesh, $\mathcal{M}_I$, and that the final mesh, $\mathcal{M}_P$, is defined in terms of a homeomorphism, $\boldsymbol{\phi}_h^* \in \mathcal{U}$, where

$$\mathcal{U} = \left\{ \mathbf{u} \in \mathcal{C}^0(\mathcal{M}_I, \mathbb{R}^n) \text{ such that } \mathbf{u}_{|e_I} \in \left[\mathcal{P}^p\left(e_I\right)\right]^n \ \ \forall e_I \in \mathcal{M}_I \right\},$$

and $\mathcal{P}^p\left(e_I\right)$ is the space of polynomials of degree at most p over the element e_I. Note that we enforce the continuity of mapping $\boldsymbol{\phi}_h^*$ in order to obtain a conformal mesh. Thus, the discretized version of the augmented Lagrangian functional becomes

$$E_{\boldsymbol{\lambda}_h,\mu}(\boldsymbol{\phi}_h) = \|M\boldsymbol{\phi}_h\|^2_{\omega_h} - (\boldsymbol{\lambda}_h, \boldsymbol{T}(\boldsymbol{\phi}_h) - \mathbf{g}_D)_{\partial\mathcal{M}_I} + \frac{1}{2}\mu^2 \|\boldsymbol{T}(\boldsymbol{\phi}_h) - \mathbf{g}_D\|^2_{\partial\mathcal{M}_I}, \quad (6)$$

being the weighted norm $\|M\boldsymbol{\phi}_h\|_{\omega_h}$

$$\|M\boldsymbol{\phi}_h\|^2_{\omega_h} = \sum_{e_I \in \mathcal{M}_I} \int_{e_I} \omega_h \, (M\boldsymbol{\phi}_h)^2 \, \mathrm{d}\Omega, \tag{7}$$

where ω_h is an element-wise constant weight function, $\boldsymbol{T} : \mathcal{U} \to \mathcal{V}$ is the trace operator, $\boldsymbol{\lambda}_h \in \mathcal{V}$ is the discretization of the Lagrange multipliers associated to the constraint, and

$$\mathcal{V} = \left\{ \mathbf{v} \in \mathcal{C}^0(\partial\mathcal{M}_I, \mathbb{R}^n) \text{ such that } \mathbf{v}_{|f_I} \in [\mathcal{P}^p(f_I)]^n \ \ \forall f_I \in \partial\mathcal{M}_I \right\},$$

being $\mathcal{P}^p(f_I)$ the space of polynomials of degree at most p defined over the boundary face f_I. Note that we approximate the Lagrange multipliers using polynomials of the same degree as the physical mesh. The scalar product, $(\cdot,\cdot)_{\partial\mathcal{M}_I}$, and its associated norm, $\|\cdot\|_{\partial\mathcal{M}_I}$, are defined as

$$(\mathbf{f},\mathbf{g})_{\partial\mathcal{M}_I} = \sum_{f_I \in \partial\mathcal{M}_I} \int_{f_I} \frac{1}{h} \mathbf{f} \cdot \mathbf{g} \, \mathrm{d}\Gamma, \qquad \|\mathbf{f}\|_{\partial\mathcal{M}_I} = (\mathbf{f},\mathbf{f})^{1/2}_{\partial\mathcal{M}_I}$$

being f_I the elemental faces that are located at the boundary of the mesh, and h a measure of the boundary face size. By taking into account the element size of the boundary elements in the boundary integrals, we balance the distortion and the constraint contributions in the discretized version of the augmented Lagrangian functional. In our work, we use the in-radius as the element size.

Note that $\boldsymbol{\phi}_h$ is expressed in terms of the physical nodes as follows

$$\boldsymbol{\phi}_h = \sum_{i=1}^{n_N} \mathbf{x}_i N_i,$$

where n_N is the number of nodes in the mesh, and $\{N_i\}_{i=1,\dots,n_N}$ is a Lagrangian basis of element-wise polynomial shape functions continuous at the element interfaces. Thus, the augmented Lagrangian functional only depends on the positions of the physical nodes:

$$E_{\boldsymbol{\lambda}_h,\mu}(\boldsymbol{\phi}_h) = E_{\boldsymbol{\lambda}_h,\mu}(\mathbf{x}_1,\dots,\mathbf{x}_n).$$

For this reason, the optimization of the discrete functional in Equation (6) can be interpreted as computing the positions of the mesh nodes in the morphed configuration.

Similarly to the continuous case, the update of the Lagrange multipliers in the discrete version of the augmented Lagrangian is performed as

$$\boldsymbol{\lambda}_{h,k+1} = \boldsymbol{\lambda}_{h,k} - \mu \left(\boldsymbol{T}(\boldsymbol{\phi}_{h,k}) - \mathbf{g}_D \right). \tag{8}$$

Note that the Lagrange multipliers, $\boldsymbol{\lambda}_h$, are expressed as

$$\boldsymbol{\lambda}_h = \sum_{j=1}^{m_N} \boldsymbol{\lambda}_j N_j^b,$$

where m_N is the number of boundary nodes, and $\{N_j^b\}_{j=1,\dots,m_N}$ is a Lagrangian basis of element-wise polynomial shape functions continuous at the element interfaces, defined at the boundary of the mesh. Accordingly, the Lagrange multipliers are determined by the boundary nodal values and, can be updated by evaluating Equation (8) at each node of the boundary mesh.

In the discrete case, the stopping condition of the augmented Lagrangian method also involves two norms. Since the term $\|\nabla E_{\boldsymbol{\lambda}_k,\mu_k}(\boldsymbol{\phi}_k)\|_{\Omega_I}$ is only used in the stopping condition, we approximate it by evaluating the Euclidean norm of the residual vector. We approximate it to avoid the computation of the $\mathcal{H}^1$ gradient of the functional $E_{\boldsymbol{\lambda}_k,\mu_k}$. The term $\|\boldsymbol{T}(\boldsymbol{\phi}_k) - \mathbf{g}_D\|_{\partial\Omega_I}$ is evaluated as the $\mathcal{L}^2$ norm of functions.

4. Implementation of the Augmented Lagrangian Optimization Framework

In this section, we detail the specific implementation of the optimization framework developed in this work. First, in Section 4.1 we explain the implementation of the optimization framework, specifying the computation of the elemental weights required during the process. Second, in Section 4.2, we illustrate the features of the chosen elemental weights by optimizing an example featuring a large displacement of a sphere.

4.1. Implementation of the optimization process

In this work, the weight function ω_h in Equation (7) is discretized using an element-wise constant function. In particular, we have that

$$\|M\boldsymbol{\phi}_h\|^2_{\omega_h} = \int_{\mathcal{M}_I} \omega_h (M\boldsymbol{\phi}_h)^2 \, \mathrm{d}\Omega_I = \sum_{e\in\mathcal{M}_I} \omega^e \int_e (M\boldsymbol{\phi}_h)^2 \, \mathrm{d}\Omega_I,$$

being ω^e a constant weight over the element e. That is, the contribution of each element into the target function takes into account the elemental weight. In this manner, it is possible to increase the mesh quality by increasing the weight of the low-quality elements. Nevertheless, it is not possible to know *a priori* which elements will have the lowest quality. Thus, we propose an iterative process to estimate the value of the elemental weights. This process is performed at each step of the augmented Lagrangian method, see Line 9 in Algorithm 1. The objective is to optimize the discrete functional in Equation (6) and, at the same time, update the values of the elemental weights. To this end, we define the distortion measure of an element [37], herein named η_e, as

$$\eta_e = \frac{\sqrt{\int_e (M\boldsymbol{\phi}_h)^2 \, \mathrm{d}\Omega}}{\sqrt{\int_e 1 \, \mathrm{d}\Omega}}. \tag{9}$$

Using the elemental distortion, we update the value of the elemental weight, ω_h^e, in two steps. An initial value is computed as

$$\widehat{\omega^e} = \max\left\{\omega^e, e^{\alpha(\eta_e - 1)}\right\}, \tag{10}$$

where α is a parameter that controls the behavior of the weights. In our implementation, we use $\alpha = 3$. However, if the new values of the weights are too high, instabilities in the non-linear solver and in the augmented Lagrangian method may be introduced. Thus, we limit the increment of the elemental weights in the following manner

$$\omega^e = \min\left\{\beta\omega^e, \widehat{\omega^e}\right\}, \tag{11}$$

where, herein, we use $\beta = 1.25$. That is, we limit the increment of the elemental weights to 25%. The weights are devised in such a way that they are non-decreasing, and to take into account the evolution of the distortion of the corresponding element. If at some point of the optimization an element gets highly distorted, its associated weight will be high.

The values of the α and β parameters have been chosen empirically and according to the following reasoning. The α parameter relates the elemental weight with the element quality. As α increases, so does the elemental weight and therefore, the element quality at the end of the optimization process will be higher. However, large values of α may hamper the convergence of the non-linear solver. The β parameter limits the variation of the elemental weights in successive iterations. If the variation of the elemental weights is too large, instabilities in the non-linear solver might be introduced and therefore, the non-linear solver might diverge.

The proposed iterative process to optimize functional (6) while estimating the weight function is described in Algorithm 2. The input data is the current value of mapping $\boldsymbol{\phi}_h$, $\boldsymbol{\lambda}_h$ and μ, the tolerance of the non-linear solver, δ, and an initial value of the elemental weights, ω_h. Lines 2 to 9 define the main loop of the function. This loop is iterated until the stopping criterion is satisfied. That is, until

$$\|\nabla E_{\boldsymbol{\lambda}_h,\mu}(\boldsymbol{\phi}_h)\| < \delta.$$

In Line 3, we optimize the discretized functional using a backtracking line-search method in which the advancing direction is selected using Newton's method, and the step length is set according to Armijo rule [26]

$$E_{\boldsymbol{\lambda}_h,\mu}(\mathbf{x}_k + \alpha_k \mathbf{p}_k) \leq E_{\boldsymbol{\lambda}_h,\mu}(\mathbf{x}_k) + c\alpha_k \mathbf{p_k} \cdot \nabla E_{\boldsymbol{\lambda}_h,\mu}(\mathbf{x}_k + \alpha_k \mathbf{p}_k),$$

where α_k is the step length, $\mathbf{p}_k$ is the advancing direction and $c = 10^{-4}$, according to [26]. This rule ensures that the decrease in the objective function is proportional to the step length and the gradient of the objective function. The linear systems related to the Newton's method iterations are solved using the GMRES [38] iterative method with restart every 200 iterations and a relative tolerance of 10^{-9}. The linear systems are preconditioned using an incomplete LU decomposition [39]. In the case of linear meshes, we use one level of fill-in, ILU(1), while in the high-order case, we use two levels of fill-in, ILU(2).

Algorithm 2 OptimizeFunction

Input: $\boldsymbol{\phi}_h$, $\boldsymbol{\lambda}_h$, μ, ω_h, δ
Output: Mapping $\boldsymbol{\phi}_h$
1: **function** `optimizeFunction`

▷ Main iteration loop
2: **while** $\|\nabla E_{\boldsymbol{\lambda}_h,\mu}(\boldsymbol{\phi}_h)\| > \delta$ **do**

▷ Optimize function using Newton backtracking line-search
3: $\boldsymbol{\phi}_h \leftarrow$ `NewtonBackTracking`$(E_{\boldsymbol{\lambda}_h,\mu},\, \delta)$

▷ Update elemental weights
4: **for** $e \in \mathcal{M}_I$ **do**
5: $\eta_e \leftarrow$ `getElementDistortion`(e)
6: $\widehat{\omega^e} \leftarrow \max\{\omega^e, e^{\alpha(\eta_e - 1)}\}$
7: $\omega^e \leftarrow \min\{\beta\omega^e, \widehat{\omega^e}\}$
8: **end for**
9: **end while**
10: **return** $\boldsymbol{\phi}_h$
11: **end function**

Then, we perform a loop over the elements of the mesh in order to update the corresponding elemental weights according to the formulas presented in Equations (9), (10) and (11), see Lines 5 to 7. Since the weight function is modified at each iteration, the convergence criterion in Line 2 has to be checked until it is satisfied. That is, when the weight function is modified, the functional gradient is also modified. Thus, the process is iterated until the weight modification does not induce a *large* modification in the functional gradient.

Our method is devised to favor that at each optimization step, a valid mesh is deformed to a valid mesh. To this end, two main ingredients have been considered. First, functional (6) penalizes inverted elements by taking an infinite value. Second, we have incorporated a backtracking line search to Newton's method. Thus, if a Newton full step was deforming a valid mesh to an invalid mesh, the backtracking line-search would decrease the step length to recover a valid mesh.

4.2. Behavior of the weight function: 3D spherical domain

The objective of this example is to show the effect of the weights in the formulation in order to obtain better quality meshes. To this end, we solve a mesh morphing problem in which we apply a large displacement to an inner spherical boundary. Figure 1a shows the initial mesh, composed of 14361 tetrahedra and 2642 nodes. The inner radius is 1 unit, the outer radius is 21 units, and the applied displacement of the inner sphere is 15 units. Figure 1b shows the final mesh when the weights are considered equal to one during the whole optimization process. Note that low-quality elements are present around the displaced sphere. Figure 1c is optimized using the proposed approach in which

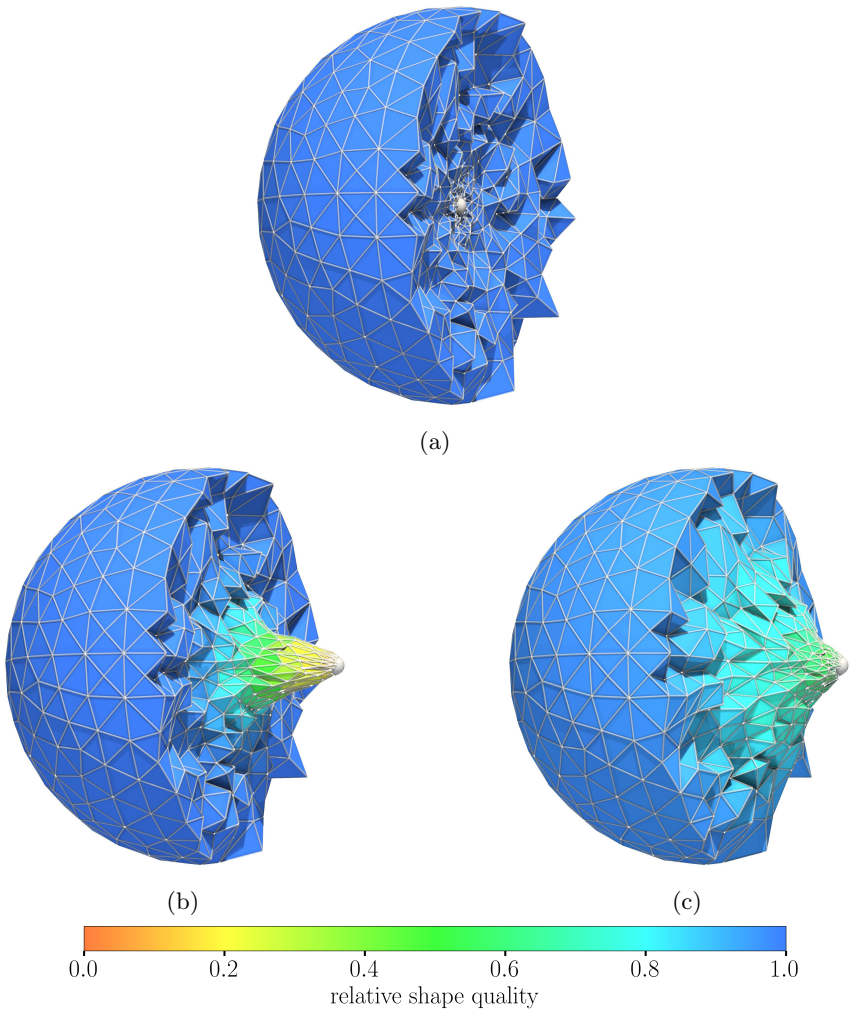


Figure 1: Cut view of the initial mesh around a sphere: (a) initial mesh; and final mesh for (b) constant and (c) non-constant weight function.

we estimate a weight function with higher values in regions in which low-quality elements are present. Note that in this case, the movement of the nodes is distributed in a larger region than in the first case. Thus, the minimum quality of the mesh is increased from 0.06 to 0.38.

Figure 2a presents the evolution of the constraint norm through the iterations of the augmented Lagrangian method. In dashed line, we depict the case when the weights are constant and equal to one, and in solid line we depict the case when using non-constant weights. Note that in both cases the constraint norm presents similar behavior. In all the iterations the constraint norm decreases

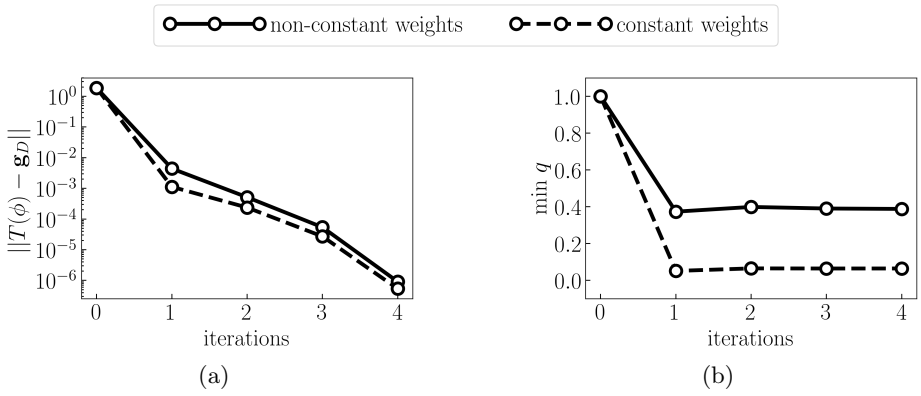


Figure 2: (a) Evolution of the constraint norm against the iterations of the augmented Lagrangian solver; and (b) evolution of the minimum element quality over the iterations of the augmented Lagrangian solver.

and in four iterations the augmented Lagrangian method converges.

Figure 2b shows the evolution of the minimum element quality over the iterations of the augmented Lagrangian method. In both cases, at the first iteration the element quality decreases and in the next iterations the minimum quality remains constant. Nevertheless, using the proposed elemental weights it is possible to increase the minimum element quality.

To obtain the final results, both cases have taken four augmented Lagrangian iterations. In the case of constant weight function, the optimization takes 27 seconds, while the case with non-constant weight function takes 44 seconds. This difference comes from the number of non-linear problems that are solved. In the case of constant weight function, each augmented Lagrangian iteration leads to a single non-linear problem. However, in the case of the non-constant weight function, twelve non-linear problems are solved in total. Although there are three times more non-linear problems, in the case of constant weight function there are 66 linear problems to be solved, while in the case of the non-constant weight function, there are 97. Note that the ratio of linear solves per second is roughly the same for the two cases.

5. Examples

This section presents several examples to show the capabilities of the presented mesh morphing method. Note that the presented examples are devised to stress the proposed augmented Lagrangian formulation. Specifically, we illustrate the intermediate boundary configurations for a two-dimensional mesh morphing problem, Sec. 5.1, an algorithmic scalability analysis of the proposed method, Sec. 5.2, a high-order mesh curving for highly stretched elements, Sec. 5.3, a large displacement of an aircraft, Sec. 5.4, and a large rotation of a propeller, Sec. 5.5.

To generate the initial meshes, we have used Pointwise [40]. The mesh morphing framework has been implemented in Python [41] using the FEniCS [42] and petsc4py [43] libraries. The optimization process has been performed in a Dell XPS 15 9560, with an Intel Core i7-7700HQ CPU at 2.80 GHz and 16 GB of RAM memory. All the examples have been executed using only one processor.

In all the examples we use the relative element quality, q_e defined in terms of the elemental distortion defined in Equation (9)

$$q_e = \frac{1}{\eta_e}.$$

The relative element quality takes values between zero for inverted elements to one for ideal elements. In all the examples we define the mesh quality as the minimum element quality.

5.1. Illustrating intermediate boundary configurations

In this example we show a two-dimensional case in which we apply a rotation of 120 degrees to the inner circles of the mesh in Figure 3a. The mesh is composed of 2466 linear triangles and 1318 nodes. We apply the proposed augmented Lagrangian formulation to perform the mesh morphing, and we show several stages of the deformation in Figures 3b, 3c and 3d. The displacement of the inner circles in the intermediate stages is automatically obtained by the method. Note that this displacement is neither a linear homotopy between the initial and the final positions, nor an incremental rotation between zero and 120 degrees. The method automatically detects the number of stages to fulfill the boundary constraint. In this case, the augmented Lagrangian solver is performed in seven iterations and takes 8.30 seconds.

Figure 4a shows the evolution of the norm constraint over the iterations of the augmented Lagrangian method. In this case the norm of the constraint monotonically decreases and in seven iterations the method converges. Figure 4b shows the evolution of the minimum element quality during the iterations of the augmented Lagrangian method. In the first iterations, the element quality diminishes in order to accommodate the large displacement of the boundary. Once the boundary condition is *sufficiently* reproduced and the boundary movement is *small*, the minimum element quality does not decrease. That is, in the last iterations the error in the boundary condition is further reduced without hampering the element quality. In the last iteration, the minimum element quality is 0.34, while the norm of the constraint is $2 \cdot 10^{-8}$.

5.2. Mesh independence of augmented Lagrangian iterations

The goal of this example is to show that the number of augmented Lagrangian iterations seems to remain almost constant for different mesh resolutions. To provide more information, we also include the number of Newton's method and GMRES iterations, and the total CPU time. Nevertheless, it is not our goal to conclude any type of scalability result for the total CPU time since

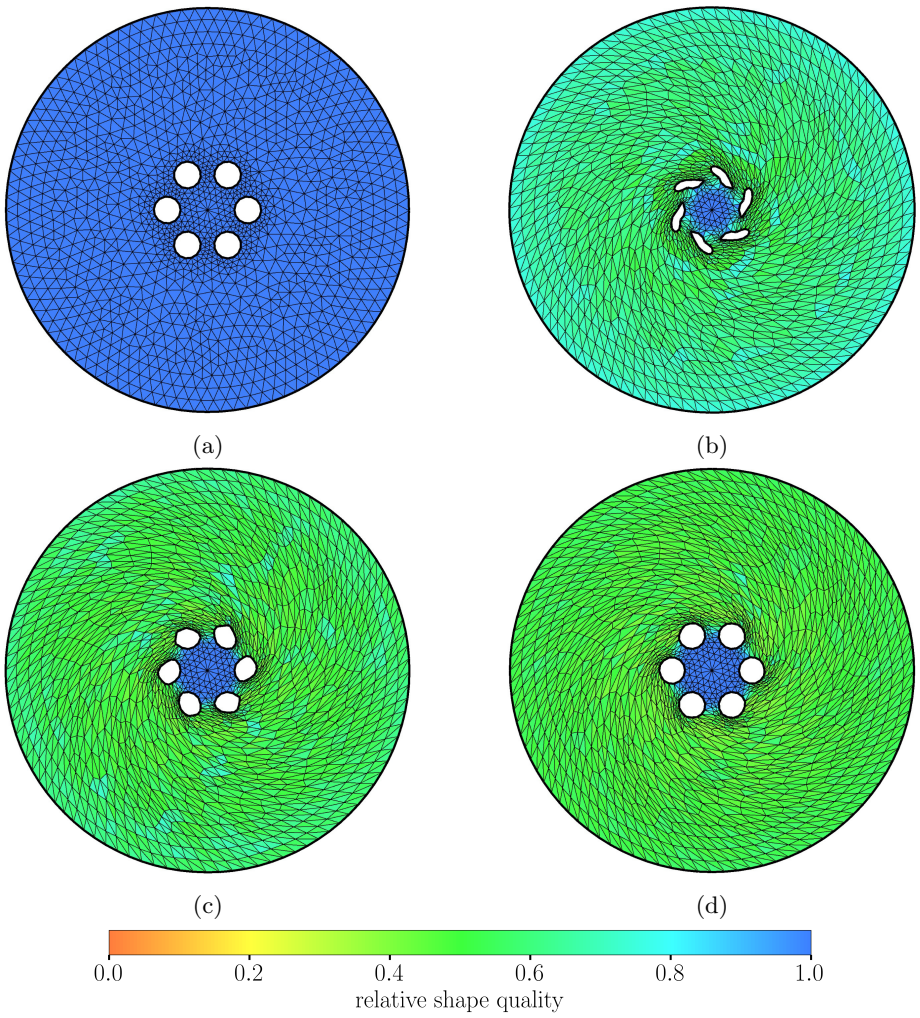


Figure 3: Initial, intermediate and final mesh configurations for a 120 degrees mesh morphing rotation. (a) First stage; (b) second stage; (c) third stage; and (d) last stage.

better pre-conditioners will need to be investigated. To this end, we generate a series of linear meshes for the exterior domain of a sphere. In each mesh, we approximately multiply by four the number of elements and the number of nodes of the previous mesh. The domain is defined by an inner sphere and an outer sphere of radius 1 and 21, respectively. The element size on the inner sphere is ten times smaller than the element size on the outer sphere. In this example, we apply a displacement of eight units to the inner sphere. The main objective is to measure several quality and computational cost indicators to analyze the scalability of proposed augmented Lagrangian technique.

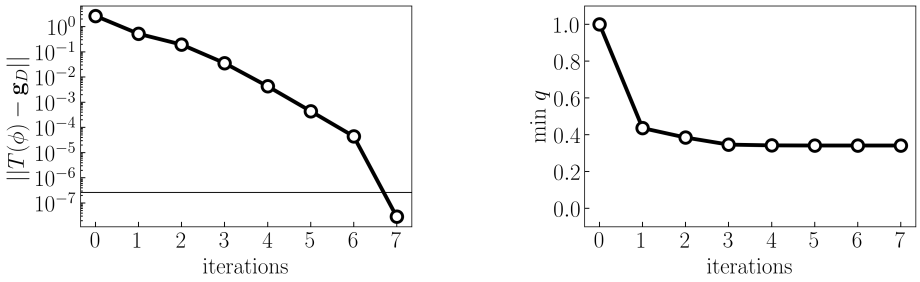


Figure 4: (a) Evolution of the constraint norm against the iterations of the augmented Lagrangian solver; and (b) evolution of the minimum element quality over the iterations of the augmented Lagrangian solver.

Table 1: Scaling results for the spherical meshes.

nodes	min quality	AL its.	non-lin solves	lin solves	GMRES its.	time (s)	time per its and nodes
3380	0.558	6	7	16	356	13.70	$1.14 \cdot 10^{-5}$
13522	0.575	6	8	16	531	60.47	$8.42 \cdot 10^{-6}$
50691	0.572	5	7	16	731	245.81	$6.63 \cdot 10^{-6}$
213340	0.559	5	7	17	1198	1129.70	$4.42 \cdot 10^{-6}$

Table 1 shows the number of elements and nodes, the minimum element quality, the number of augmented Lagrangian iterations, the number of non-linear and linear solves, the number of GMRES iterations, and the optimization time (seconds) to obtain the final mesh. We obtain similar minimum qualities, approximately 0.56, for all the meshes of the analysis. Moreover, the number of augmented Lagrangian iterations, the number of non-linear solves and the number of linear solves is roughly the same for all the test cases. Nevertheless, the number of GMRES iterations increases with the mesh size.

Figure 5 shows the distribution of linear and non-linear solves during the augmented Lagrangian iterations. Each square denotes a linear solve, corresponding to an iteration of the non-linear solver. Furthermore, the groups of linear solves within thick lines denote a non-linear solve, corresponding to the iterations that approximate the elemental weights. Finally, each iteration of the augmented Lagrangian is depicted with different colors. Note that there are augmented Lagrangian iterations in which only one non-linear solve is needed to approximate the weights, and other iterations with more than one non-linear solve. These iterations correspond to the ones in which the element quality decreases because of the current boundary displacement and thus, the elemental weights have to be increased to enhance element quality. Note that the behavior of the proposed augmented Lagrangian formulation is similar for all the presented meshes. Not only the number of linear solvers is roughly the same, but also number of linear solvers for each non-linear solver.

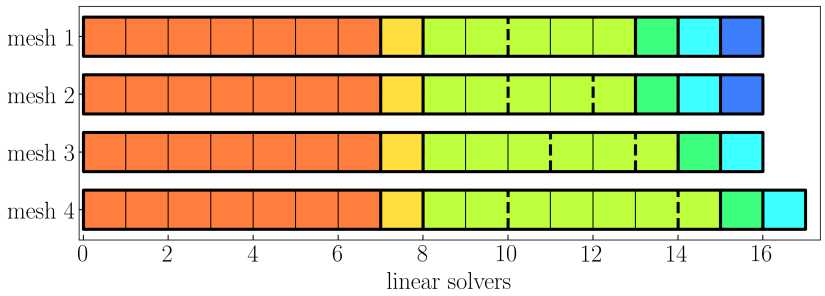
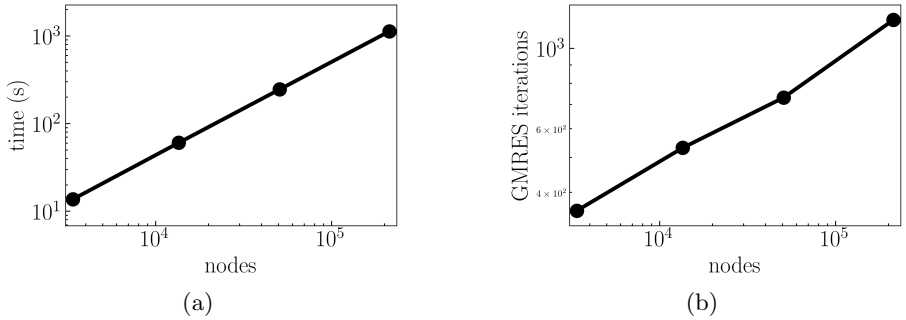


Figure 5: Distribution of linear and non-linear solves during the augmented Lagrangian iterations. Each square denotes a linear solve, corresponding to an iteration of the non-linear solver. The groups of linear solves within thick lines denote a non-linear solve, and each iteration of the augmented Lagrangian is depicted with different colors.



(a) (b)

Figure 6: (a) Computational time to solve the optimization process against the number of nodes; and (b) number of GMRES iterations against the number of nodes.

Figure 6a shows the computational time in seconds to obtain the final mesh against the number of nodes of each mesh. Note that at each step, the number of nodes is roughly increased by a factor four, and the time to obtain the final mesh is also roughly increased by a factor four. In this case, the time scaling is proportional to $n^{1.05}$. In Figure 6b, we show the GMRES iterations to obtain the final mesh against the number of nodes. The number iterations iterations to obtain the final configuration increases roughly linearly with the number of nodes. Although the number of iterations increases, by memory bandwidth density, the cost per degree of freedom is improved when bigger is the mesh until the point that the memory bandwidth is saturated.

Note that the objective of this work is not to obtain a scalable implementation of the proposed formulation. Nevertheless, the proposed augmented Lagrangian formulation leads to obtaining roughly the same number of linear systems to be solved, and a linear time scaling with the number of unknowns.

5.3. Curving highly stretched elements: sphere mesh of polynomial degree three

In this example, we show the application of the proposed method to curve a high-order mesh with highly stretched elements. To this end, we generate a

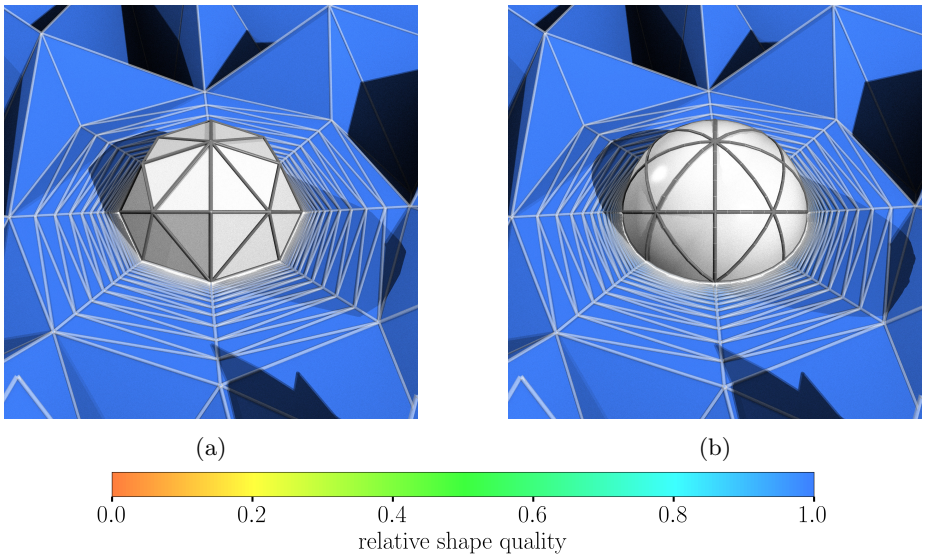


Figure 7: Cut view of the curved high-order mesh around a sphere: (a) initial straight-sided mesh; and (b) final curved high-order mesh.

tetrahedral mesh of polynomial degree three around a sphere of radius one, see Figure 7a. We generate a boundary layer composed of 40 layers of elements, a wall distance of 10^{-5}, and a growth factor of 1.3. The first layer of elements define a stretching ratio of $1 : 2 \cdot 10^5$. The whole mesh is composed of 6577 elements and 30147 nodes. Since the mesh is composed of elements of polynomial degree three, there are 20 nodes for each element. We apply the proposed augmented Lagrangian method in order to curve the initial straight-sided mesh into a curved one, see Figure 7b.

Figure 8 shows the evolution of norm of the constraint, Figure 8a, and the minimum element quality, Figure 8b, through the iterations of the augmented Lagrangian solver. The augmented Lagrangian method has converged in five iterations. After the first iteration, the norm of the constraint is of the order of 10^{-8} and the minimum element quality is 0.87. In the next iterations, the proposed augmented Lagrangian improves the element quality while the norm of the constraint oscillates since the Lagrange multipliers are being approximated. The whole process takes five iterations and, at each iteration, only one linear problem is solved. Thus, in this example we have obtained a three-dimensional curved high-order mesh with highly stretched elements by solving only five linear problems. The whole process takes around 36 minutes.

5.4. Large displacement: aircraft

In this example we apply the proposed technique to apply a mesh moving to an aircraft corresponding to the geometry provided by the 4^{th} Drag Prediction

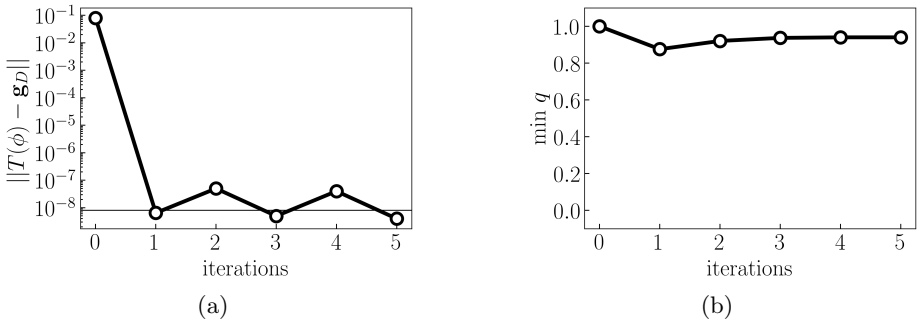


Figure 8: (a) Evolution of the constraint norm against the iterations of the augmented Lagrangian solver; and (b) evolution of the minimum element quality over the iterations of the augmented Lagrangian solver.

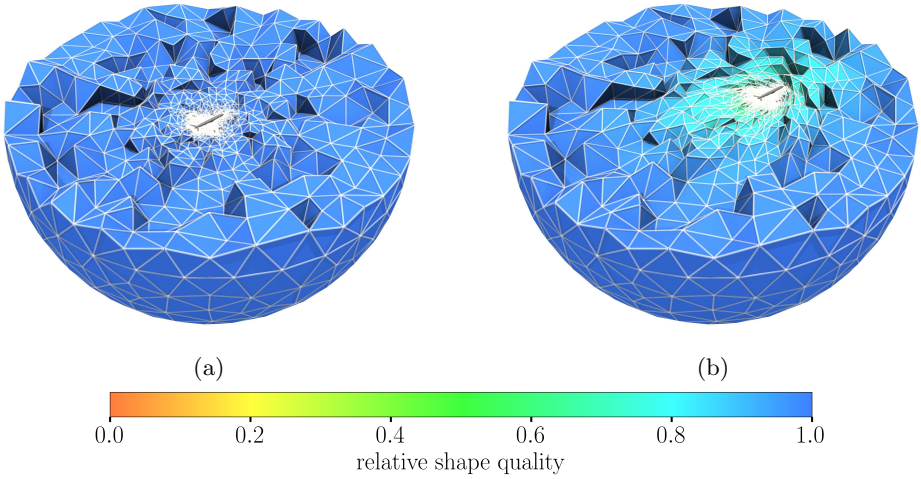


Figure 9: Cut view of the mesh around an aircraft. (a) Initial mesh and; (b) final mesh.

Workshop [44]. The length of the aircraft has been normalized to one unit, and we apply a forward displacement of two units. Figure 9a shows a global cut view of the initial mesh around the aircraft, while Figures 10a and 10c show a horizontal and longitudinal cut view around the aircraft, respectively. The mesh is composed of 153478 linear tetrahedra and 29766 nodes.

We apply the proposed optimization process to obtain the morphed mesh, see Figure 9b. This mesh is composed of valid elements in which the minimum quality is 0.33. Figures 10b and 10d show a detailed view of the aircraft mesh. Note that the elements of lower quality are located around the aircraft, in order to accommodate the large displacement of the mesh. That is, the elements in the rear part of the aircraft are elongated while the elements in the frontal part are compressed. Further away from the aircraft, there are high quality elements

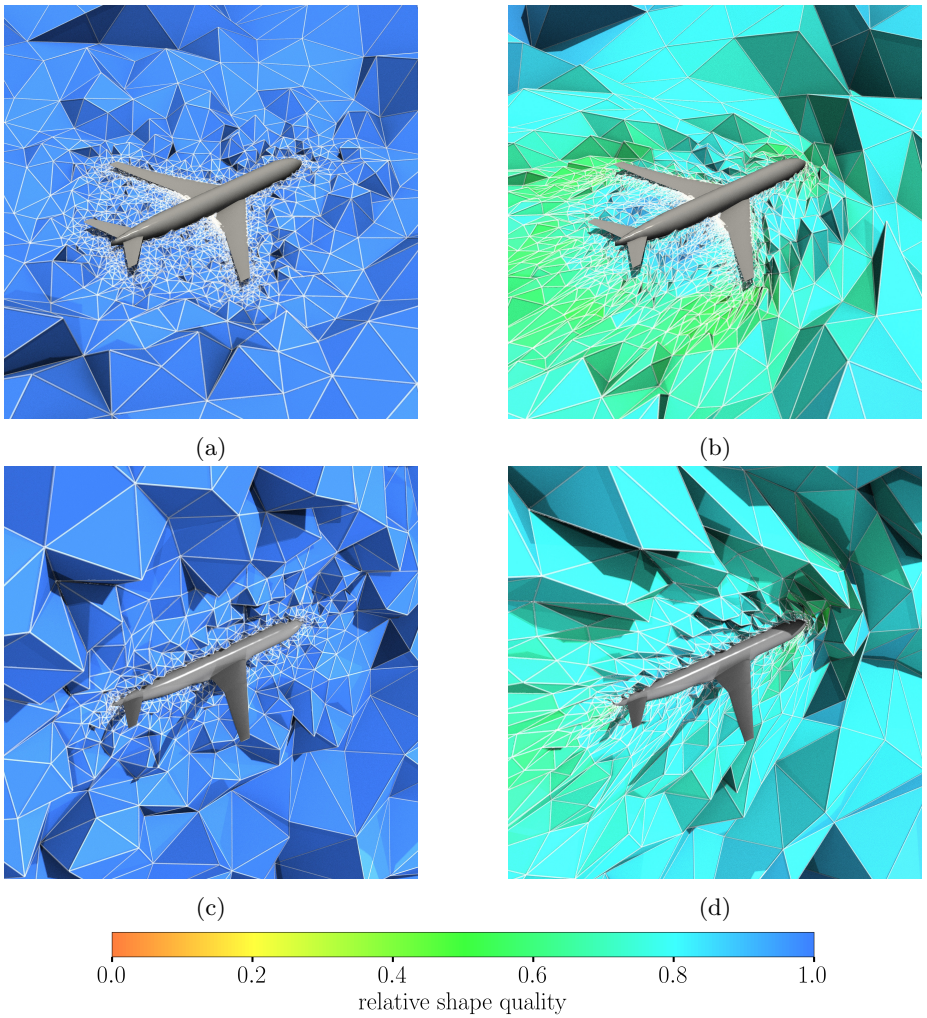


Figure 10: Detail of the cut view of the mesh around an aircraft: In rows, (a,b) horizontal and (c,d) longitudinal cut views. In columns, (a,c) initial and (b,d) final meshes.

since the nodes are almost at their initial positions.

Figure 11 shows the evolution of norm of the constraint, Figure 11a, and the minimum element quality, Figure 11b, through the iterations of the augmented Lagrangian solver. The method has converged in six iterations and, after the first iteration, the norm of the constraint is reduced to 10^{-4}, and the minimum element quality is around 0.05. During the next iterations, the proposed augmented Lagrangian method improves the minimum quality while reducing the norm of the constraint.

The whole process is performed in 60 minutes. We highlight that this ex-

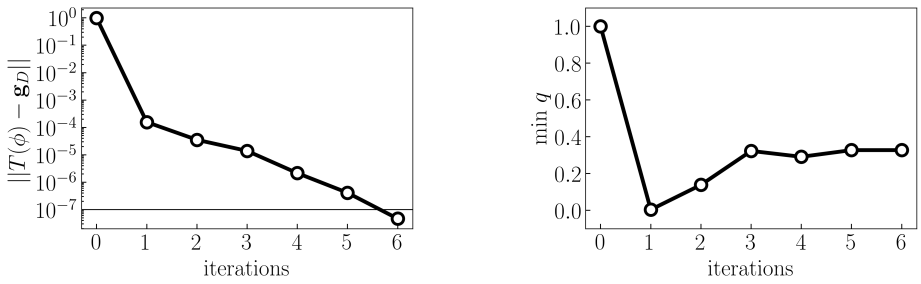


Figure 11: (a) Evolution of the constraint norm against the iterations of the augmented Lagrangian solver; and (b) evolution of the minimum element quality over the iterations of the augmented Lagrangian solver (b).

ample features a real test geometry with non-uniform element size and a large deformation compared to the size of the aircraft. The element size along the aircraft varies from 0.05 to 0.01. Note that the applied displacement is two hundred times larger than the smallest element size. Thus, the element quality may rapidly decrease as the aircraft is displaced. This leads to non-linear problems that require numerous iterations to be solved.

5.5. Large rotation: propeller

In this example, we compare the results obtained by applying our method using successive small rotations and only one large rotation. To this end, we first rotate the mesh around a propeller, see Figures 12a and 12c, seventy-two degrees in increments of one degree. Note that this angle corresponds to the angle between two consecutive blades of the propeller. At each increment, we solve the non-linear problem using the proposed augmented Lagrangian method. Then, we compare the results by imposing one large rotation to the whole mesh. The mesh is composed of 180167 linear tetrahedra and 32868 nodes.

At each step, the augmented Lagrangian method is initiated using the optimized mesh of the previous step. Nevertheless, note that the mesh quality is computed taking into account the initial mesh. Similarly, the Lagrange multipliers and the penalty parameter are initialized using the values of the previous step. Figures 12b and 12d show the mesh at the end of the displacement process.

In Figure 13, we show the whole rotation process in increments of six degrees. Note that the mesh accommodates the movement of the propeller in order to obtain a valid mesh composed of high-quality elements. As the propeller rotates, it drags the elements at the outer part of the mesh in order to obtain a valid mesh.

Figure 14 shows the evolution of the minimum quality element along the rotation process. At the first iteration, the minimum quality is one, and it starts to decrease as the propeller rotates. Although in the first iterations the mesh quality rapidly decreases, in the rest of the morphing process the minimum

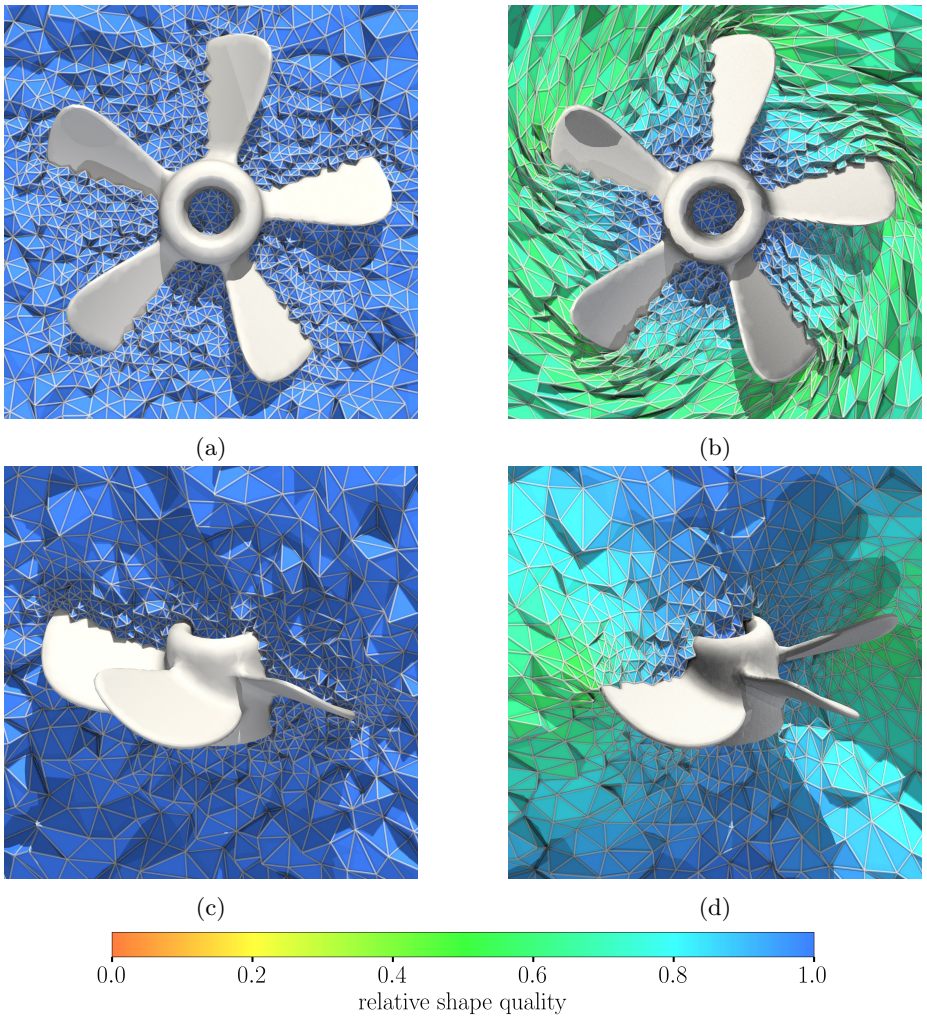


Figure 12: Cut view of the mesh around a propeller: In rows, (a,b) horizontal and (c,d) longitudinal cut views. In columns, (a,c) initial and (b,d) final meshes.

quality roughly remains constant. At the end of the morphing process, the worst element has a quality of 0.40.

Note that in this example we have applied the rotation of the propeller in small steps. Nevertheless, we have also morphed the mesh from the starting position to the final one in one step. Thus, we apply the proposed formulation to rotate the mesh around a propeller 72 degrees. The whole process takes four augmented Lagrangian iterations, and nine non-linear solves. The final configuration is computed in 636 seconds. Figure 15 shows the final configuration of the propeller by evolving the rotation in increments of one degree (15a), and

by directly imposing the final configuration (15b). Note that both configurations present similar results of the final mesh. The maximum difference between corresponding nodes is 3.7 units, and the whole geometry measures 400 units. Thus, the relative error between the two meshes is less than 1%. Furthermore, the minimum element quality of both meshes is similar. In the case of substepping, the minimum quality is 0.4, while in the case of directly imposing the final configuration, the minimum quality is 0.39.

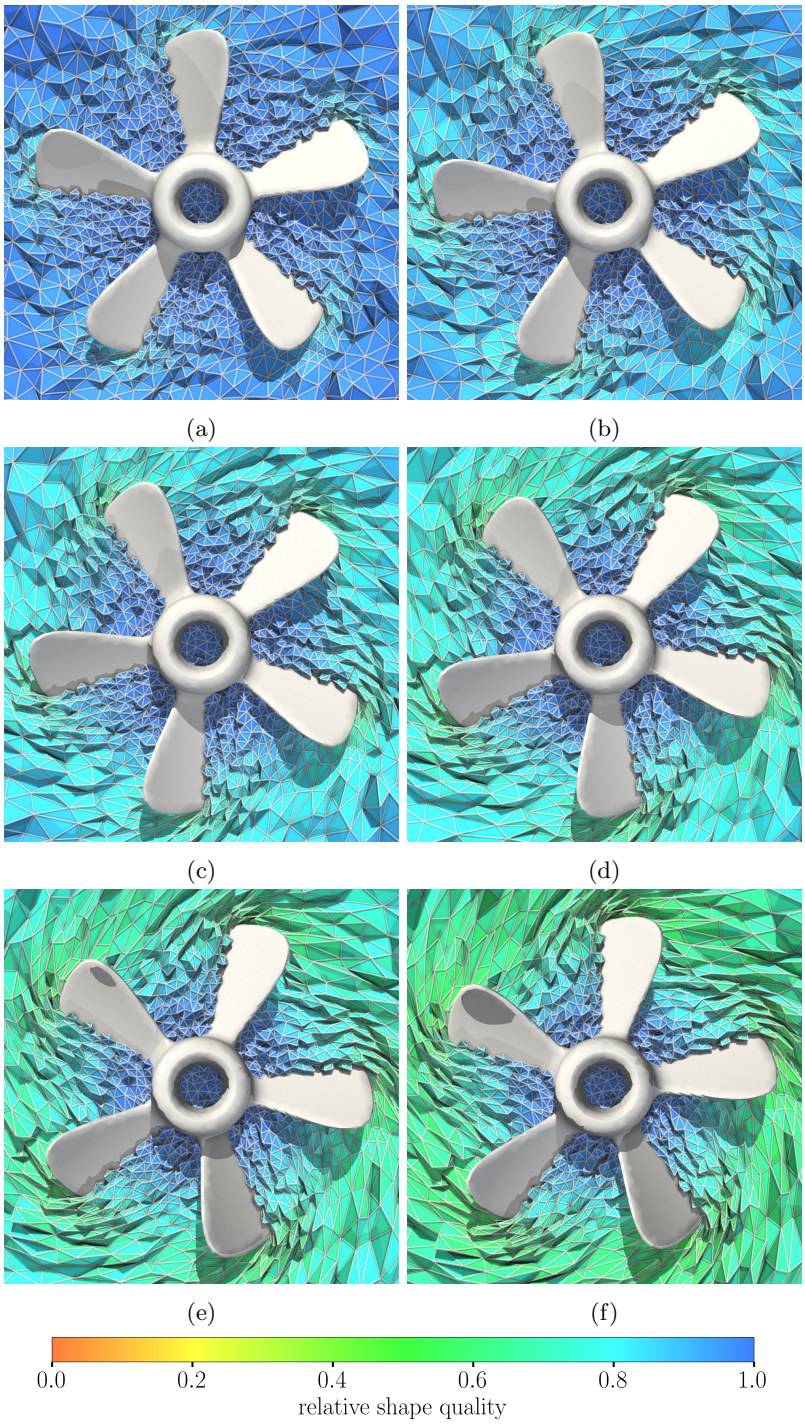


Figure 13: Intermediate rotation steps at increments of twelve degrees: (a) 12 degrees, (b) 24 degrees, (c) 36 degrees, (d) 48 degrees, (e) 60 degrees; and (e) 72 degrees.

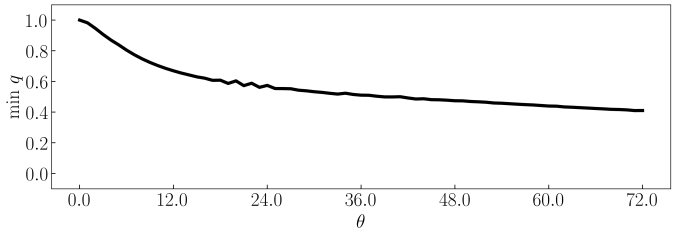


Figure 14: Evolution of the minimum element quality against the rotation angle.

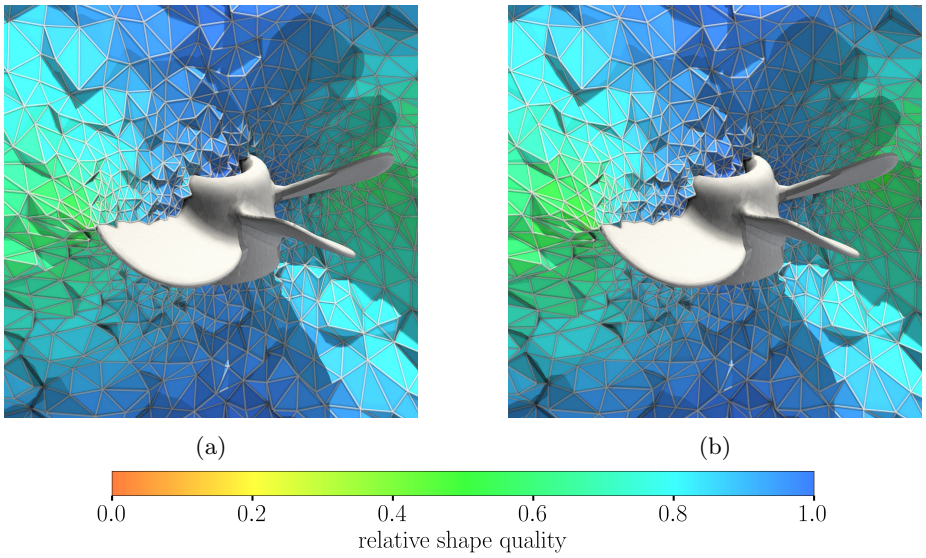


Figure 15: Comparison of the final configurations of the propeller by: (a) evolving the rotation in steps of one degree; and (b) directly imposing the final position.

6. Concluding Remarks and Future Work

In this work, we have proposed a new mesh morphing method by discretizing a constrained optimization problem solved with an augmented Lagrangian formulation in Hilbert spaces. We introduce the boundary constraint into the target functional in a weak sense, and the boundary condition is imposed automatically and gradually during the optimization process. We do not allow invalid mappings during the optimization process and therefore, the method does not need untangling capabilities. Thus, at each stage, we transform a domain to another domain using an invertible mapping.

In order to distribute the distortion along the domain, we introduce a weight function in the target functional. The main idea is to assign a higher weight to areas with high distortion in the optimization process. In this manner, we obtain higher quality mappings. Nevertheless, we need to introduce an iterative process in order to estimate the weight function, since it is not known *a priori*. In the proposed formulation, we explicitly target the volume mesh quality, and we penalize inverted elements in order to have a valid mesh during the whole optimization process. Thus, the boundary condition does not need to define a valid boundary mesh for the process to obtain a valid final mesh. Nevertheless, in practical applications, since we update the boundary condition after each penalty iteration, at some point of the optimization process the boundary condition becomes valid. Therefore, the proposed method can be interpreted as an incremental method in which the intermediate boundary positions are automatically computed without the need to define valid boundary meshes.

The full optimization algorithm is composed of three nested loops. The outer loop defines the augmented Lagrangian method. In this loop, we update the values of the Lagrange multipliers and the penalty parameter. In the middle loop, we estimate the weight function in order to obtain high-quality mappings during the optimization process. Finally, the inner loop defines the backtracking line-search non-linear solver, in which a series of linear problems are solved.

The discretization of the continuous problem defines a mesh morphing framework with applications to mesh moving and high-order mesh curving. In the examples, we have shown that the method handles three-dimensional cases with non-uniform element size, large deformations, and high-order mesh curving with highly stretched elements. Moreover, we have also shown that the number of iterations to converge the augmented Lagrangian method does roughly not depend on the element size.

We have shown in the examples that the constraint is reproduced with high accuracy, and the nodes can be as close to the surface as prescribed through the tolerance of the boundary condition. If desired, it is possible to *project* the nodes to the target positions, although the error of reproducing the constraint may increase. That is, interpolative conditions feature zero error at the interpolation nodes, and non-zero residual at intermediate points.

Several examples have been performed to analyze the applicability and the features of the proposed morphing approach. First, we have illustrated the intermediate boundary configurations for a two-dimensional mesh morphing

problem. Then, we have performed an algorithmic scalability of the proposed method. Following, we have used the technique to curve a high-order mesh of polynomial degree 3 on a sphere with elements with high stretching close to the geometry. Finally, we have shown large deformations of an aircraft and a propeller.

The main motivation of setting the augmented Lagrangian method in a Hilbert space and then discretize the resulting minimization problem is to mitigate the influence of the mesh resolution in the number of required non-linear solves. Note that the number of iterations to solve the continuous problem is determined by the *difficulty* of the problem, the initial μ and its scaling factor. By solving the discretized version of the continuous problem we expect a similar number of iterations of the augmented Lagrangian method since we are approximating the solution of the continuous problem while measuring all the terms with the adequate inner products and norms. Although we do not provide theoretical results on this reasoning, our numerical experiments on mesh independence seem to support this claim.

Alternatively, we could have discretized Equation (1) to obtain a constrained optimization problem defined in $\mathbb{R}^n$, and then we would apply the augmented Lagrangian method to this constrained optimization problem. Nevertheless, in this manner, instead of considering the same inner products and norms of the continuous problem, we would have used the discrete Euclidean scalar product and norm in $\mathbb{R}^n$. The differences of the inner products and norms would lead to a different discrete problem where the number of non-linear solves would increase with the mesh resolution.

In hyperelastic and optimization-based mesh curving frameworks, it is interesting to ensure the existence of minimizers of the target functional. With this objective in mind, it is sufficient to show that the target functional is polyconvex. That is, the target functional can be written using a convex function of the sub-determinants of the input matrix, see [7, 9]. Note that this is the case for our continuous functional and therefore, we ensure that the there exists a minimizer of the functional. Not that the existence of minimizers does not imply that the final mesh is valid if the mesh distortion allows inverted configurations. To enforce feasible configurations, we regularize the shape distortion in such a manner that is not modified for positive determinants of the Jacobian, and leads to unbounded values for non-positive determinants of the Jacobian. Note that a full step of Newtons method could lead to an invalid configuration with unbounded distortion and therefore, to enforce feasibility we equip the non-linear solver with a backtracking line search procedure.

In all the examples, we have scaled the penalty parameter in each iteration with a small enough factor to properly ensure the continuation of the boundary condition. This enhances the convergence of the augmented Lagrangian solver. If we were using larger scaling factors we would increase the number of non-linear solver iterations and even the problem would not converge. We have equipped the non-linear solver with a backtracking line search procedure to improve the global convergence of the solver. Actually, this non-linear solver has converged all the examples considered in this study. However, if the back-

tracking line search procedure was disabled we would not able to converge some of the examples. Although, we could have used a trust region method, we preferred the simplicity of the backtracking line-search approach. Nevertheless, it might be interesting to perform a comparison between both globalization methods to contrast their robustness and computational cost for the mesh morphing problem.

To avoid poor conditioning of the related linear systems, it is required to start with a large enough penalty parameter to sufficiently enforce the boundary displacement and small enough to avoid large matrix entries. Note that when higher the polynomial degree and the mesh resolution are, larger is the condition number. In practice, we avoid these issues by pre-conditioning with incomplete LU factorizations with one or two levels of fill-in. As we have seen in the mesh independence example, the number of GMRES iterations to solve a mesh morphing problem increases with the mesh resolution. To improve the scaling of the linear solver, we will investigate alternative combinations of iterative solvers and pre-conditioners. Moreover, in the near future, we consider parallelizing the code in order to improve the computational efficiency of the proposed method, and be able to apply it to more computationally demanding mesh moving and curving problems.

The main objective of this work is to propose a new mesh morphing and mesh curving method suitable for low- and high-order meshes. The main difference with other existing high-order mesh morphing and mesh curving methods is that the proposed method automatically computes always valid intermediate configurations. Thus, in our approach it is not necessary to include an initial untangling stage to ensure valid high-order meshes. Nevertheless, other mesh morphing and curving methods based on functional minimization might use a similar augmented Lagrangian formulation to avoid the untangling stage. In the future, it would be interesting to perform a comparison between different mesh volume functionals using a similar augmented Lagrangian formulation.

Acknowledgments

This project has received funding from the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme under grant agreement No 715546. This work has also received funding from the Generalitat de Catalunya under grant number 2017 SGR 1731. The work of the corresponding author has been partially supported by the Spanish Ministerio de Economía y Competitividad under the personal grant agreement RYC-2015-01633. The work of the third author has been supported by the Spanish Ministerio de Economía y Competitividad under grant agreement CTM2014-55014-C3-3-R, together with Generalitat de Catalunya under grant number 2017 SGR 1278.

References

[1] C. J. Budd, W. Huang, R. D. Russell, Adaptivity with moving grids, Acta Numer. 18 (2009) 111–241.

[2] M. L. Staten, S. J. Owen, S. M. Shontz, A. G. Salinger, T. S. Coffey, A comparison of mesh morphing methods for 3d shape optimization, in: Proceedings of the 20th International Meshing Roundtable, Springer, 2011, pp. 293–311.

[3] V. Garanzha, I. Kaporin, Regularization of the barrier variational method, Comp Math Math Phys 39 (9) (1999) 1426–1440.

[4] S. Sherwin, J. Peiró, Mesh generation in curvilinear domains using high-order elements, Int. J. Numer. Meth. Eng. 53 (1) (2002) 207–223.

[5] P.-O. Persson, J. Peraire, Curved mesh generation and mesh refinement using Lagrangian solid mechanics, in: 47th AIAA Aerospace Sciences Meeting including The New Horizons Forum and Aerospace Exposition, 2009, p. 949.

[6] Z. Xie, R. Sevilla, O. Hassan, K. Morgan, The generation of arbitrary order curved meshes for 3D finite element analysis, Comput. Mech. 51 (2012) 361–374.

[7] V. A. Garanzha, L. Kudryavtseva, S. V. Utyuzhnikov, Variational method for untangling and optimization of spatial meshes, J. Comput. Appl. Math. 269 (2014) 24–41.

[8] S. M. Shontz, S. A. Vavasis, A robust solution procedure for hyperelastic solids with large boundary deformation, Engineering with Computers 28 (2) (2012) 135–147.

[9] M. Turner, J. Peiró, D. Moxey, Curvilinear mesh generation using a variational framework, Computer-Aided Design 103 (2018) 73–91.

[10] P. M. Knupp, Algebraic mesh quality metrics, SIAM J. Numer. Anal. 23 (1) (2001) 193–218.

[11] J. M. Escobar, E. Rodríguez, R. Montenegro, G. Montero, J. M. González-Yuste, Simultaneous untangling and smoothing of tetrahedral meshes, Comput. Meth. Appl. Mech. Eng. 192 (25) (2003) 2775–2787.

[12] C. Schüller, L. Kavan, D. Panozzo, O. Sorkine-Hornung, Locally injective mappings, in: Comput. Graph. Forum., Vol. 32, Wiley Online Library, 2013, pp. 125–135.

[13] S. Z. Kovalsky, N. Aigerman, R. Basri, Y. Lipman, Large-scale bounded distortion mappings, ACM Trans. Graph. 34 (6) (2015) 191–1.

[14] A. Gargallo-Peiró, X. Roca, J. Peraire, J. Sarrate, Distortion and quality measures for validating and generating high-order tetrahedral meshes, Eng. Comput. 31 (3) (2015) 423–437.

[15] E. Ruiz-Gironés, X. Roca, J. Sarrate, High-order mesh curving by distortion minimization with boundary nodes free to slide on a 3D CAD representation, Comput. Aided Design 72 (2016) 52–64.

[16] T. Toulorge, C. Geuzaine, J.-F. Remacle, J. Lambrechts, Robust untangling of curvilinear meshes, J. Comput. Phys. 254 (2013) 8 – 26.

[17] T. Toulorge, J. Lambrechts, J.-F. Remacle, Optimizing the geometrical accuracy of curvilinear meshes, J. Comput. Phys. 310 (2016) 361–380.

[18] M. Stees, S. M. Shontz, A high-order log barrier-based mesh generation and warping method, Procedia Engineering 203 (2017) 180–192.

[19] D. P. Sanjaya, K. J. Fidkowski, Improving high-order finite element approximation through geometrical warping, AIAA Journal (2016) 3994–4010.

[20] M. Fortunato, P. Persson, High-order unstructured curved mesh generation using the winslow equations, J. Comput. Phys. 307 (2016) 1–14.

[21] D. Moxey, D. Ekelschot, Ü. Keskin, S. Sherwin, J. Peiró, High-order curvilinear meshing using a thermo-elastic analogy, Comput. Aided Design 72 (2016) 130–139.

[22] M. Rabinovich, R. Poranne, D. Panozzo, O. Sorkine-Hornung, Scalable locally injective mappings, ACM T. Graphic. 36 (2) (2017) 16.

[23] K. Ito, K. Kunisch, Augmented Lagrangian-SQP-methods in Hilbert spaces and application to control in the coefficients problems, SIAM J. Optiz. 6 (1) (1996) 96–125.

[24] J. Maruhn, An augmented Lagrangian algorithm for optimization with equality constraints in Hilbert spaces, Master's thesis, Virginia Polytechnic Institute and State University (2001).

[25] H. Attouch, M. Soueycatt, Augmented Lagrangian and proximal alternating direction methods of multipliers in Hilbert spaces. Applications to games, PDE's and control, Pac. J Optim. 5 (1) (2008) 17–37.

[26] J. Nocedal, S. Wright, Numerical Optimization, Second Edition, Springer Verlag, 2006.

[27] E. Ruiz-Gironés, A. Gargallo-Peiró, J. Sarrate, X. Roca, An augmented Lagrangian formulation to impose boundary conditions for distortion based mesh moving and curving, Procedia Engineer. 203 (2017) 362–374.

[28] A. Gargallo-Peiró, X. Roca, J. Peraire, J. Sarrate, Optimization of a regularized distortion measure to generate curved high-order unstructured tetrahedral meshes, Int. J. Numer. Meth. Eng. 103 (5) (2015) 342–363. doi:10.1002/nme.4888.
URL http://dx.doi.org/10.1002/nme.4888

[29] A. Kelly, L. Kaczmarczyk, C. Pearce, Mesh Improvement Methodology for 3D Volumes with Non-Planar Surfaces, in: Proceedings of the 21st International Meshing Roundtable, 2011.

[30] L. Liu, Y. Zhang, T. J. R. Hughes, M. A. Scott, T. W. Sederberg, Volumetric t-spline construction using boolean operations, Eng. Comput. (2013) 1–15doi:10.1007/s00366-013-0346-6.

[31] A. Gargallo-Peiró, X. Roca, J. Peraire, J. Sarrate, A distortion measure to validate and generate curved high-order meshes on CAD surfaces with independence of parameterization, Int. J. Numer. Meth. Eng. 106 (13) (2016) 1100–1130.

[32] E. Ruiz-Gironés, J. Sarrate, X. Roca, Defining an $\mathcal{L}_2$-disparity measure to check and improve the geometric accuracy of non-interpolating curved high-order meshes, Procedia Engineer. 124 (2015) 122–134.

[33] E. Ruiz-Gironés, J. Sarrate, X. Roca, Generation of curved high-order meshes with optimal quality and geometric accuracy, Procedia Engineer. 163 (2016) 315–327.

[34] R. Poya, R. Sevilla, A. J. Gil, A unified approach for *a posteriori* high-order curved mesh generation using solid mechanics, Comput. Mech. 58 (3) (2016) 457–490.

[35] L. Liu, C. L. Tai, Z. Ji, G. Wang, Non-iterative approach for global mesh optimization, Comput. Aided Design 39 (9) (2007) 772–782.

[36] P. M. Knupp, Algebraic mesh quality metrics for unstructured initial meshes, Finite Elem. Anal. Des. 39 (3) (2003) 217–241.

[37] X. Roca, A. Gargallo-Peiró, J. Sarrate, Defining quality measures for high-order planar triangles and curved mesh generation, in: Proceedings of the 20th International Meshing Roundtable, Springer International Publishing, 2012, pp. 365–383.

[38] Y. Saad, M. H. Schultz, GMRES: A generalized minimal residual algorithm for solving nonsymmetric linear systems, SIAM Journal on scientific and statistical computing 7 (3) (1986) 856–869.

[39] Y. Saad, Iterative methods for sparse linear systems, Vol. 82, siam, 2003.

[40] Pointwise Inc., Mesh Generation Software for CFD — Pointwise, Inc., http://www.pointwise.com.

[41] Python Software Foundation, Python, http://www.python.org.

[42] M. S. Alnæs, J. Blechta, J. Hake, A. Johansson, B. Kehlet, A. Logg, C. Richardson, J. Ring, M. E. Rognes, G. N. Wells, The FEniCS Project Version 1.5, Archive of Numerical Software 3 (100). doi:10.11588/ans.2015.100.20553.

[43] PETSc for Python (2018).
URL https://bitbucket.org/petsc/petsc4py

[44] 4th AIAA CFD Drag Prediction Workshop (2011).
URL https://aiaa-dpw.larc.nasa.gov/Workshop4/workshop4.html