Implicit High-Order Meshing using Boundary and Interface Fitting

Jorge-Luis Barrera¹, Tzanio Kolev¹, Ketan Mittal¹,², Vladimir Tomov¹

Abstract

We propose a method for implicit high-order meshing that aligns easy-to-generate meshes with the boundaries and interfaces of the domain of interest. Our focus is particularly on the case when the target surface is prescribed as the zero isocontour of a smooth discrete function. Common examples of this scenario include using level set functions to represent material interfaces in multimaterial configurations, and evolving geometries in shape and topology optimization. The proposed method formulates the mesh optimization problem as a variational minimization of the sum of a chosen mesh-quality metric using the Target-Matrix Optimization Paradigm (TMOP) and a penalty term that weakly forces the selected faces of the mesh to align with the target surface. The distinct features of the method are use of a source mesh to represent the level set function with sufficient accuracy, and adaptive strategies for setting the penalization weight and selecting the faces of the mesh to be fit to the target isocontour of the level set field. We demonstrate that the proposed method is robust for generating boundary- and interface-fitted meshes for curvilinear domains using different element types in 2D and 3D.

Keywords: High-order, Implicit meshing, Mesh morphing, r-adaptivity, Finite elements, TMOP

1. Introduction

High-order finite element (FE) methods are becoming increasingly relevant in computational science and engineering disciplines due to their potential for better simulation accuracy and favorable scaling on modern architectures [1] [2] [3] [4]. A vital component of these

¹Performed under the auspices of the U.S. Department of Energy under Contract DE-AC52-07NA27344 (LLNL-JRNL-837413)
²Lawrence Livermore National Laboratory, 7000 East Avenue, Livermore, CA 94550
²Corresponding author, mittal3@llnl.gov
methods is high-order computational meshes for discretizing the geometry. Such meshes are essential for achieving optimal convergence rates on domains with curved boundaries/interfaces, symmetry preservation, and alignment with the key features of the flow in moving mesh simulations [5, 6, 7].

To fully benefit from high-order geometry representation, one must control the quality of a high-order mesh while avoiding geometrical approximation errors. However, constructing an optimal high-order mesh is not trivial and can often become a bottleneck for curvilinear domains. This issue is further exacerbated in time-dependent problems when remeshing might be needed. The goal of the current work is to address this issue using implicit meshing where we take easy-to-generate meshes and align them to the target interface and/or boundary with minimal user intervention. In particular, our focus is on the case when the target surface is prescribed implicitly by a smooth discrete function without an analytic parameterization. Common examples of this scenario include use of level set functions to represent curvilinear domains as a combination of geometric primitives in Constructive Solid Geometry (CSG) [8], material interfaces in multimaterial configurations [9], and evolving parameterized geometries in shape and topology optimization applications [10, 11, 12], amongst other applications.

Boundary conforming high-order meshes are typically generated by starting with a conforming linear mesh that is projected to a higher order space before the mesh faces are curved to fit the boundary [13, 14, 15, 16, 17, 18, 19]. Some notable approaches for directly generating high-order boundary fitted meshes are Rangarajan’s method based on Universal meshes [20] and the Vorocrust method for polyhedral meshes [21]. We also note Mittal’s distance function-based approach for approximate tangential relaxation [22] and the DistMesh algorithm for meshing geometries specified by implicit functions [23].

For generating interface fitted meshes, existing methods mainly rely on topological operations where the input mesh is split across the interface to generate an interface conforming mesh [24, 25]. Some exceptions are Ohtake’s method for adapting linear triangular surface meshes to align with domains with sharp features [26] and Le Goff’s method for aligning meshes to interfaces prescribed implicitly using volume fractions [27]. Barring [23, 20, 21, 26, 27], existing methods mainly rely on an initial conforming meshing for bound-
ary fitting and on topological operations such as splitting for interface fitting.

In contrast to existing work, we propose a high-order boundary and interface fitting method that is predominantly algebraic and seldom requires topological operations, extends to different element types (quadrilaterals/triangles/hexahedrons/tetrahedrons) in 2D and 3D, and works for implicit parameterization of the target surface using discrete finite element (FE) functions. We formulate the implicit meshing challenge as a mesh optimization problem where the objective function is the sum of a chosen mesh-quality metric using the Target-Matrix Optimization Paradigm (TMOP) \cite{[28, 29]} and a penalty term that weakly forces selected faces of a mesh to align with the target surface prescribed as the zero level set of a discrete function. Additionally, we use an adaptive strategy to mark the faces for alignment and set the penalization weight, to ensure robustness of the method for nontrivial curvilinear boundaries/interfaces. We also introduce the notion of a source mesh that can be used to accurately represent the level set and its gradient with sufficient accuracy. This mesh is decoupled from the mesh being optimized, which allows to represent the domain of interest with higher level of detail. This approach is crucial for cases where the target boundary is beyond the domain of the mesh being optimized or the input mesh does not have sufficient resolution around the zero level set.

The remainder of the paper is organized as follows. In Section 2 we review the basic TMOP components and our framework to represent and optimize high-order meshes. The technical details of the proposed method for surface fitting and tangential relaxation are described in Section 3. Section 4 presents several academic tests that demonstrate the main features of the methods, followed by conclusions and direction for future work in Section 5.

2. Preliminaries

In this section, we describe the key notation and our prior work that is relevant for understanding our newly developed boundary- and interface-fitting method.

2.1. Discrete Mesh Representation

In our finite element based framework, the domain $\Omega \in \mathbb{R}^d$, $d = \{2, 3\}$, is discretized as a union of $N_E$ curved mesh elements, each of order $p$. To obtain a discrete representation
of these elements, we select a set of scalar basis functions \( \{ \bar{w}_i \}_{i=1}^{N_p} \), on the reference element \( \bar{E} \). For example, for tensor-based elements (quadrilaterals in 2D, hexahedra in 3D), we have \( N_p = (p + 1)^d \), and the basis spans the space of all polynomials of degree at most \( p \) in each variable, denoted by \( Q_p \). These \( p \)th-order basis functions are typically chosen to be Lagrange interpolation polynomials at the Gauss-Lobatto nodes of the reference element.

The position of an element \( E \) in the mesh \( \mathcal{M} \) is fully described by a matrix \( \mathbf{x}_E \) of size \( d \times N_p \) whose columns represent the coordinates of the element control points (also known as nodes or element degrees of freedom). Given \( \mathbf{x}_E \) and the positions \( \bar{x} \) of the reference element \( \bar{E} \), we introduce the map \( \Phi_E : \bar{E} \rightarrow \mathbb{R}^d \) whose image is the geometry of the physical element \( E \):

\[
x(\bar{x}) = \Phi_E(\bar{x}) \equiv \sum_{i=1}^{N_p} \mathbf{x}_{E,i} \bar{w}_i(\bar{x}), \quad \bar{x} \in \bar{E}, \ x = x(\bar{x}) \in E,
\]

where \( \mathbf{x}_{E,i} \) denotes the \( i \)-th column of \( \mathbf{x}_E \), i.e., the \( i \)-th node of element \( E \). To represent the whole mesh, the coordinates of the control points of every element are arranged in a global vector \( \mathbf{x} \) of size \( d \times N_x \) that stores the coordinates of all node positions and ensures continuity at the faces of all elements. Here \( N_x \) is the global number of control points for a principal direction. Throughout the manuscript \( x \) will denote the position function defined by (1), while bold \( \mathbf{x} \) will denote the global vector of all node coordinates.

### 2.2. Geometric Optimization and Simulation-Based \( r \)-adaptivity with TMOP

The input of TMOP is the user-specified transformation matrix \( W \), from reference-space \( \bar{E} \) to target element \( E_t \), which represents the ideal geometric properties desired for every mesh point. Note that after discretization, there will be multiple input transformation matrices \( W \) – one for every quadrature point in every mesh element. The construction of this transformation is guided by the fact that any Jacobian matrix can be written as a composition of four components:

\[
W = \underbrace{\zeta}_{\text{[volume]}} \underbrace{R}_{\text{[rotation]}} \underbrace{Q}_{\text{[skewness]}} \underbrace{D}_{\text{[aspect-ratio]}}.
\]

A detailed discussion on how TMOP’s target construction methods encode geometric information into the target matrix \( W \) is given by Knupp in [30], and various examples of target construction for different mesh adaptivity goals are given in [28, 31, 32].
Using (1), the Jacobian of the mapping $\Phi_E$ at any reference point $\bar{x} \in \bar{E}$ from the reference-space coordinates to the current physical-space coordinates is defined as

$$A(\bar{x}) = \frac{\partial \Phi_E}{\partial \bar{x}} = \sum_{i=1}^{N_w} x_{E,i} [\nabla \bar{w}_i(\bar{x})]^T. \quad (3)$$

Combining (3) and (2), the transformation from the target coordinates to the current physical coordinates (see Fig. 1) is defined as

$$T = AW^{-1}. \quad (4)$$

With the target transformation $W$ defined in the domain, we next specify a mesh quality metric $\mu(T)$ that compares the transformations $A$ and $W$ in terms of the geometric parameters of interest specified in (2). For example, $\mu_2 = \frac{|T|^2}{2\tau} - 1$ is a shape metric\(^3\) that depends on the skew and aspect-ratio components, but is invariant to orientation/rotation and volume. Similarly, $\mu_{77} = \frac{1}{2}(\tau - \frac{1}{\tau})^2$ is a size metric that depends only on the volume of the element. We also have shape+size metrics such as $\mu_{80} = \gamma \mu_2 + (1 - \gamma)\mu_{77}$ that depend on volume, skew and aspect-ratio, but are invariant to orientation/rotation. Note that the mesh quality metrics are defined such that they evaluate to 0 for an identity transformation, i.e. $\mu(T) = 0$ when $T = I$ $(A = W)$. This allows us to pose the mesh optimization problem as minimization of $\mu(T)$, amongst other advantages [33].

\(^3\)We follow the metric numbering in [33, 34].
The quality metric $\mu(T)$ is used to define the TMOP objective function for $r-$adaptivity

$$F(x) = \sum_{E \in M} F_E(x_E) = \sum_{E \in M} \int_{E_t} \mu(T(x))dx_t,$$

(5)

where $F$ is a sum of the TMOP objective function for each element in the mesh ($F_E$), and $E_t$ is the target element corresponding to the element $E$. In (5), the integral is computed as

$$\sum_{E \in M} \int_{E_t} \mu(T(x_t))dx_t = \sum_{E \in M \times E_t} w_q \det(W(\bar{x}_q)) \mu(T(x_q)),$$

(6)

where $M$ is the current mesh with $N_E$ elements, $w_q$ are the quadrature weights, and the position $x_q$ is the image of the reference quadrature point location $\bar{x}_q$ in the target element $E_t$.

Optimal node locations are determined by minimizing the TMOP objective function (5). This is performed by solving $\partial F(x)/\partial x = 0$ using the Newton’s method where we improve the mesh degrees-of-freedom (nodal positions) as

$$x_{k+1} = x_k - \alpha H^{-1}(x_k)J(x_k).$$

(7)

Here, $x_k$ refers to the nodal positions at the $k$-th Newton iteration during $r-$adaptivity, $\alpha$ is a scalar determined by a line-search procedure, and $H(x_k)$ and $J(x_k)$ are the Hessian ($\partial^2 F(x_k)/\partial x$) and the gradient ($\partial F(x_k)/\partial x$), respectively, associated with the TMOP objective function. The line-search procedure requires that $\alpha$ is chosen such that $F(x_{k+1}) < 1.2F(x_k)$, $|J(x_{k+1})| < 1.2|J(x_k)|$, and that the determinant of the Jacobian of the transformation at each quadrature point in the mesh is positive, $\det(A(x_{k+1})) > 0$. These line-search constraints have been tuned using many numerical experiments and have demonstrated to be effective in improving mesh quality. For Newton’s method, we solve the problem $H^{-1}(x_k)J(x_k)$ using a Krylov subspace method such as the Minimum Residual (MINRES) method with Jacobi preconditioning; more sophisticated preconditioning techniques can be found in [35]. Additionally, the optimization solver iterations (7) are done until the relative $l_2$ norm of the gradient of the objective function with respect to the current and original mesh nodal positions is below a certain tolerance $\varepsilon$, i.e., $|J(x)|/|J(x_0)| \leq \varepsilon$. We set $\varepsilon = 10^{-10}$ for the results presented in the current work.
Figure 2: (a) Original and (b) optimized 4th order meshes for a turbine blade.

Using the approach described in this section, we have demonstrated $r$–adaptivity with TMOP for geometry and simulation-driven optimization; see Fig. 2 for example of high-order mesh optimization for a turbine blade using $W = I$ with the shape metric $\mu_2(T)$.

3. Boundary & Interface Fitting

Our goal for boundary and interface fitting is to enable alignment of a selected set of mesh nodes to some target surface of interest prescribed as the zero isocontour of a smooth discrete level set function, $\sigma(x)$. Figure 3(a) and (b) show a simple example of a circular interface represented using a level set function and a triangular mesh with multi-material interface that is to be aligned to the circular interface. To effect alignment with the zero isocontour of $\sigma(x)$, we modify the TMOP objective function (5) in the following way:

$$F(x) = \sum_{E \in \mathcal{M}} \int_{E_t} \mu(T(x)) dx_t + w_\sigma \sum_{s \in S} \sigma^2(x_s).$$

(8)

Here, $F_\sigma$ is a penalty-type term that depends on the penalization weight $w_\sigma$, the set of nodes $S$ marked for fitting (e.g., the mesh nodes lying on the material interface in Fig. 3(b)), and the level set function $\sigma(x)$, evaluated at the positions $x_s$ of all nodes $s \in S$. The $F_\sigma$ term penalizes the nonzero values of $\sigma(x_s)$ for all $s \in S$. Minimizing this term represents weak enforcement of $\sigma(x_s) = 0$, only for the nodes in $S$, while ignoring the values of $\sigma$ for the nodes outside $S$. Minimizing the full nonlinear objective function, $F = F_\mu + F_\sigma$, would produce a balance between mesh quality and surface fitting. Note that all nodes are treated together,
Figure 3: (a) Level set function $\sigma(x)$, (b) a Cartesian mesh with material interface nodes marked for fitting, and (c) mesh optimized to fit the zero level set of $\sigma(x)$.

i.e., the nonlinear solver would make no explicit separation between volume nodes of $\mathcal{M}$ and the nodes $s \in \mathcal{S} \subseteq \mathcal{M}$ marked for fitting. Furthermore, as there is no pre-determined unique target position for each node of $\mathcal{S}$, the method naturally allows tangential relaxation along the interface of interest, so that mesh quality can be improved while maintaining a good fit to the surface. Figure 3(c) shows an example of a triangular mesh fit to a circular interface using (8). In this example, we use the shape metric $\mu_2$ with equilateral targets and a constant penalization weight, $w_\sigma = 10^3$.

The first step in our method is to use a suitable strategy for representing the level set function with sufficient accuracy (Section 3.1). Next, we choose the set of nodes $\mathcal{S}$ that will be aligned to the zero level set of $\sigma$ such that we can get the best possible fit for the given mesh (Section 3.2). Finally, we set the penalization weight $w_\sigma$ such that an adequate fit is achieved to the target surface while optimizing the mesh with respect to the quality metric $\mu$ (Section 3.3). The adaptive strategy for setting $w_\sigma$ requires us to modify the line-search and convergence criterion of the Newton’s method (Section 3.4). For completeness, the derivatives of $F_\sigma$ are discussed in Section 3.5. Using various examples, we demonstrate in Section 4 that our method extends to both simplices and hexahedrals/quadrilaterals of any order, and is robust in adapting a mesh interface and/or boundary to nontrivial curvilinear shapes.
3.1. Level Set Representation

The following discussion is related to the case when \( \sigma(x) \) is a discrete function so that its values and derivatives can’t be computed analytically. Then the first step in our implicit meshing framework is to ensure that the level set function \( \sigma(x) \) is defined with sufficient accuracy. A drawback of discretizing \( \sigma(x) \) on the mesh being optimized, \( \mathcal{M} \), is that the resulting mesh will be sub-optimal in terms of mesh quality and interface/boundary fit if (a) the mesh does not have enough resolution to represent \( \sigma(x) \) and its gradient with sufficient accuracy, especially near the zero level set, or (b) the target level set is outside the initial domain of the mesh. For the latter, it’s impossible to compute the necessary values and derivatives of the level set function accurately at the boundary nodes that we wish to fit, once the mesh moves outside the initial domain; see Figure 4(a) for an example.

To address these issues, we introduce the notion of a background/source mesh \( (\mathcal{M}_B) \) for discretizing \( \sigma(x_B) \), where \( x_B \) represents the positions of the source mesh nodes, as in Figure 4. Since \( \mathcal{M}_B \) is independent of \( \mathcal{M} \), we can choose \( \mathcal{M}_B \) based on the desired accuracy. We achieve this by using adaptive nonconforming mesh refinements \[36\] around the zero level set of \( \sigma(x_B) \); see Figure 4(b) for an example. Using a source mesh for \( \sigma(x_B) \), however, requires transfer of the level set function and its derivatives from \( \mathcal{M}_B \) to the nodes \( S \in \mathcal{M} \) prior to each Newton iteration. This transfer between the source mesh and the current mesh is done using \textit{gslib}, a high-order interpolation library \[37\]:

\[
\sigma(x) = I(x, x_B, \sigma(x_B)), \quad \partial \sigma(x) = I(x, x_B, \partial \sigma(x_B)), \quad \partial^2 \sigma(x) = I(x, x_B, \partial^2 \sigma(x_B)),
\]  \tag{9}

Figure 4: Using a background mesh to fit discretely prescribed domain boundaries. (a) Initial unfitted mesh and target boundary curve (orange). (b) The boundary curve is prescribed implicitly as the zero level set of a discrete function on an adaptively refined nonconforming background mesh.
where $\mathcal{I}$ represents the interpolation operator that depends on the current mesh nodes ($x$), the source mesh nodes ($x_B$), and the source function $\sigma(x_B)$ or its gradients. A detailed description of how high-order functions can be transferred from a mesh to an arbitrary set of points in physical space using gslib is described in Section 2.3 of [38].

While surfaces such as the circular interface in Fig. 3 and sinusoidal boundary in Fig. 4 are straightforward to define using smooth level-set functions, more intricate domains are typically defined as a combination of nonsmooth functions for different geometric primitives, which are not well suited for our penalization-based formulation (8). Consider for example a two-material application problem in Fig. 5(a) where the domain is prescribed as a combination of geometric primitives for a circle, rectangle, parabola, and a trapezium. The resulting step function $G(x_B)$ is 1 inside one material and -1 inside the other material. For such cases, we start with a coarse background mesh and use adaptive mesh refinement around the zero level set of $G(x_B)$, see Fig 5(b). Then we compute a discrete distance function [39] from the zero level set of $G(x_B)$, see Fig 5(c). The advantage of using this distance function in (8) is that it is (i) generally smoother and (ii) maintains the location of the zero level set.

3.2. Marking for Interface Fitting

For a given level set function, the choice of $S$ impacts the ability of the penalization-based approach to adapt the mesh nodes to the target level set function. A naive approach for
marking is to partition the mesh into two fictitious materials as:

\[
\eta_E = \begin{cases} 
0, & \text{if } \int_E \sigma(x) \geq 0 \\
1, & \text{otherwise}
\end{cases}
\]

where \(\eta_E\) is the material indicator for element \(E\), and then mark the nodes at the material interface for fitting. To demonstrate the limitations of this approach, consider the triangular mesh shown in Figure 6(a) that is marked for fitting to the circular level set shown in Fig. 3. Marking using (10) leads to elements that have adjacent faces marked for fitting (highlighted in red). Such a configuration can lead to vanishing Jacobians at the vertex shared by adjacent faces, as the mesh deforms to align to the level set of interest. Additionally, the quality of the elements surrounding the mesh is sub-optimal as evident in Figure 6(b).

The fundamental issue here is that whenever we have multiple adjacent faces of an element marked for fitting, the fit could be sub-optimal depending on the complexity of the target shape/geometry. To address this challenge we update the material indicator for each element in the mesh as

\[
\tilde{\eta}_E = \begin{cases} 
\eta_E, & \text{if } N_{E,M} \leq 1 \\
1 - \eta_E, & \text{else if } N_{E,M} = N_{E,F} - 1 \\
\eta_E, & \text{(optionally) mark } E \text{ for splitting} \end{cases}
\]

where \(N_{E,M}\) is the number of marked faces of element \(E\), and \(N_{E,F}\) is the total number of faces of element \(E\). Note that (11) is formulated as a two pass approach where we first loop
over all elements with \( \eta_E = 0 \) and then with \( \eta_E = 1 \). This ensures that conflicting decisions are not made for adjacent elements surrounding the interface. For triangular meshes, the approach (11) usually guarantees that there is at most one face marked per element, without the need to split/refine any element in the mesh. Figures 6(c, d) show an example of marking using (11), which avoids the issue of vanishing Jacobian at the vertex shared by adjacent faces.

In quadrilateral elements, when \( 1 < N_{E,M} < N_{E,F} - 1 \), we can optionally do conforming splits on each element to bisect the vertex connecting adjacent faces that have been marked for fitting. The conforming split increases the computational cost associated with the mesh due to increased number of degrees-of-freedom, but leads to better mesh quality. Figure 7 shows an example of a comparison of a quadrilateral mesh fit to the circular level set using the marking strategy (10) and (11).

For tetrahedrons and hexahedrons however, conforming splits independent of adjacent elements are not possible, and we are currently exploring nonconforming refinement strategies.

3.3. Adaptive Penalization Weight

Recall that the balance between mesh quality and node fitting is controlled by the penalization weight \( w_\sigma \) in (8). Numerical experiments show that use of a constant \( w_\sigma \) requires tweaking on a case-by-case basis, and can result in a sub-optimal fit if \( w_\sigma \) is too small, in which case the objective function is dominated by the mesh quality metric term, or if \( w_\sigma \) is
Figure 8: Impact of $w_\sigma$ on the surface fitting error.

too large, in which case the conditioning of the Hessian matrix is poor. Let the maximum surface fitting error be defined as

$$|\sigma|_{S,\infty} := \max_{s \in S} |\sigma(x_s)|.$$

Figure 8 demonstrates how $|\sigma|_{S,\infty}$ varies for the uniform quad mesh fit to a circular interface (recall Figure 7(a, b)) for different fixed values of $w_\sigma$. As we increase $w_\sigma$ from 1 to $10^4$, the fitting error decreases. However, the error worsens if we increase $w_\sigma$ further.

To address this issue we use an adaptive approach for setting $w_\sigma$ where we monitor $|\sigma|_{S,\infty}$, and scale $w_\sigma$ by a user-defined constant ($\alpha_\sigma = 10$ by default) if the relative decrease in the maximum nodal fitting error between subsequent Newton iterations is below a prescribed threshold ($\epsilon_{\Delta \sigma} = 0.001$ by default). That is,

$$w_{k+1,\sigma} = \begin{cases} 
\alpha_\sigma \cdot w_{k,\sigma} & \text{if } |\sigma|_{k,S,\infty} - |\sigma|_{k+1,S,\infty} < \epsilon_{\Delta \sigma}, \\
w_{k,\sigma} & \text{otherwise}
\end{cases}, \quad (12)$$

where we use the subscript $k$ to denote a quantile at the $k$th Newton iteration. Figure 8 shows that this adaptive approach for setting $w_\sigma$ significantly improves the quality of the mesh fit to the desired level set. Updating the value of $w_\sigma$ changes the definition of the
objective function (8), which requires some modifications of the line search and convergence
criterion of the Newton iterations to achieve overall convergence, compared to the constant
\( w_\sigma \) case; details will follow in Section 3.4. Nevertheless, our numerical tests suggest that this
impact is negligible in comparison to the improvement of the fitting error.

3.4. Convergence & Line-Search Criterion

Recall that in the general TMOP approach, the line-search and convergence criteria
for the Newton’s method are based on the magnitude and the derivatives of the objective
function, see Section 2.2. In the penalization-based formulation (8), the current criteria do
not suffice because the magnitude and derivatives of the objective function depend on the
penalization weight \( w_{k,\sigma} \), which can change between subsequent Newton iterations due to
(12).

We modify our line-search criteria by adding two additional inequalities, namely, \( \alpha \) in (7)
is chosen to ensure:

\[
|\sigma|_{k+1,S,\infty} < 1.2 |\sigma|_{k,S,\infty},
\]

\[
\text{det}(A(x_{k+1})) > 0.001 \text{det}(A(x_0)).
\]

The inequality (13) prevents sudden jumps of the fitting error. The constraint (14) is mostly
applicable in regimes when \( w_\sigma \) is big enough to make the quality term \( F_\mu \) effectively inactive.
Such regimes represent a situation when one is willing to sacrifice mesh quality for more
accurate fitting. When the quality term \( F_\mu \) is relatively small, it may be unable to prevent
the appearance of infinitesimally small positive Jacobians; the constraint (14) is used to
alleviate this situation.

The convergence criterion is also modified to utilize the fitting error, i.e., the Newton’s
method is used until \( |\sigma|_{S,\infty} \) is below a certain user-specified threshold \( (\epsilon_\sigma = 10^{-5} \) by default).
We also use an optional convergence criterion based on a limit on the maximum number of
consecutive Newton iterations through which the penalization weight \( w_\sigma \) is adapted using
(12); this limit is \( N_\sigma = 10 \) by default. This latter criterion avoids excessive computations in
cases where the mesh topology does not allow the fitting error to reduce beyond a certain
limit.
3.5. Derivatives

As our default choice for nonlinear optimization is the Newton’s method, we must compute first and second order derivatives of $F_\sigma$ with respect to the mesh nodes. Let the FE position function be $x = (x_1 \ldots x_d)^T$ where $d$ is the space dimension; each component can be written as $x_a(\bar{x}) = \sum_i x_{a,i} \bar{w}_i(\bar{x})$, where $x = x(\bar{x})$, see Section 2.1. Then the Newton’s method solves for the full vector

$$x = (x_{1,1} \ldots x_{1,N_x}, x_{2,1} \ldots x_{2,N_x}, \ldots x_{d,1} \ldots x_{d,N_x})^T$$

that contains the positions of all mesh nodes. The formulas for the first and second derivatives of $F_\sigma$ are the following:

$$\frac{\partial F_\sigma(x)}{\partial x_{a,i}} = 2\omega_\sigma \sum_{s \in S} \sigma(x_s) \frac{\partial \sigma(x_a)}{\partial x_a} \frac{\partial x_a(\bar{x}_s)}{\partial x_{a,i}} = 2\omega_\sigma \sum_{s \in S} \sigma(x_s) \frac{\partial \sigma(x_a)}{\partial x_a} \bar{w}_i(\bar{x}_s),$$

$$\frac{\partial^2 F_\sigma(x)}{\partial x_{b,j} \partial x_{a,i}} = 2\omega_\sigma \sum_{s \in S} \left( \frac{\partial \sigma(x_s)}{\partial x_b} \frac{\partial \sigma(x_s)}{\partial x_a} + 2\omega_\sigma \frac{\partial^2 \sigma(x_s)}{\partial x_b \partial x_a} \right) \bar{w}_i(\bar{x}_s) \bar{w}_j(\bar{x}_s),$$

$$a, b = 1 \ldots d, \quad i, j = 1 \ldots N_x.$$  \hspace{1cm} (15)

The above formulas require the spatial gradients of $\sigma$ at the current positions $\{x_s\}_{s \in S}$ of the marked nodes. These gradients can be closed-form expressions, when $\sigma$ is prescribed analytically, or the gradients are obtained from the background mesh $M_B$ (see Section 3.1), when $\sigma$ is a discrete function.

3.6. Algorithm/Summary

In this section, we summarize our penalization-based method for boundary and interface fitting with TMOP. The inputs to our method are the active/current mesh $M$ that is defined through the global vector $x$ of nodal positions, a user-selected target construction option to form $W$ as in (2), a mesh quality metric $\mu$, a source/background mesh with nodal coordinates $x_B$ along with the level set function $\sigma(x_B)$ as explained in Section 3.1, initial penalization weight $w_\sigma$, parameters for adaptive penalization-weight ($\alpha_\sigma$ and $\epsilon_\Delta_\sigma$ as in Section 3.3), and parameters for convergence criterion $\epsilon_\sigma$ and $N_\sigma$ as in Section 3.4. Algorithm 1 summarizes our penalization-based method where we use subscript $k = 0 \ldots N_{opt}$ to denote different quantiles at the $k$th Newton iteration.
Algorithm 1: Implicit Meshing

**Input:** $x$, $\mu$, $x_B$, $\sigma(x_B)$, $\alpha$, $\epsilon_\sigma$, $w_\sigma$, $\epsilon_\sigma$, $N_{opt}$

**Output:** $x$

1. $n_\sigma := 0$, $k = 0$, $x_0 = x$

2. Determine $S$, the set of nodes for fitting (Section 3.2)

3. $\sigma(x_0) = \mathcal{I}(x_0, x_B, \sigma(x_B))$

4. while $|\sigma|_{k,S,\infty} > \epsilon_\sigma$ and $n_\sigma < N_\sigma$ and $k < N_{opt}$ do

5. $W_i = I$ for each quadrature point $i$

6. $\Delta x = \mathcal{H}^{-1}(x_k)\mathcal{J}(x_k) \rightarrow$ solve using MINRES (Section 2.2).

7. $x_{k+1} = x_k - \alpha \Delta x$, with $\alpha$ determined using line-search (Section 2.2)

8. if $|\sigma|_{k,S,\infty} - |\sigma|_{k+1,S,\infty} < \epsilon_\sigma$ then

9. $w_{k+1,\sigma} \rightarrow \alpha \cdot w_{k,\sigma}$

10. $n_\sigma = 0$

else

12. $n_\sigma = n_\sigma + 1$

end

14. $\sigma(x_{k+1}) = \mathcal{I}(x_{k+1}, x_B, \sigma(x_B))$

15. $k = k + 1$

end

17. $x = x_k$

4. Numerical Results

In this section, we demonstrate the main properties of the method using several examples. The presented tests use $W = I$ as the target matrix, and the following shape metrics:

$$
\mu_2 = \frac{|T|^2}{2\tau} - 1,
$$

$$
\mu_{303} = \frac{|T|^2}{3\tau^{2/3}} - 1,
$$

where $|T|$ and $\tau$ are the Frobenius norm and determinant of $T$, respectively. Both metrics are polyconvex in the sense of [40, 41], i.e., the metric integral $F_\mu$ in (5) theoretically has a
minimizer. Exploring the effect of the smoothness of sigma on the convexity of the objective function (8) will be the subject of future studies.

Our implementation utilizes MFEM, a finite element library that supports arbitrarily high-order spaces and meshes [3]. This implementation is freely available at https://mfem.org.

4.1. Fitting to a Spherical Interface

As a proof of concept, we adapt 3rd order hexahedral (hex) and tetrahedral (tet) meshes to align to a spherical surface; see Figure 9(a) and (b). The domain is a unit-sized cube, $\Omega \in [0,1]^3$, and the level set function $\sigma$ is defined such that its zero isosurface is located at a distance of 0.3 from the center of the domain ($x_c = (0.5, 0.5, 0.5)$). Although this level set is simple enough to be defined analytically, the presented computations represent and use $\sigma$ as a discrete finite element function. Additionally, no background mesh is used in this example, i.e., $x_B = x$.

The initial hex mesh is an $8 \times 8 \times 8$ Cartesian-aligned mesh. The material indicators are setup such that there are a total of 32 elements that have more than one face marked for fitting. The tet mesh is obtained by taking a $4 \times 4$ Cartesian-aligned hex mesh and splitting each hex into 24 tetrahedrons (4 tets-per-hex face). The material indicators are setup such that all faces marked for fitting belong to different elements. The optimized meshes, shown in Fig. 9(c) and (d), have a maximum error of $O(10^{-10})$ at the interface with respect to the zero level set, which is achieved in 76 and 44 Newton iterations for the hexahedral and tetrahedral mesh, respectively. The hexahedral mesh requires more iterations as compared

Figure 9: Initial (a) hexahedral and (b) tetrahedral meshes showing the interfaces for the 3D surface fitting tests. Optimized (c) hexahedral and (d) tetrahedral mesh obtained using [8].
to tetrahedral mesh partly because there are multiple elements with adjacent faces marked for fitting, which requires more work from the adaptive weight mechanism to force those mesh faces to align with the spherical interface while ensuring positive Jacobians everywhere in the mesh.

4.2. Boundary Fitting for Geometric Primitive-Based Domain in 3D

This next example demonstrates implicit high-order meshing using boundary fitting for a nontrivial curvilinear geometry specified as a combination of simple geometric primitives. Here, the target domain is represented as an intersection of a cube (side=0.5, centered around $x_c = (0.5, 0.5, 0.5)$) with a sphere (radius=0.3, centered around $x_c$) that has three Cartesian-aligned cylinders (radius=0.15, length=0.5) removed from it. Figure 10 shows the CSG tree with geometric primitives and the boolean operations that are used to construct the target geometry.

Using the approach outlined in Section 3.1, we combine the geometric primitives on a third-order source mesh that is adaptively refined five times around the zero level set of the target domain. We then compute the distance function on this background mesh using the $p$-Laplacian solver [39]. This distance function is used as the level set function in (8). Figure 11(a) shows a slice-view of the background mesh along with the zero iso-surface of the level set function, and Fig. 11(b) shows a slice-view of the level set function computed as the distance from the zero level set of the geometric primitive-based description of the domain. Our numerical experiments showed that using a third order source mesh with five

![Figure 10: CSG tree with geometric primitives used to define the target geometry.](image-url)
adaptive refinements was computationally cheaper than using a lower order mesh with more adaptive refinements to obtain the same level of accuracy for capturing the curvilinear boundary using the distance function.

The input mesh to be fit for this problem is a uniform second-order $128 \times 128 \times 128$ Cartesian-aligned mesh for $\Omega \in [0, 1]^3$. This mesh is trimmed based on the location of the zero level set and is then optimized using (8). Figure 12 shows the input and the optimized mesh, where the achieved fitting error is $|\sigma|_{S,\infty} = \mathcal{O}(10^{-6})$. 
4.3. Interface Fitting for Shape Optimization Application

This example serves to demonstrate the applicability of the proposed approach to setup the initial multimaterial domain to be used in a shape optimization problem. Figure 13 shows the cross section of a tubular reactor that consists of a highly conductive metal (red) and a low conductivity heat generating region (blue). The design optimization problem is formulated such that the energy production in the system is maximized while keeping the overall volume of the red region fixed. To achieve this, the shape of the red subdomain is morphed using a gradient-based approach in our in-house design optimization framework that requires an interface fitted mesh as an input. This initial fitted mesh is generated using the method described in this manuscript.

Exploiting the cyclic symmetry, we discretize a portion of the domain via a triangular mesh. Since this initial mesh does not need to align with the material interface, it can be conveniently generated using an automatic mesh generator [42]. We then use an approach similar to the previous section for interface fitting where the multimaterial domain is realized as a combination of geometric primitives (annulus, parabola, and trapezium, as highlighted in Figure 13). The finite element distance function from the interface is computed, and used as the level-set function $\sigma(x)$ in (8).

Figure 14(a) shows the input non-fitted mesh, colored by the adaptively assigned a material indicator, such that there is at most one face per triangle marked for fitting. Figure
Figure 14(b) shows the adapted mesh which aligns with the material interface of the domain. The maximum interface fitting error of the optimized mesh is $O(10^{-7})$. Finally, Figures 15(a)-(d) show the initial interface-fitted domain and the shape optimized domain along with the corresponding temperature fields.

Figure 14: Original mesh (a) and interface fitted mesh (b) for the reactor design problem.

Figure 15: Interface fitted domain and shape optimized domain along with corresponding temperature fields from our in-house design optimization framework.
5. Conclusion & Future Work

We have presented a novel method for implicit high-order meshing which can align easy-to-generate meshes with the interfaces and boundaries of the domain of interest. We formulate the mesh optimization problem as a variational minimization of the sum of a chosen mesh-quality metric and a penalty term that weakly forces the selected faces of the mesh to align with the target surface. The key technical components of the proposed method are the use of a source mesh to represent the level set function with sufficient accuracy, adaptively setting the penalization weight, and careful marking of the faces of the mesh to be fit to the target isocontour of the level set field. We have demonstrated that the method is effective for generating boundary- and interface-fitted meshes for nontrivial curvilinear geometries.

In future work, we will improve the method by developing mesh refinement strategies for hexahedral and tetrahedral meshes, which are required when the mesh topology limits how well the mesh can fit to the target surface (Section 3.2). We will also explore ways for aligning meshes to domains with sharp features [26, 43], as we currently assume that the level set function $\sigma$ used in (8) is sufficiently smooth around its zero level set. Finally, we will also look to optimize our method and leverage accelerator-based architectures by utilizing partial assembly and matrix-free finite element calculations [44].

References

[1] M. Deville, P. Fischer, E. Mund, High-order methods for incompressible fluid flow, Cambridge University Press, 2002.

[2] P. Fischer, M. Min, T. Rathnayake, S. Dutta, T. Kolev, V. Dobrev, J.-S. Camier, M. Kronbichler, T. Warburton, K. Świrydowicz, et al., Scalability of high-performance pde solvers, The International Journal of High Performance Computing Applications 34 (5) (2020) 562–586.

[3] R. Anderson, J. Andrej, A. Barker, J. Bramwell, J.-S. Camier, J. Cerveny, V. A. Dobrev, Y. Dudouit, A. Fisher, T. V. Kolev, W. Pazner, M. Stowell, V. Z. Tomov, I. Akkerman, J. Dahm, D. Medina, S. Zampini, MFEM: a modular finite elements methods library, Comput. Math. Appl. 81 (2020) 42–74. doi:10.1016/j.camwa.2020.06.009.
[4] T. Kolev, P. Fischer, M. Min, J. Dongarra, J. Brown, V. Dobrev, T. Warburton, S. Tomov, M. S. Shephard, A. Abdelfattah, V. Barra, N. Beams, J.-S. Camier, N. Chalmers, Y. Dudouit, A. Karakus, I. Karlin, S. Kerkemeier, Y.-H. Lan, D. Medina, E. Merzari, A. Obabko, W. Pazner, T. Rathnayake, C. W. Smith, L. Spies, K. Swirydowicz, J. Thompson, A. Tomboulides, V. Tomov, Efficient exascale discretizations: High-order finite element methods, International Journal of High Performance Computing Applications To appear (2021). 1

[5] X.-J. Luo, M. Shephard, L.-Q. Lee, L. Ge, C. Ng, Moving curved mesh adaptation for higher-order finite element simulations, Engineering with Computers 27 (1) (2011) 41–50. 2

[6] V. Dobrev, T. Kolev, R. Rieben, High-order curvilinear finite element methods for Lagrangian hydrodynamics, SIAM Journal on Scientific Computing 34 (5) (2012) 606–641. 2

[7] W. Boscheri, M. Dumbser, High order accurate direct arbitrary-Lagrangian-Eulerian ADER-WENO finite volume schemes on moving curvilinear unstructured meshes, Computers & Fluids 136 (2016) 48–66. 2

[8] A. A. Requicha, H. B. Voelcker, Constructive solid geometry (1977). 2

[9] M. Sussman, P. Smereka, S. Osher, A level set approach for computing solutions to incompressible two-phase flow, J. Comput. Phys. 114 (1) (1994) 146–159. doi:10.1006/jcph.1994.1155. 2

[10] J. Sokolowski, J.-P. Zolésio, Introduction to shape optimization, in: Introduction to shape optimization, Springer, 1992, pp. 5–12. 2

[11] G. Allaire, C. Dapogny, P. Frey, Shape optimization with a level set based mesh evolution method, Computer Methods in Applied Mechanics and Engineering 282 (2014) 22–53. 2
[12] M. Hojjat, E. Stavropoulou, K.-U. Bletzinger, The vertex morphing method for node-based shape optimization, Computer Methods in Applied Mechanics and Engineering 268 (2014) 494–513.

[13] Z. Q. Xie, R. Sevilla, O. Hassan, K. Morgan, The generation of arbitrary order curved meshes for 3d finite element analysis, Computational Mechanics 51 (3) (2013) 361–374.

[14] M. Fortunato, P.-O. Persson, High-order unstructured curved mesh generation using the winslow equations, Journal of Computational Physics 307 (2016) 1–14.

[15] D. Moxey, M. Green, S. Sherwin, J. Peiró, An isoparametric approach to high-order curvilinear boundary-layer meshing, Computer Methods in Applied Mechanics and Engineering 283 (2015) 636–650.

[16] 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, International Journal for Numerical Methods in Engineering 106 (13) (2016) 1100–1130.

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

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

[19] E. Ruiz-Gironés, X. Roca, Automatic penalty and degree continuation for parallel preconditioned mesh curving on virtual geometry, Computer-Aided Design 146 (2022) 103208.

[20] R. Rangarajan, H. Kabaria, A. Lew, An algorithm for triangulating smooth three-dimensional domains immersed in universal meshes, International Journal for Numerical Methods in Engineering 117 (1) (2019) 84–117.
A. Abdelkader, C. L. Bajaj, M. S. Ebeida, A. H. Mahmoud, S. A. Mitchell, J. D. Owens, A. A. Rushdi, Vorocrust: Voronoi meshing without clipping, ACM Transactions on Graphics (TOG) 39 (3) (2020) 1–16. 2

K. Mittal, P. Fischer, Mesh smoothing for the spectral element method, Journal of Scientific Computing 78 (2) (2019) 1152–1173. 2

P.-O. Persson, G. Strang, A simple mesh generator in MATLAB, SIAM review 46 (2) (2004) 329–345. 2

L. Chen, H. Wei, M. Wen, An interface-fitted mesh generator and virtual element methods for elliptic interface problems, Journal of Computational Physics 334 (2017) 327–348. 2

Isovalue discretization algorithm.
URL https://www.mmgtools.org/mmg-remesher-try-mmg/mmg-remesher-options/isovalue-discretization-algorithm 2

Y. Ohtake, A. G. Belyaev, Dual-primal mesh optimization for polygonized implicit surfaces with sharp features, J. Comput. Inf. Sci. Eng. 2 (4) (2002) 277–284. 2, 22

N. Le Goff, F. Ledoux, S. Owen, Volume preservation improvement for interface reconstruction hexahedral methods, Procedia Engineering 203 (2017) 258–270. 2

V. A. Dobrev, P. Knupp, T. V. Kolev, K. Mittal, V. Z. Tomov, The Target-Matrix Optimization Paradigm for high-order meshes, SIAM J. Sci. Comp. 41 (1) (2019) B50–B68. 3, 4

P. Knupp, Introducing the target-matrix paradigm for mesh optimization by node movement, Engr. with Comptr. 28 (4) (2012) 419–429. 3

P. Knupp, Target formulation and construction in mesh quality improvement, Tech. Rep. LLNL-TR-795097, Lawrence Livermore National Lab.(LLNL), Livermore, CA (2019). 4, 16
[31] V. A. Dobrev, P. Knupp, T. V. Kolev, K. Mittal, R. N. Rieben, V. Z. Tomov, Simulation-driven optimization of high-order meshes in ALE hydrodynamics, Comput. Fluids (2020).

[32] V. A. Dobrev, P. Knupp, T. V. Kolev, K. Mittal, V. Z. Tomov, HR-adaptivity for nonconforming high-order meshes with the Target-Matrix Optimization Paradigm, Eng. Comput. (2021). doi:10.1007/s00366-021-01407-6

[33] P. Knupp, Metric type in the target-matrix mesh optimization paradigm, Tech. Rep. LLNL-TR-817490, Lawrence Livermore National Lab.(LLNL), Livermore, CA (United States) (2020).

[34] P. Knupp, Geometric parameters in the target matrix mesh optimization paradigm, Partial Differential Equations in Applied Mathematics (2022) 100390.

[35] E. Ruiz-Gironés, X. Roca, Automatic penalty and degree continuation for parallel preconditioned mesh curving on virtual geometry, Comput.-Aided Des. 146 (2022) 103208. doi:10.1016/j.cad.2022.103208

[36] J. Cerveny, V. Dobrev, T. Kolev, Nonconforming mesh refinement for high-order finite elements, SIAM Journal on Scientific Computing 41 (4) (2019) C367–C392.

[37] P. Fischer, GSLIB: sparse communication library [Software], https://github.com/gslib/gslib (2017).

[38] K. Mittal, S. Dutta, P. Fischer, Nonconforming Schwarz-spectral element methods for incompressible flow, Computers & Fluids 191 (2019) 104237.

[39] A. G. Belyaev, P.-A. Fayolle, On variational and pde-based distance function approximations, in: Computer Graphics Forum, Vol. 34, Wiley Online Library, 2015, pp. 104–118.

[40] V. A. Garanzha, Polyconvex potentials, invertible deformations, and thermodynamically consistent formulation of the nonlinear elasticity equations, Comp. Math. Math. Phys. 50 (9) (2010) 1561–1587.
[41] V. Garanzha, L. Kudryavtseva, S. Utyuzhnikov, Variational method for untangling and optimization of spatial meshes, J. Comput. Appl. Math. 269 (2014) 24–41. [16]

[42] T. D. Blacker, S. J. Owen, M. L. Staten, W. R. Quadros, B. Hanks, B. W. Clark, R. J. Meyers, C. Ernst, K. Merkley, R. Morris, et al., Cubit geometry and mesh generation toolkit 15.1 user documentation, Tech. rep., Sandia National Lab.(SNL-NM), Albuquerque, NM (United States) (2016). [20]

[43] M. J. Zahr, A. Shi, P.-O. Persson, Implicit shock tracking using an optimization-based high-order discontinuous galerkin method, Journal of Computational Physics 410 (2020) 109385. [22]

[44] J.-S. Camier, V. Dobrev, P. Knupp, T. Kolev, K. Mittal, R. Rieben, V. Tomov, Accelerating high-order mesh optimization using finite element partial assembly on gpus, arXiv preprint arXiv:2205.12721 (2022). [22]