Computer-Aided Design 190 (2026) 103966

Contents lists available at ScienceDirect

Computer-Aided Design

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

Mesh generation of curvilinear polygons for the high-order virtual element method (VEM)☆

Kaloyan S. Kirilov [a], Jingtian Zhou [a], Joaquim Peiró [a,*], Mashy Green [b], David Moxey [b], Lourenço Beirão da Veiga [c], Alessandro Russo [c], Franco Dassi [c]

[a] Department of Aeronautics, Imperial College London, South Kensington Campus, London, SW7 2AZ, UK
[b] Department of Engineering, King's College London, Strand, London, WC2R 2LS, UK
[c] Department of Mathematics and Applications, University of Milano-Bicocca, Piazza dell'Ateneo Nuovo 1, Milano, 20126, Italy

ARTICLE INFO

Keywords:
Virtual element method
High-order approximation
High-order polygonal mesh generation
Polygonal mesh quality optimisation

ABSTRACT

We present a proof-of-concept methodology for generating curvilinear polygonal meshes suitable for high-order discretisations by the Virtual Element Method (VEM). A VEM discretisation requires the definition of a set of boundary and internal points used to define basis functions and compute integrals of polynomials. The procedure to locate these points on the boundary borrows ideas from previous work on *a posteriori* high-order mesh generation in which the geometrical inquiries to a B-rep model of the computational domain are performed via an interface to CAD libraries.

Here we describe the steps of the procedure that transforms a straight-sided polygonal mesh, generated using third-party software, into a curvilinear boundary-conforming mesh. We discuss criteria for ensuring and verifying the validity of the mesh. Using an elliptic partial differential equation with Dirichlet boundary conditions as a model problem, we show that VEM discretisations on such meshes achieve the expected rates of convergence as the mesh resolution is increased. This is followed by an illustrative application of the method to the generation of a curvilinear polygonal mesh for an aerofoil geometry.

We discuss polygonal curvilinear mesh quality and its enhancement, and use the motion of a cell vertex to appraise three elemental quality metrics, namely convexity, regularity and isotropy, and highlight some of the difficulties associated in their use for mesh quality optimisation. A derivative-free optimisation method is utilised to enhance curvilinear polygonal meshes by maximising a suitable measure of mesh quality. We propose such measure as a combination of the three quality metrics and apply it to optimise a distorted initial mesh for a ring geometry. We show that a suitable version of the convexity metric is effective in untangling invalid meshes. The VEM solution of a model elliptic equation is obtained for a ring geometry where a distorted and an optimised mesh show low errors, indicating that the VEM is robust and relatively insensitive to mesh distortion, and a reduction of the error in the optimised mesh.

Finally, we use a more complex geometry, a computational domain for an aerofoil, as a benchmark to further illustrate the ability of the convexity metric to untangle meshes, and also to assess the suitability of two quality measures as optimisation targets to improve the overall quality of curvilinear polygonal meshes.

1. Introduction

Polytopal meshes, with arbitrarily shaped 2D polygonal or 3D polyhedral computational cells, have been routinely used for the discretisation of partial differential equations (PDEs) by finite volume methods [1]. The main advantages of using unstructured polytopal meshes are their ability to discretise complex computational domains and their potential to reduce the computational complexity of the PDE solver. Recently, there has been a growing interest in developing discretisation methods that support meshes of polygonal and polyhedral cell shapes with low and high approximation orders. A literature review of the large variety of polytopal methods is given in Ref. [2].

☆ This article is part of a Special issue entitled: 'SIAM IMR 2023/24' published in Computer-Aided Design.

* Corresponding author.
E-mail addresses: kaloyan.kirilov19@imperial.ac.uk (K.S. Kirilov), jingtian.zhou22@imperial.ac.uk (J. Zhou), j.peiro@imperial.ac.uk (J. Peiró), mashy.green@kcl.ac.uk (M. Green), david.moxey@kcl.ac.uk (D. Moxey), lourenco.beirao@unimib.it (L. Beirão da Veiga), alessandro.russo@unimib.it (A. Russo), franco.dassi@unimib.it (F. Dassi).

https://doi.org/10.1016/j.cad.2025.103966
Received 31 March 2025; Received in revised form 18 August 2025; Accepted 1 September 2025
Available online 15 September 2025

The majority of these methods make use of polytopal meshes with straight edges and faces that, especially for high-order methods, can deteriorate the accuracy of the solution in the presence of curved boundaries or interfaces. As it is well known from the Finite Element Method literature, the representation of the domain geometry with planar facets introduces an error that can stagnate convergence if the order of the approximation is increased. It is therefore important to ensure that the curved interfaces and boundaries are accurately approximated to guarantee the expected order of convergence. To achieve this, one should aim at defining discrete spaces on curved elements in such a way that the domain geometry is accurately represented. Examples of this are the high-order polynomial maps employed in isoparametric finite elements [3], and the use of a CAD representation of the computational domain in isogeometric analysis [4].

There are a few approaches that permit the definition of those discrete spaces in the context of curvilinear polytopal meshes, such as the polytopal discontinuous Galerkin [5], hybrid high-order [2,6], and virtual element methods [7,8]. Although here we focus on 2D curvilinear mesh generation for the VEM, the proposed methodology is equally applicable to other polygonal methods that deal with curved edges.

To the best of our knowledge, very few methods exist for the generation of boundary-conforming curvilinear polygonal meshes. One exception is Ref. [9] that adopts a NURBS-enhanced VEM strategy where the edges on boundaries and interfaces are NURBS curves, each defined to exactly match the CAD description of the boundary. The approach that we propose here differs from that in Ref. [9] in that the geometrical information required by the VEM discretisation is obtained through an application programming interface (API) that performs all the required geometrical enquiries on a standard CAD representation of the boundary which eliminates the need to define an individual NURBS representation of each edge that matches the CAD definition. This is more general, employs similar procedures to those employed by current state-of-the-art curvilinear high-order mesh generators [10,11], and thus facilitates the extension of the methodology to 3D problems. However, we will present the main ideas using a 2D proof-of-concept in the following sections.

The outline of the paper is as follows. A description of the basic ideas underlying the VEM discretisation of elliptic PDEs is presented in Section 2. Section 3 describes the proposed methodology for the *a posteriori* mesh generation of curvilinear polygonal meshes with a particular emphasis on how to achieve boundary conformity to CAD geometries. To assess the validity of the generated curvilinear meshes, Section 4 will show that the expected rate of convergence of the solution of a model elliptic PDE is achieved in these meshes. Section 5 will illustrate the application of the method to the generation of a curvilinear polygonal mesh for an aerofoil geometry. Issues concerning polygonal curvilinear mesh quality and its enhancement will be discussed in Section 6 where we use the motion of a cell vertex to appraise three quality metrics, namely convexity, regularity and isotropy, and highlight some of the difficulties associated in their use for mesh optimisation. It will also show a variant of the convexity metric to be adept at untangling invalid meshes where edges intersect. An example of mesh optimisation of a high-order mesh discretising a more complex geometry, a computational domain for an aerofoil, is presented in Section 6.6.3. Finally, conclusions will be drawn in Section 7.

2. High-order VEM basics

This section describes the basics of the VEM to discretise an elliptic PDE in two-dimensional domains with curved boundaries or interfaces. More specifically, we will focus on the information required to proceed with the workflow of the VEM curvilinear mesh generation pipeline, which will be presented later in Section 3. Since this paper is a proof-of-concept methodology for generating polygonal meshes with curved edges, here we will give a brief description of the VEM and refer the reader to Ref. [7] for more details.

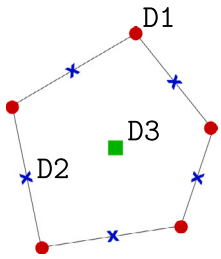

Fig. 1. Degrees of freedom for the straight case with polynomial order $k = 2$.

In the remainder of the paper, we will use a linear elliptic PDE with Dirichlet boundary conditions, i.e. find a function $u(x, y)$ such that

$$-\Delta u = f \ \text{ in } \Omega; \quad u = g \ \text{ in } \partial\Omega, \tag{1}$$

where f and g are functions with appropriate regularity conditions. We consider this standard model problem in order to better illustrate the main features of the VEM and to identify the geometrical operations required to formulate it. We will first recall how to formulate a VEM on a straight-sided polygonal mesh [12] and then extend such approach to deal with curved edges.

Let E be a polygon with *all* straight edges. The local VEM space on E is defined as

$$V_h^k(E) := \left\{ \begin{array}{l} v \in H^1(E) \text{ s.t. } v|_{\partial E} \in C^0(\partial E), \\ \Delta v \in \mathbb{P}_{k-2}(E), \\ v|_e \in \mathbb{P}_k(e) \, \forall e \subset \partial E \end{array} \right\}. \tag{2}$$

where $\mathbb{P}_s(\mathcal{O})$ denotes the set of polynomials of order s on the set $\mathcal{O}$. A function $v \in V_h^k(E)$ is uniquely determined by the following degrees of freedom:

D1: value of v at the vertices;
D2: $k-1$ values of v on the edge nodes;
D3: $k(k-1)/2$ moments $\int_E v\, p_{k-2} \, \mathrm{d}E$,

Fig. 1 shows the degrees of freedom associated with a VEM space of order 2 on a pentagon.

Although such degrees of freedom uniquely determine a function $v \in V_h^k(E)$, such a function is virtual, i.e. its values are not known point-wise. This seems to be a limitation of the space $V_h^k(E)$ but, in the virtual element framework, there is no need to know the explicit expression of the functions in $V_h^k(E)$ to obtain the solution of the PDE (1). Indeed, starting from the degrees of freedom, one can create projection operators, assemble the global matrix and finally compute the solution [13]. To illustrate how this can be achieved, we will show how to compute a specific integral *without* knowing point-wise values of the function $v \in V_h^k(E)$. Suppose that we would like to compute

$$\int_E \nabla v \cdot \nabla p_k \, \mathrm{d}E, \tag{3}$$

where $p_k \in \mathbb{P}_k(E)$ using only the degrees of freedom of the virtual function $v \in V_h^k(E)$. Since we do not know point-wise the virtual function v_h *inside* E, we cannot apply directly a quadrature rule formula. To obtain the value of the integral in Eq. (3), we integrate by parts and get

$$\int_E \nabla v \cdot \nabla p_k \, \mathrm{d}E = -\int_E v\, \Delta p_k \, \mathrm{d}E + \int_{\partial E} v\, (\mathbf{n} \cdot \nabla p_k) \, \mathrm{d}s.$$

Now we claim that the right-hand side of this equation is computable from the degrees of freedom. Indeed, the integral involves the virtual function and a linear combination of polynomials of order $k-2$ which corresponds to the degrees of freedom D3. The integral over ∂E can then be split into integrals on each edge of E where the virtual function v is a polynomial of degree k which can be reconstructed starting from the degrees of freedom D1 and D2.

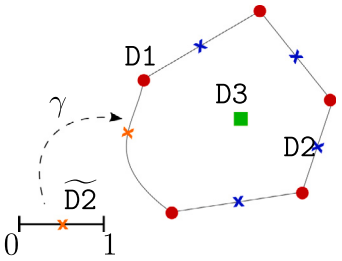

Fig. 2. Degrees of freedom for the curved case with polynomial order $k = 2$. The mapping $\gamma(t)$ $(0 \le t \le 1)$ is used to define the curved edge.

2.1. Extension to curved edges

To deal with polygons with curved edges, the idea is to put geometry information within $V_h^k(E)$. Given a polygon E, we denote by ∂E and $\widetilde{\partial E}$ the set of straight and curved edges, respectively. Then, the VE space defined on this polygon becomes

$$\begin{aligned} \mathcal{V}_h^k(E) := \Big\{ & v \in H^1(E) \text{ s.t. } v|_{\partial E \cup \widetilde{\partial E}} \in C^0(\partial E \cup \widetilde{\partial E}), \\ & \Delta v \in \mathbb{P}_{k-2}(E), \\ & v|_e \in \mathbb{P}_k(e) \, \forall e \subset \partial E, \\ & v|_{\mathfrak{e}} \in \widetilde{\mathbb{P}}_k(\mathfrak{e}) \, \forall \mathfrak{e} \subset \widetilde{\partial E} \Big\}. \end{aligned} \tag{4}$$

In this definition, we have introduced the space

$$\widetilde{\mathbb{P}}_k(\mathfrak{e}) := \mathbb{P}_k([0,1]) \circ \gamma,$$

where γ is a sufficiently regular map that describes the curved edge of the polygon, see Fig. 2. Such a polynomial space of degree k in the one variable parameter space of the curved edge is the key point to obtain a functional space on polygons with curved edges.

In this case a function $v \in \mathcal{V}_h^k(E)$ is also virtual since we do not know its explicit expression, but it can be uniquely determined by the following degrees of freedom:

D1: value at the vertices;
D2: $k-1$ values on straight edges;
$\widetilde{\text{D2}}$: $k-1$ values on the parameter space $[0, 1]$ associated with the curved edge $\mathfrak{e}$;
D3: $k(k-1)/2$ moments $\int_E v \, p_{k-2} \, \mathrm{d}E$.

We can better appreciate that the curved space is an extension of the straight one by comparing the definition of spaces $V_h^k(E)$ and $\mathcal{V}_h^k(E)$. Indeed, if a polygon E does not have any curved edges, Eqs. (2) and (4) yield identical spaces. Further evidence of this are the degrees of freedom: these two spaces share the degrees of freedom D1, D2 and D3 but there is an additional type of degrees of freedom, $\widetilde{\text{D2}}$, when the polygon has a curved edge.

The fact that $\mathcal{V}_h^k(E)$ extends of $V_h^k(E)$ provides practical advantages. Specifically, if we can compute integrals on polygons with curved edges and on the curved edges themselves, the VEM framework remains unchanged. Similarly in this setting, the degrees of freedom are the only essential ingredients needed to construct projection operators, assemble the global matrix, and compute the solution [7]. Moreover, the implementation requires only a modification of the integration procedure on the edges, *without* further changes to the overall structure of the code.

However, we need additional information from the geometrical viewpoint:

- The coordinates of a set of quadrature points and their corresponding weights for evaluating integrals over curved edges, and in polygons characterised by curved edges.
- The mapping $\gamma(t)$ defining the curved edge which is used to compute tangent and normal vectors appearing in some of the integrals.

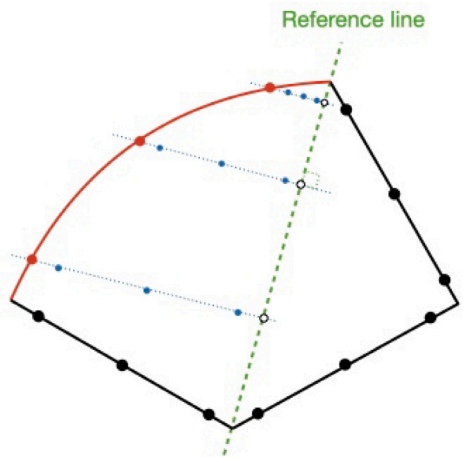

Fig. 3. VEM quadrature points: The edge quadrature points are shown as large dots on the edges of the polygon. The generation of internal quadrature points is illustrated for the curved edge (in red) only. Here we define a reference line and trace perpendicular lines to it passing through the edge quadrature points. On the perpendicular lines, standard quadrature points (small dots) are located on the segment within the edge quadrature point and the intersection point (circle) with the reference line. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

The former point is particularly interesting because the numerical integration over curved edges is standard and uses quadrature rules on the parameter space, but the integration inside polygons characterised by curved edges is still an active research topic [14,15].

In the numerical experiments to be undertaken in Sections 4 and 6, we use the approach proposed in Ref. [15]. In brief, and following the notation of Fig. 3, a reference line is built to trace perpendicular lines through the quadrature points of each edge. Then, on each of these lines, standard quadrature points are located on the segment within the edge quadrature point and the intersection point with the reference line. The choice of the reference line is crucial for this method: it has to be chosen so that no quadrature points may fall outside of the polygon.

3. *A posteriori* high-order polygonal mesh generation

We seek to generate polygonal meshes suitable for high-order VEM discretisations that conform to a computational domain boundary defined in terms of a standard CAD boundary representation (B-rep) [16]. We follow essentially an *a posteriori* high-order mesh generation approach where we modify a straight-sided polygonal mesh and transform it into a curvilinear mesh that conforms to the boundary. This process is illustrated in Fig. 4.

The methodology aims to be completely detached from the VEM solver and to support every user-defined geometrical order. All this is made possible by extending the current capabilities of the open-source high-order mesh generator *NekMesh* [17] and its application programming interface (API) for geometrical inquiries to external CAD libraries such as OpenCascade [18] and CADfix [19]. In the following we will refer to these libraries as the CAD engine or the API.

As in the classical *a posteriori* high-order mesh generation pipelines, it is necessary to generate a valid straight-sided mesh first, and then use high-order tools to curve the boundary and interface mesh edges whilst ensuring the new boundary-conforming elements are valid and of high quality. A graphical illustration of this process is given in Fig. 4. We follow an approach where the straight-sided polygonal mesh is

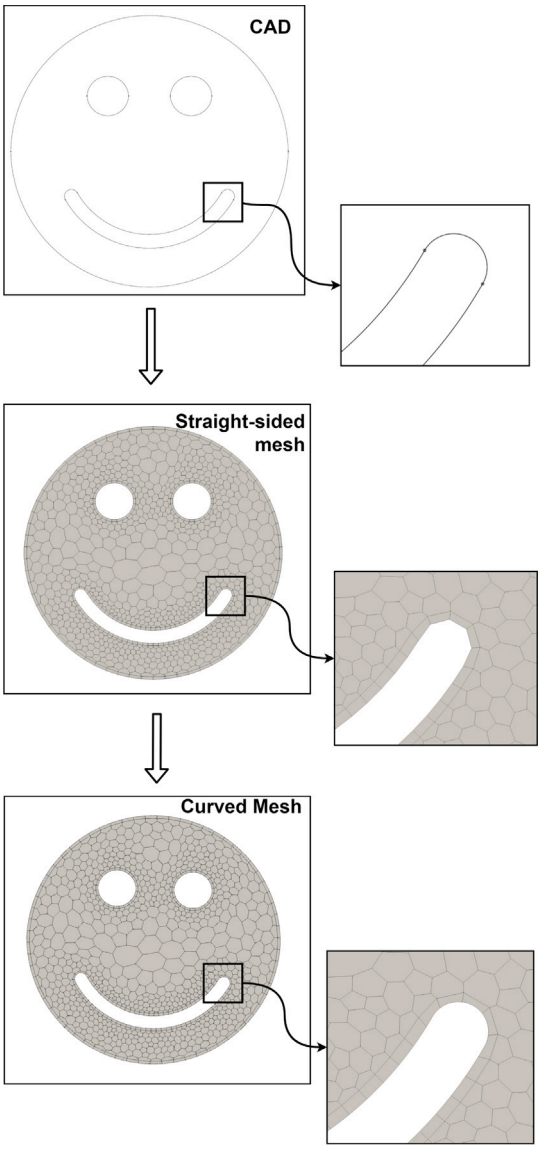

Fig. 4. The *a posteriori* approach to high-order VEM mesh generation. From top to bottom: CAD B-Rep definition of the domain, straight-sided polygonal mesh, and curvilinear high-order mesh.

generated using a third-party software, in our case STAR-CCM+ [20]. Then we perform *a posteriori* connectivity identification between the linear polygonal vertices, edges and the corresponding CAD. objects. We construct and project the high-order nodes on the CAD as specified by a particular combination of quadrature rules. Finally, using these projected nodal points, we use the CAD API to retrieve all the geometrical information relevant to the VEM solver [21].

3.1. Generation of the straight-sided polygonal mesh

Two main strategies could be employed to generate the straight-sided polygonal mesh. First is the classical bottom-up strategy, where the vertices are inserted directly onto CAD objects (B-Splines, NURBS, etc.) inside *NekMesh*. Then one can generate seed points and perform Voronoi tesselations, including refinements, as described in Ref. [9]. This ensures CAD conformity and would allow direct use of the isogeometric VEM without any further mesh manipulations.

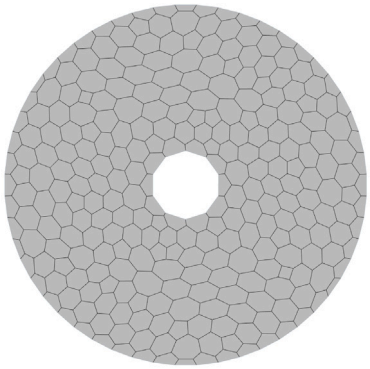

Fig. 5. A coarse linear mesh of a ring with interior radius $R_1 = 0.2$ and exterior radius $R_2 = 1$.

To detach the geometrical information from the numerical discretisation, so that the interaction with the CAD B-rep is not handled by the VEM solver, we start the generation process from a straight-sided polygonal mesh created using a third-party software. Here we employ STAR-CCM+ [20], a robust commercial polyhedral mesh generator that can read CAD information directly. Furthermore it can be combined with multiple fine control features, including anisotropic prism/quad layers, curvature refinement, maximum edge deviation from the CAD, vertex projection on CAD surfaces, multi-surface proximity mesh control and separate patch (curve and surface) control.

The main requirements on the user side for this step are ensuring boundary conformity of the polygonal vertices and a CAD deviation distance within the minimum edge length. Additionally, the user should ensure that the first layer of elements is thick enough to accommodate the edge projection on the CAD. In the following, and for simplicity, we will use a circular ring domain to illustrate some of these features. An example of a straight-sided polygonal mesh generated using STAR-CCM+ is depicted in Fig. 5.

Once created in STAR-CCM+, the straight-sided mesh is exported to a `.ccm` file which is read by *NekMesh* through the CCM OpenFoam importer [22]. More specific details about the implementation are given in Section 3.5. This input process populates the classical *NekMesh* data structure of elements, edges and vertices. These also include topological information, for instance the connectivity between the mesh entities, and potential boundary flags set by the user in STAR-CCM+. However these do not contain the CAD information at this stage.

3.2. API to a CAD engine for geometrical queries

To use any of the high-order tools available, one needs to first obtain the CAD information and then link the boundary mesh entities with the corresponding CAD objects. *NekMesh* achieves that through an API that links to CAD engines. At this moment, *NekMesh* supports OpenCascade [18] and ITI CADFix [19]. This API can read geometrical objects such as points, lines, and topological information from a standard STEP file format [23] which can be created using state-of-the-art CAD software. Moreover, the API is responsible for calculating the necessary geometrical information related to the CAD for the following functions:

- mapping parametric location t on a CAD curve to its Cartesian location $\overline{x}$,
- mapping Cartesian location $\overline{x}$ to parametric coordinate t,
- calculating the closest distance d to a CAD curve given a coordinate $\overline{x}$, and
- evaluating normal $\overline{N}$ and tangent vectors $\overline{T} = \overline{x}'(t)$ to the curve.

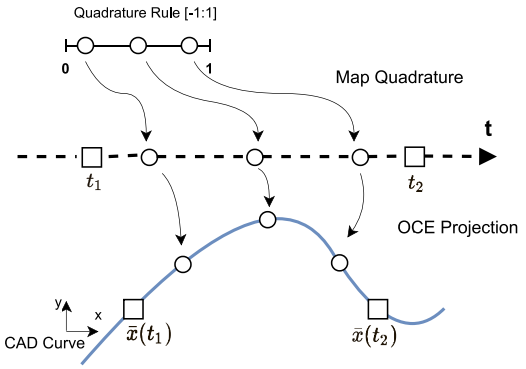

Fig. 6. Parametric projection for an edge with a single CAD curve.

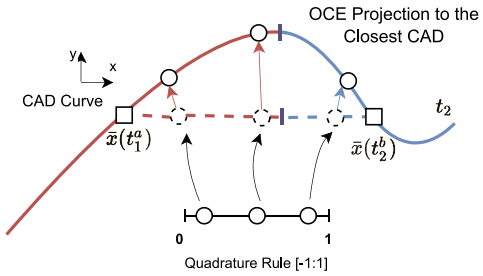

Fig. 7. Projection for a straight edge with 2 CAD curves.

At this point, both the linear mesh and the CAD objects are available and we can connect the entities by shortlisting the edges, boundary vertices and boundary elements from the edge boundary flags.

Due to the tendency of STAR-CCM+ polygonal mesh generator to place vertices further away from the CAD curve, one cannot just use these boundary flags, but needs to identify the closest curve to every vertex. In order to do this, *NekMesh* creates a thin bounding box in Cartesian space around every CAD curve, within a geometrical tolerance in the range of 0.001 to 0.01 times the maximum length of the box, and stores it in a k-D tree data structure [24]. Then exploiting this k-D tree, we shortlist only several potential CAD curve candidates for every vertex. For these, we calculate the distance of the vertex to the parametric CAD curves with the help of the API. If the shortest one is further away than a small distance, this vertex and the corresponding edge are left straight to avoid mesh entanglement. Otherwise, the vertex is projected to the CAD curve with its corresponding parametric location, t.

The extension to edges is straightforward. If the two vertices belong to the same CAD curve, then this edge is clearly part of this CAD object, and it is marked as such. An important exception is when the two edge vertices have no common CAD object. This could happen in the junction between two connected CAD curves, where STAR-CCM+ inserts an edge. As will be discussed later, this rare case requires a special curving technique.

This strategy has also been used extensively for 3D element tetrahedral, hexahedral and prismatic elements. Therefore, the extension to a 3D polyhedral one is relatively straightforward, but with the API now performing geometrical queries on 3D CAD objects: curves and surfaces.

3.3. CAD projection of additional points

The main difficulty in curving the edges occurs when the two vertices of an edge lie on the same CAD object. Here *NekMesh* employs the CAD curve and the API to parametrically create the high-order edge-nodes according to a quadrature rule, defined on the reference segment $[-1, 1]$, which is typically a form of Gaussian quadrature. To evaluate their positions on the curve, we utilise the mapping $\gamma(t)$ which defines the edge in the region $0 \leq t_1 \leq t \leq t_2 \leq 1$. We therefore construct a mapping between the intervals $[-1, 1]$ and $[t_1, t_2]$, then apply $\gamma(t)$ to locate the points in Cartesian space along with their parametric coordinates t_j. Finally, we calculate the distance, d_j, between the quadrature points on the straight edge and the corresponding one on the curve. In an unlikely scenario that the distance d_j is larger than the edge length, a projection error could have occurred in the CAD engine, and therefore, the edge is linearised. This process ensures exact parametric projection to the CAD curves, mesh boundary conformity and easy access with the API to the necessary geometrical information for output (see Fig. 6).

A rare exception to this process happens when STAR-CCM+ generates an element that spans two CAD objects. In this case, the edge is located across two CAD curves, so the previous approach cannot be applied. Therefore, we generate the quadrature points on the straight edge first. Then we project the edge node j to the closest Cartesian location of the two CAD objects. Note that this introduces a small error in the location of the quadrature points but, if the curves are smooth, it has a negligible effect on the solution accuracy. A schematic of the two processes can be seen in Fig. 7.

The projection of points employs the geometrical procedures available in the API. These procedures rely upon third-party implementations such as OpenCascade which are reasonably robust in general. However, their applicability may be limited in some instances such as, for instance, when the curved edges exhibit inflections or very rapid changes in curvature.

Following these steps, *NekMesh* can produce curved meshes with arbitrary user-defined order. Moreover, the mesh generator is completely detached from the VEM solver, it is based on the user's requirements, and supports any combinations of classical 1D quadrature rules such as Gauss, Gauss-Lobbato, Gauss-Radau, etc. However, the construction of the quadrature points within the polygonal element is left to the solver side, due to the variety of techniques adopted by the different VEM solvers.

3.4. Towards ensuring mesh validity

In regions with high curvature, it is possible to generate tangled polygonal elements after the edge projection step, where the addition of curvature causes the element to self-intersect. Therefore, in the legacy *NekMesh* pipeline with *standard* element shapes such as triangles and quadrilaterals, we calculate the distortion of each element using the Jacobian of the mapping from the standard reference space [25]. A negative value indicates an invalid tangled element and hence this element needs to be either linearised or refined. This approach is not easily applicable to arbitrary polygons in the VEM due to the difficulty in defining such mapping. In 2D domains a visual inspection of the mesh often helps identifying invalid elements, however this is not feasible in 3D. One can devise a method to detect the presence of invalid elements by calculating the signed area of a polygon as an integral over its boundary. If the area of the computational domain, its boundary viewed as a polygon, differs from the sum of the areas of the polygonal elements calculated in the same fashion, then there are tangled elements in the mesh. These elements can then be identified (in 2D) by calculating the winding number of the polygon, which will be different from zero if self-intersection occurs.

An alternative method for imposing mesh validity is to construct a single sufficiently thick boundary layer in the regions of concern which accommodates the curving of the mesh effected by the CAD projection. It has been shown in the literature [3] that this minimum thickness $\delta_{\min}$ for a quadrilateral element can be found using the relationship

$$\frac{\delta_{\min}}{R} \geq \frac{c^2}{8R^2} \tag{5}$$

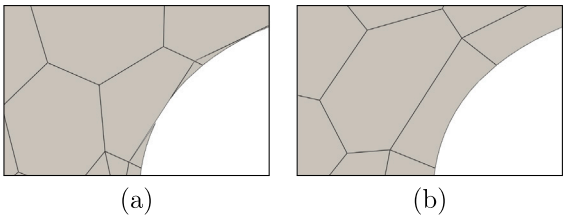

Fig. 8. A mesh with a layer of stretched quadrilaterals: (a) A layer thickness below the value $\delta_{\min}$ given by Eq. (5) leads to self-intersection and thus invalid elements, and (b) A valid mesh is obtained with a value above the minimum thickness.

where R is the radius of curvature and c the length of the straight-sided edge. This value can also be used as a conservative estimate of the mesh size required to ensure validity for convex polygonal elements with four edges or more. Fig. 8 illustrates the application of this criterion in the case of a mesh with a layer of quadrilateral cells near the boundary with values of δ above and below $\delta_{\min}$, with the latter leading to an invalid mesh.

It is worth noting that techniques currently employed in high-order meshing for deforming a straight-sided mesh to accommodate boundary curvature, see for instance Refs. [10,11], could also be implemented using a VEM formulation and applied in this context. However, such VEM implementation is left for future work. Instead, we will discuss polygonal mesh quality measures and their implementation within a mesh quality optimisation process in Section 6.

3.5. Implementation

The various steps of the mesh generation method described in previous sections have been implemented within the open-source code *NekMesh*. A schematic flowchart of the implementation is presented in Fig. 9. The "Remedial steps" box in the flowchart denotes the procedures employed to ensure mesh validity described in Section 3.4.

The implementation performs all the required geometrical queries via the API to the CAD engine, and the communication with the VEM solver is via a simple file-based interface system. The solver is asked to read the information defining the linear mesh (`.mesh`) and the high-order geometrical information on the curved edges (`.nmg`).

The `.mesh` file includes only information about the linear mesh: the vertices and their Cartesian coordinates, the edges with the IDs of the two vertices, and an additional flag showing the corresponding boundary condition. Finally, for the polygonal elements, we first indicate the number of vertices of the polygon and their IDs.

The `.nmg` file, on the other hand, does not have any connectivity details or elements. Instead, it communicates only a minimal amount of geometrical information about the previously populated curved edges. *NekMesh* first provides the ID of the curved edge and the data from the `.mesh` file. Then, for every quadrature point $\overline{x}_j$, determines and writes to the file the following geometrical information with the help of the CAD engine:

- location inside the standard element: $0 \leq t_j \leq 1$,
- Cartesian location: $\overline{x}_j = \overline{x}(t_j)$,
- unit normal to the CAD curve: $\overline{N}$ (pointing inside the curvature), and
- tangent to the CAD curve: $\overline{x}'(t_j)$.

Considering the test ring geometry from Fig. 13 and the VEM solver [8,21] at order $k = 2$, *NekMesh* constructs the combination of quadrature rules automatically as required by the solver: a Gauss–Lobatto rule with $n = 3$, and Gauss rules with $n = 2, 3$. The geometrical information evaluated on the inner circle from the `.nmg` file is displayed in Fig. 10.

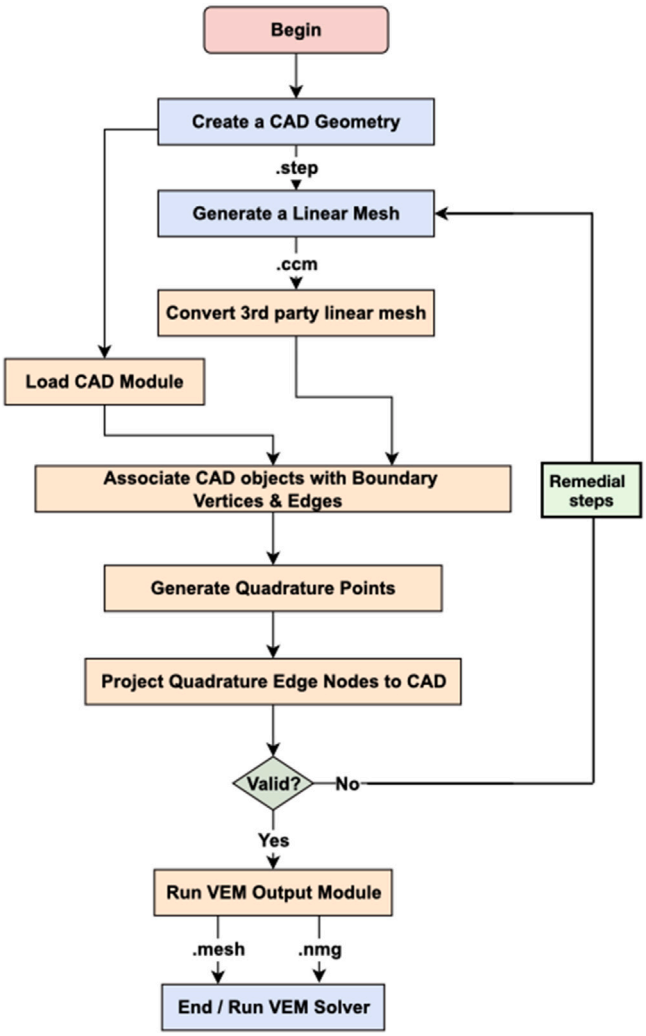

Fig. 9. The workflow of the proposed pipeline form the CAD definition to the curvilinear high-order mesh.

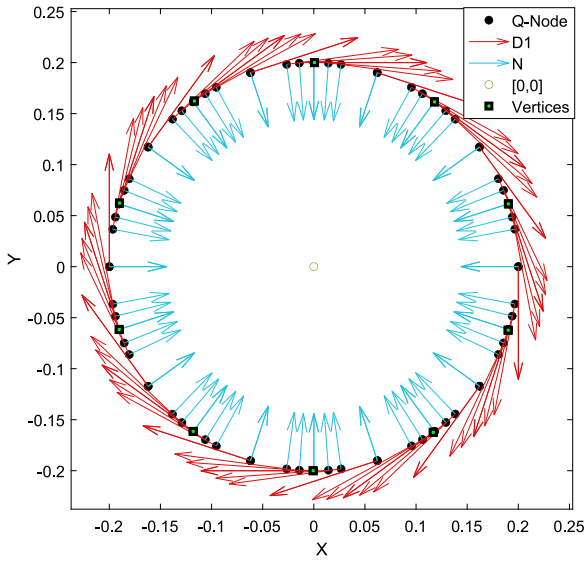

Fig. 10. Visualisation of the high-order geometrical information (with $k = 2$) communicated to the VEM solver via the `.nmg` file.

Our choice of using a simple file-based interface system is informed by the need to provide state-of-the-art VEM solvers with easy and robust access to the high-order curvilinear information without the need to interact with the B-rep of the computational domain.

4. VEM verification

This section aims at illustrating the validity of the meshes generated by the proposed methodology. It will show that the VEM discretisations of the model linear elliptic PDE in a simple domain, a circular ring, achieve the expected rate of convergence in both straight-sided and curvilinear meshes.

From the numerical analysis viewpoint, inaccurate representation of curved boundaries may corrupt the numerical approximation of the discrete solution. Indeed, the error in a numerical solution, ϵ, can be split in two main contributions:

$$\epsilon = \epsilon_f + \epsilon_g,$$

where the error ϵ_f arises due to the discretisation of the functional spaces and the involved (bilinear) forms, and the ϵ_g represents the error on the solution that stems from the approximation of the geometry of the computational domain.

The contribution of the error ϵ_g is negligible when dealing with domains with straight boundaries since piecewise straight segments perfectly match a straight boundary. Therefore the error of a numerical solution is *only* ϵ_f due to the discrete functional spaces used.

However, when the domain is curved and we approximate curved boundaries with piecewise straight segments $\epsilon_g \approx h^2$ where h is the size of the mesh [8]. As a consequence, improving the approximation provided by the discrete functional spaces may not decrease the whole numerical error we have.

In the following we will perform numerical experiments to verify these claims and show that the high-order meshes generated by *NekMesh* combined with the curvilinear virtual element spaces reviewed in Section 2 overcome this issue.

Let Ω denote a domain consisting of a circle of radius 1 centred at the origin, with a circular hole removed at the origin with radius 0.2. We define the right-hand side and the boundary conditions in such a way that the solution of the model elliptic PDE on Ω is the function

$$u(x, y) = \log(x^2 + y^2).$$

We discretise the computational domain Ω in two ways: one with straight-sided edges, referred to as `noGeo`, and one with curved edges, denoted `withGeo`. For each of these mesh types, we construct a sequence of four meshes with decreasing mesh size. Fig. 11 shows an example of a `withGeo` mesh.

The VEM approach reviewed in Section 2 with $k = 2$ is then used to solve this problem. For each of these meshes we compute the errors in the L^2 norm and the H^1 semi-norm.

The trend of these errors is depicted in Fig. 12. We begin by analysing the error in the L^2 norm. For the straight-sided case we know that

$$\epsilon_f \approx h^3 \quad \text{and} \quad \epsilon_g \approx h^2.$$

The geometrical error is dominant, and so we observe a decay of order two of the error. However, when we consider curved meshes using the exact geometry, the error in approximating the geometry decreases faster as $h \to 0$. For instance, assuming regularity of the domain, a second-order curved edge approximation leads to an error $\epsilon_g = O(h^3)$. Furthermore we have $\epsilon_f \approx h^3$ and, as a consequence, the trend of the error is not affected by the geometrical approximation error and we observe an error decay of order 3.

In the case of the H^1 semi-norm error, we obtain the expected error decay of order 2 for both type of meshes, but the absolute value of the error computed with the curved mesh is smaller. Also the better convergence trend observed using the `withGeo` meshes is due to the better approximation of the geometry. Indeed, the total error in the `noGeo` mesh have two contributions with the same order in the H^1 norm, namely

$$\epsilon_f \approx h^2 \quad \text{and} \quad \epsilon_g \approx h^2.$$

Consequently the error trend is not corrupted but its value is affected by two contributions. While in the `withGeo` approach the error due to the geometry is null, the final error is *only* influenced by ϵ_f and, as a consequence, it is smaller.

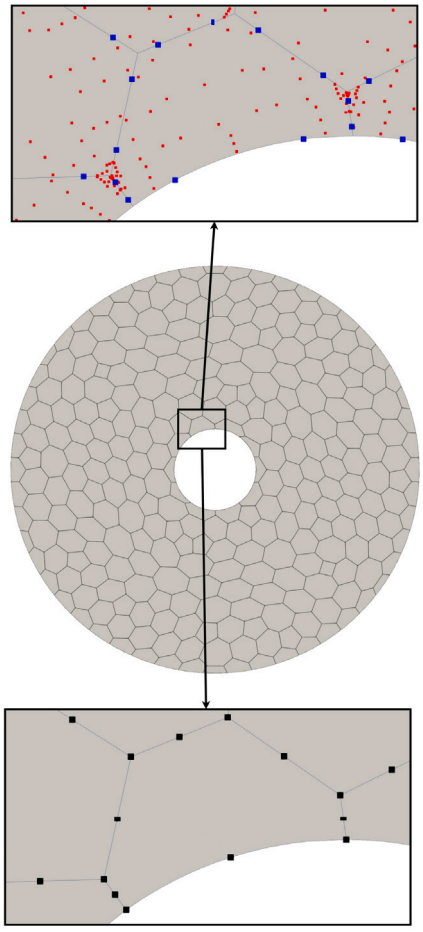

Fig. 11. A curvilinear high-order polygonal mesh of the domain Ω. The mesh of the domain only displays the edges of the polygonal elements. An enlargement of the mesh near the boundary illustrates the additional degrees-of-freedom required for the high-order discretisation. The top window shows the locations of the quadrature points on the edges (blue) and on the interior (red) of the polygonal cells. The bottom window shows the locations of the points used for interpolation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

5. Example of application: A practical 2D geometry

This section illustrates the application of the high-order mesh generation procedure to a computational domain for an automotive aerofoil geometry exhibiting variable curvature along its length. The geometry of the aerofoil is defined by four NURBS curves.

The linear polygonal mesh generated by STAR-CCM+ is depicted in Fig. 14. The mesh resolution has been purposely increased in the regions near the leading and trailing edges of the aerofoil.

Fig. 15 shows the edges of the polygonal mesh together with two enlargements showing the location of the interpolation and quadrature points used in the high-order discretisation.

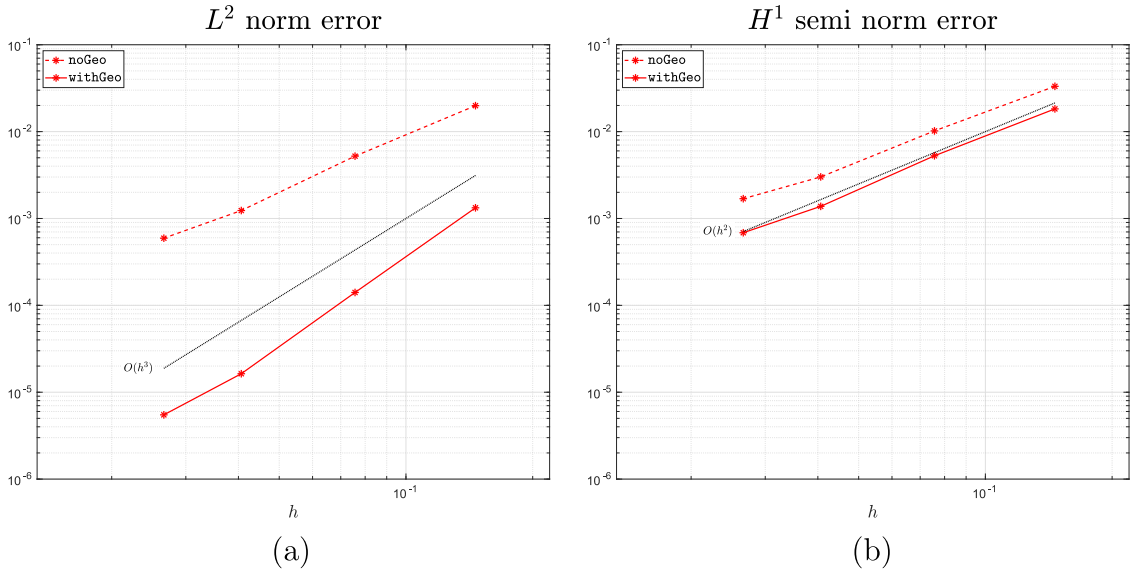

Fig. 12. Mesh convergence of the high-order VEM discretisation: (a) error in the L^2 norm; and (b) error in the H^1 semi-norm.

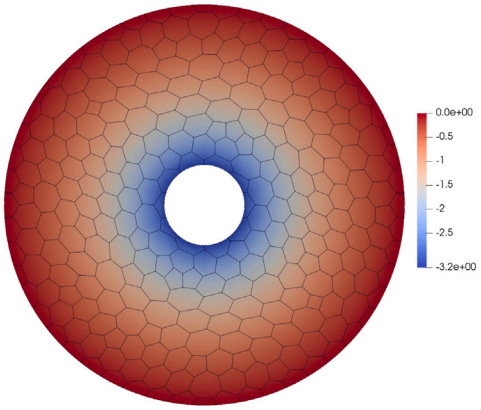

Fig. 13. High-order VEM solution ($k = 2$) of the model linear elliptic PDE with Dirichlet boundary conditions computed on a curvilinear mesh (`withGeo`).

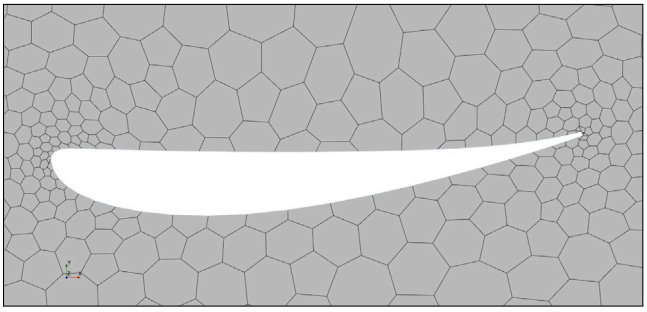

Fig. 14. A straight-sided linear mesh of an automotive aerofoil.

6. Curvilinear polygonal mesh optimisation

This section discusses the assessment of polygonal curvilinear mesh quality, proposes and appraises the use of geometrical mesh quality metrics, and utilises three of these metrics, namely convexity, regularity and isotropy, to define an objective function for mesh quality optimisation. It will also show that the convexity metric can be used to untangle meshes with invalid elements.

Mesh quality of high-order elements is often defined in terms of the distortion induced by an affine mapping with respect to a reference element [26–28], but such affine mappings are not available for polytopes. Non-affine mappings can be obtained via the use of generalised barycentric coordinates [29] but they are more expensive to compute for non-convex polytopes. As an alternative, affine approximations of the mapping have been proposed in Ref. [30].

Reviews of *a priori* criteria of mesh quality for polygonal meshes derived from convergence and stability criteria from the VEM literature have been carried out in Refs. [31–33], while additional considerations of anisotropy have been discussed in Refs. [34,35]. Some of these mesh quality metrics are derived from error analysis of the VEM and require the calculation of a geometrical kernel [36], the region of interior points where all the vertices of the polygon are visible, which is not smooth and expensive to compute. Further, these mesh quality metrics have been proposed for straight-sided polygons and only a few are applicable to curvilinear polygonal elements.

Therefore we favour geometrical criteria of mesh quality that are computable for a curvilinear polygon (or polyhedron), and we adopt the criterion that the "best" shape is a *straight-edged* regular polygon. Along these lines, we seek a mesh of elements with the following geometrical properties: convexity, regularity and isotropy. These are defined in the following sections.

6.1. Convexity

We utilise a simple measure of the convexity of an N-sided polygon as

$$Q_E^C = 4N \tan\left(\frac{\pi}{N}\right) \frac{|\Omega_E|}{|\partial\Omega_E|^2} \tag{6}$$

where

$$|\Omega_E| = \int_{\Omega_E} d\mathbf{x}; \quad |\partial\Omega_E| = \int_{\partial\Omega_E} ds \tag{7}$$

represent the area of the polygon and the length of its boundary, respectively. Note that $0 \le Q_E^C \le 1$ and $Q_E^C = 1$ for a regular polygon. Further the limit case

$$\lim_{N\to\infty} Q_E^C = 4\pi \frac{|\Omega_E|}{|\partial\Omega_E|^2} \tag{8}$$

corresponds to a circle for $Q_E^C = 1$.

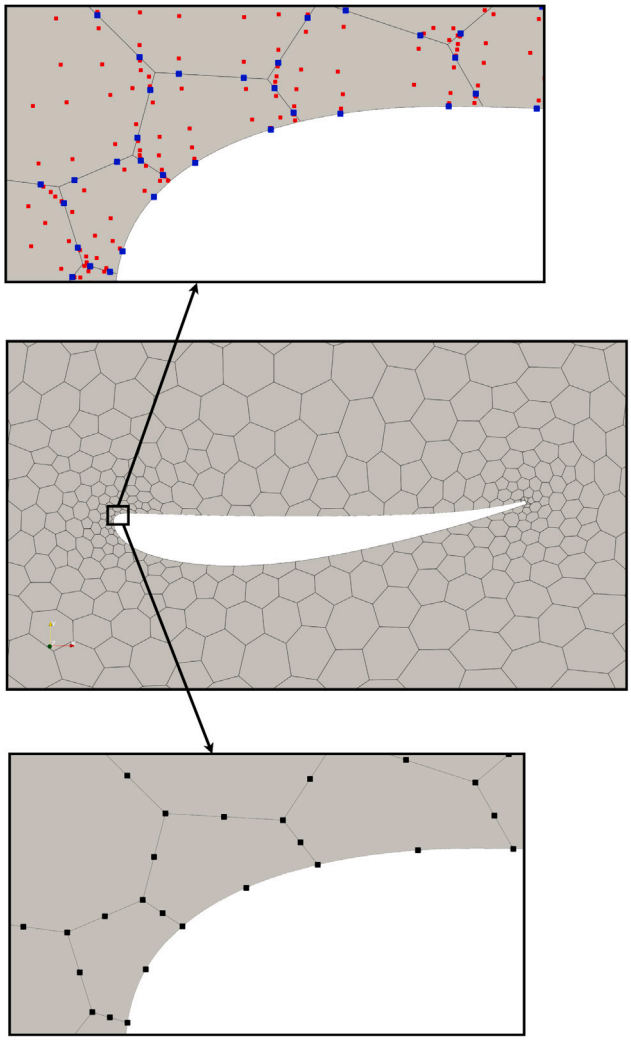

Fig. 15. A curvilinear high-order polygonal mesh of an automotive aerofoil. The mesh of the domain only displays the edges of the polygonal elements. An enlargement of the mesh near the boundary illustrates the additional degrees-of-freedom required for the high-order discretisation. The top window shows the locations of the quadrature points on the edges (blue) and on the interior (red) of the polygonal cells. The bottom window shows the locations of the points used for interpolation. (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

6.2. Regularity

Regularity of a polygon requires the length of its sides to be the same. Denoting by L_e the length of an edge, regularity can be expressed as the ratio of minimum to maximum edge lengths, namely

$$Q_E^R = \frac{\min L_e}{\max L_e} \tag{9}$$

Here we also have $0 \leq Q_E^R \leq 1$, with a value $Q_E^R = 1$ corresponding to equal edge lengths, and values $Q_E^R \approx 0$ indicating the presence of very small edges.

6.3. Isotropy

Here we define the 2×2 inertia matrix, $\mathbf{M}_I$, as

$$\mathbf{M}_I = \frac{1}{|\Omega_E|} \int_{\Omega_E} (\mathbf{x} - \bar{\mathbf{x}})(\mathbf{x} - \bar{\mathbf{x}})^T d\mathbf{x} \in \mathbb{R}^{2\times 2}, \tag{10}$$

where $\bar{\mathbf{x}}$ is the centre of gravity given by

$$\bar{\mathbf{x}} = \frac{1}{|\Omega_E|} \int_{\Omega_E} \mathbf{x}\, d\mathbf{x}. \tag{11}$$

The ratio of the eigenvalues of the inertia matrix represents the imbalance in the mass distribution of the shape with reference to its centre of gravity. Thus an isotropy quality metric can be defined as

$$Q_E^I = \frac{\lambda_{\min}}{\lambda_{\max}}, \tag{12}$$

where $\lambda_{\max}$ and $\lambda_{\min}$ are the maximum and minimum eigenvalues of $\mathbf{M}_I$, respectively. Again we have $0 \leq Q_E^I \leq 1$ and a value $Q_E^I \approx 1$ indicates the mass is evenly distributed and the polygonal shape is isotropic.

6.4. Integral evaluation

Using the Gauss' theorem

$$\int_{\Omega_E} \nabla \cdot \mathbf{f}\, d\mathbf{x} = \int_{\partial\Omega_E} \mathbf{f} \cdot \mathbf{n}\, ds, \tag{13}$$

where s is the arc length of the curved edge and $\mathbf{n}(s)$ is its unit normal, we can reduce the evaluation of the area integrals (7), (11), and (10) to integrals over the boundary. Their respective integrands can be expressed in the general form

$$\nabla \cdot \mathbf{f} = (x_i - \bar{x}_i)^m (x_j - \bar{x}_j)^n \tag{14}$$

and, amongst the various two-dimensional choices available for $\mathbf{f}$, we select

$$\mathbf{f} = \left[\frac{1}{m+1}(x_i - \bar{x}_i)^{m+1}(x_j - \bar{x}_j)^n, 0\right] \tag{15}$$

Denoting $g(s) = \mathbf{f} \cdot \mathbf{n}$ in Eq. (13), the line integral is evaluated as a sum of integrals over the curvilinear edges of the polygon

$$\int_{\partial\Omega_E} \mathbf{f} \cdot \mathbf{n}\, ds, = \sum_e \int_e g_e(s)\, ds \tag{16}$$

where $g_e(s)$ is the value of $\mathbf{f} \cdot \mathbf{n}$ along the edge e. To evaluate the line integral along each edge, the curvilinear edge is represented by a mapping $\mathbf{x}(\xi)$ with $-1 \leq \xi \leq 1$, and the integral approximated through a quadrature of the form

$$\int_e g_e(s)\, ds = \int_{-1}^{1} g_e(s(\xi)) \frac{ds}{d\xi}(\xi)\, d\xi \approx \sum_{j=1}^{n_q} w_j\, g_e(s(\xi_j)) \frac{ds}{d\xi}(\xi_j), \tag{17}$$

where $\xi_1 \ldots, \xi_{n_q}$ are the locations of the quadrature points, $w_1, \ldots, w_{n_q}$ are their respective weights. Here we use a Gaussian quadrature rule with $n_q = k + 2$ so the expression (17) is exact if the integrand $g(s)$ is a polynomial of order k.

6.5. Appraisal of the quality metrics

We design a simple benchmark to assess the variation of the chosen metrics with respect to the motion of the vertices of a polygon. By moving only one of the vertices, we can visualise the elemental mesh quality as a function $Q(x, y)$, where (x, y) are the coordinates of the moving vertex. We will consider two polygons for mesh quality metric assessment, namely a square-shaped octagon with straight edges and a square with quadratic edges as representative of linear and high-order polygonal elements, respectively. Both elements have the same number of nodes for consistency of comparison.

6.5.1. Linear polygonal element quality metrics

The linear polygon, shown in Fig. 16, is a square-shaped octagon with equal sides. Seven of its vertices, represented by black dots in the figure, are fixed and the eighth vertex (shown as a white dot) is moved to points within the T-shaped region. The quality metrics are evaluated at each point to obtain the contour map of their values. The

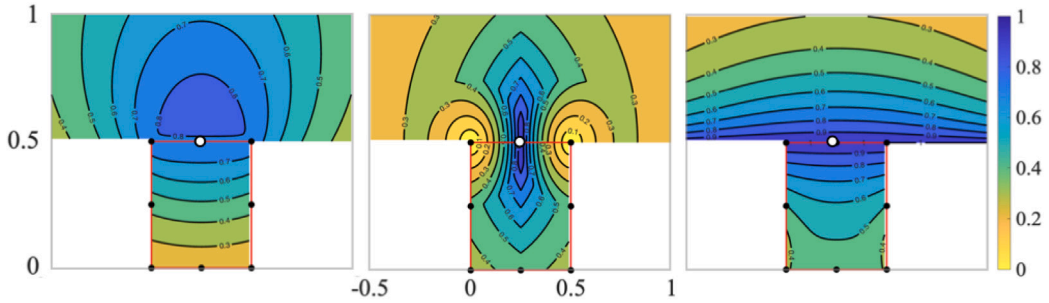

Fig. 16. Contour maps, from left to right, of convexity, regularity and isotropy quality metrics for a polygonal element with five straight edges. The contours represent the values of the relevant quality metric of the polygon as the vertex highlighted as a white dot moves in the T-shaped region.

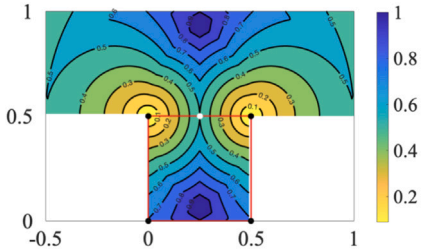

Fig. 17. An illustration of the presence of multiple maxima of the regularity metric when the ratio of side lengths is large. The elemental polygon is a square-shaped pentagon in this example.

maps are consistent with the symmetry of the problem and show that the variation of all the metrics with the position of the moving vertex is smooth.

The convexity metric map exhibits a clear maximum value, but also a "flat" region at $y = 0.5$. The flat region appears there because the motion of the free vertex along that line does not change the values of the area and boundary length of the polygon. The presence of such regions may negatively affect the convergence of iterative optimisation methods.

The regularity map has two axes of symmetry: the lines $x = 0.25$ and $y = 0.5$. A maximum of this metric occurs at their intersection point $(x, y) = (0.25, 0.5)$, but the region around this point is flat and narrow. Furthermore, large values of the ratio of the lengths may lead to the presence of multiple extrema of the quality metric, as illustrated in the case of a square-shaped pentagon shown in Fig. 17. Both these issues may result on slow, or lack of, convergence to a local optimum for some optimisation methods.

The maxima of the isotropy metric map occur at $y = 0.5$ for all the values of the x-coordinate. Similarly to the convexity metric, this leads to a flat region there since both metrics depend on the shape of the polygon only. Intuitively, we can add points to the side of a polygon that will generate a polygon with more sides but having the same area and perimeter length. However, the effect on iterative optimisation procedures is more severe here because all the points located at $y = 0.5$ are maxima of the isotropy metric.

6.5.2. High-order polygonal element quality metrics

The quadratic polygonal element, shown in Fig. 18, is a square-shaped octagon with equal curvilinear sides. Seven of its nodes, represented by black dots for the four vertices and blue for three mid-side nodes in the figure, are fixed and the eighth node (shown as a white dot) is moved, as previously, to points within the T-shaped region. The main difference with the previous case is that now the motion of the mid-side node induces curvature in the associated curvilinear edge and thus the values of the calculated lengths and areas will be slightly different for the quadratic element.

The comments made for the convexity and regularity metrics in Section 6.5.1 apply here too as the differences in length and area due to the curved edge are not large, but the quality in this case decreases faster closer to the edges. This is more evident in the case of the isotropy metric where its values become very small near the vertices at the bottom of the figure. This due to the smallest eigenvalue of the inertia matrix reaching values close to zero, even negative. These instances are an indication that self-intersection occurs. This is illustrated by the dashed line representing the curved edge in the figure, which shows the edge exiting the region of validity when its mid-node gets close to the farthest vertices. Unfortunately, any hope of using this as an identifier for element validity is in vain as there cases where the element self-intersects and the minimum eigenvalue is small but still positive.

To alleviate some of the negative aspects of the use of these metrics in the context of mesh optimisation discussed above, we propose to investigate a linear combination of the three metrics in the next section.

6.6. Mesh quality optimisation

We adopt a measure of elemental mesh quality as a linear combination of the three metrics, namely

$$Q_E = aQ_E^C + bQ_E^R + (1 - a - b)Q_E^I \quad 0 \le a, b \le 1, \quad a + b \le 1 \tag{18}$$

so that $0 \le Q_E \le 1$ and the element exhibits the best quality when $Q_E \approx 1$. To evaluate and optimise the quality of the mesh we define an objective function of mesh quality, Q, as the average of elemental qualities, i.e.

$$Q = \frac{1}{N_E} \sum_{E=1}^{N_E} Q_E \tag{19}$$

where N_E is the number of elements in the mesh. The variables of the problem are the coordinates of the interior nodes while the boundary nodes are fixed. We seek a maximum value of Q to obtain their optimal location. Given that the objective function, Q, may not be differentiable, or even continuous, we use a derivative-free optimiser NLOPT [37] with the algorithm PRAXIS [38]. Convergence of the optimisation process is assumed when the maximum displacement, $\delta_{\max}$, amongst all the nodes within consecutive iterations is small, i.e. $\delta_{\max}/L_{\max} \le \epsilon$, where $L_{\max} = \max_{e \in \Omega} L_e$. Here we have used $\epsilon = 10^{-8}$.

The next step is an attempt to find values of the parameters a and b in Eq. (18) that would lead to optimal values of the global mesh quality Q defined in Eq. (19). As a way of illustration, we will consider a simple

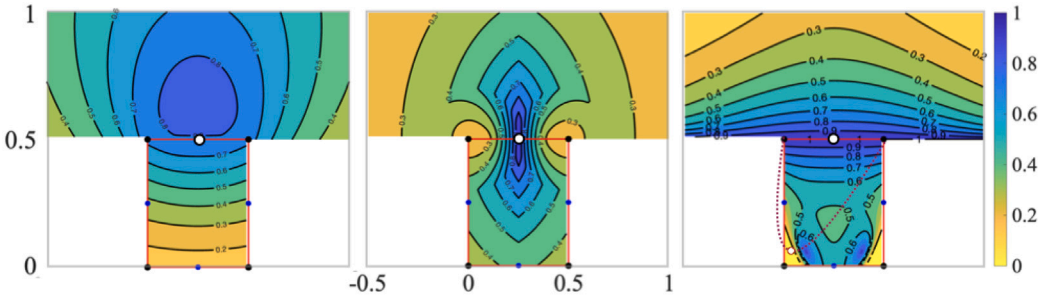

Fig. 18. Contour maps, from left to right, of convexity, regularity and isotropy quality metrics for a quadratic quadrilateral element with four straight edges and a curved one that changes through the motion of its mid-point highlighted by the white dot. The contours represent the values of the relevant quality metric of the high-order quadrilateral as the white dot moves in the T-shaped region.

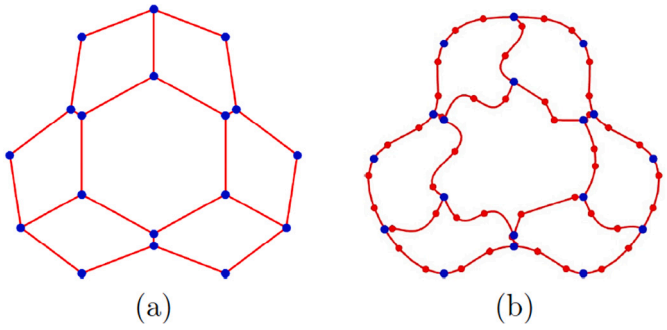

Fig. 19. Initial meshes in the optimisation process: (a) Straight-sided mesh; (b) Initial high-order boundary-conforming mesh.

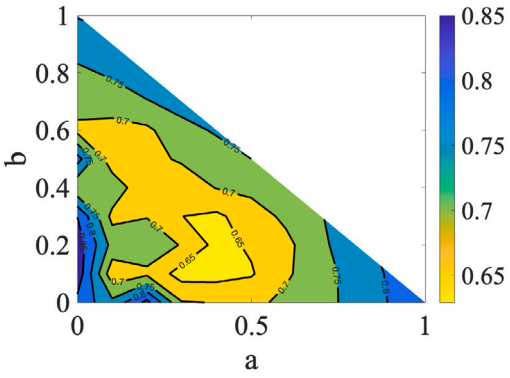

Fig. 20. Colour map of the mesh quality metric $Q(a, b)$ defined in Eq. (19) for the optimised meshes obtained by varying the parameters a and b in the combined elemental mesh metric (18).

test case where a straight-sided polygonal mesh shown in Fig. 19(a) is transformed into a curvilinear boundary-conforming mesh with internal straight sides first, and then the interior nodes on these sides are slightly displaced in a random fashion to obtain the mesh depicted in Fig. 19(b). This mesh is the starting point for the mesh optimisation process.

The initial mesh is optimised for a set of (a, b) values to obtain an optimal value of the mesh quality metric $Q(a, b)$ which is displayed as a contour map in Fig. 20. The map of the metric $Q(a, b)$ exhibits two regions of optimality in the vicinity of $a = 0$ and $a = 1$. Values around $a = 1$ are to be preferred to ensure that the contribution of convexity is larger than those of regularity and isotropy that may introduce multiple extrema as discussed in Section 6.5. Furthermore, numerical results not reported herein indicate that faster convergence is achieved, independently of the polynomial order chosen, when $a > 0.6$.

Examples of high-order meshes ($k = 3$) optimised using this approach are shown in Fig. 21. The mesh in Fig. 21(a) corresponds to values $(a, b) = (1, 0)$, i.e. only the convexity mesh quality metric is used. Convexity is clearly achieved and we can also observe that the internal edges are almost straight, but the distribution of nodes along the edges is far from uniform. To encourage a more uniform nodal distribution is therefore advisable to increase the contribution of the other metrics. The best results were obtained with a weight $b = 0.1$ or the regularity metric. To show the effect of the isotropy metric we show the meshes obtained with weights $a = 0.8, 0.7, 0.6$ in Figs. 21(b,c,d), respectively. The "optimal" mesh is obtained for values $(a, b) = (0.8, 0.1)$ and is shown in Fig. 21(b).

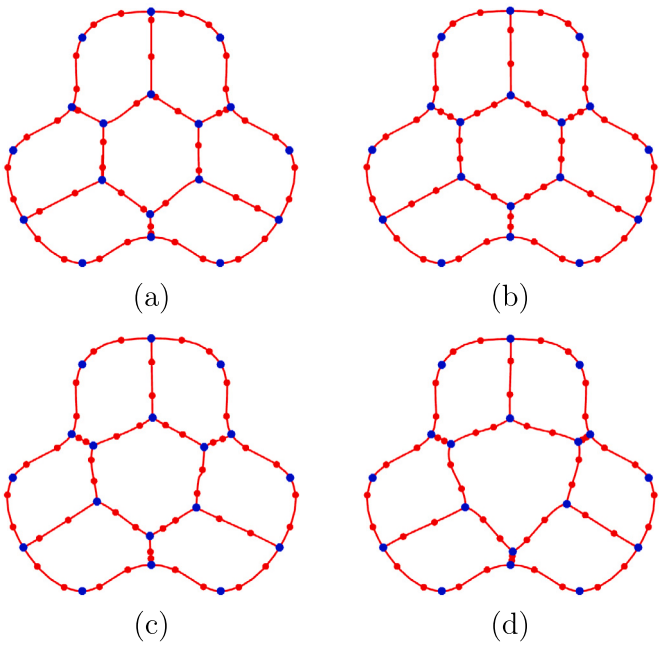

Fig. 21. Optimised meshes. The values of the parameters a and b in Eq. (18) used in the optimisation are: (a) $a = 1$, $b = 0$; (b) $a = 0.8$, $b = 0.1$; (c) $a = 0.7$, $b = 0.1$; (d) $a = 0.6$, $b = 0.1$.

6.6.1. *Dealing with tangled meshes*

Invalid meshes due to element overlap or self-intersection. Identifying where these occur is difficult, but their presence can be detected

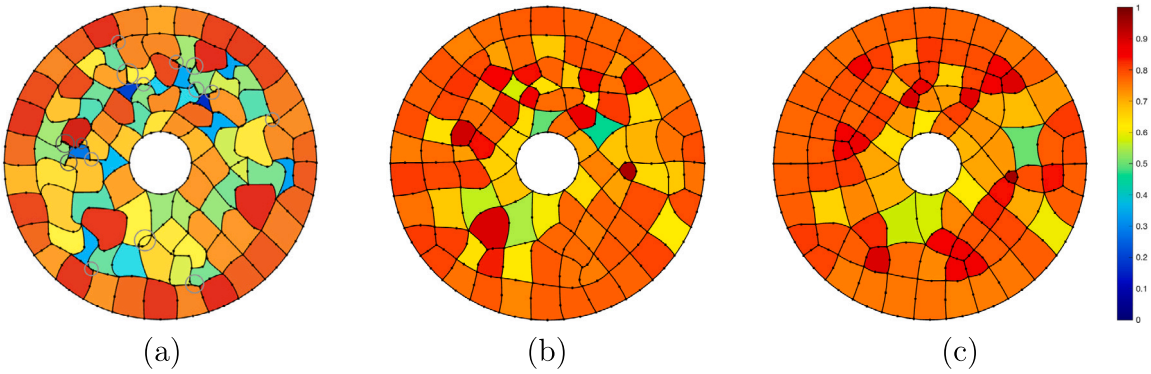

Fig. 22. Ring shape: (a) Initial tangled mesh with the grey circles indicating where self-intersections occur, (b) mesh untangled after 10 optimisation iterations, and (c) Optimised mesh. The colour legend on the right represents the values of the quality metric Q_E^C given by Eq. (8).

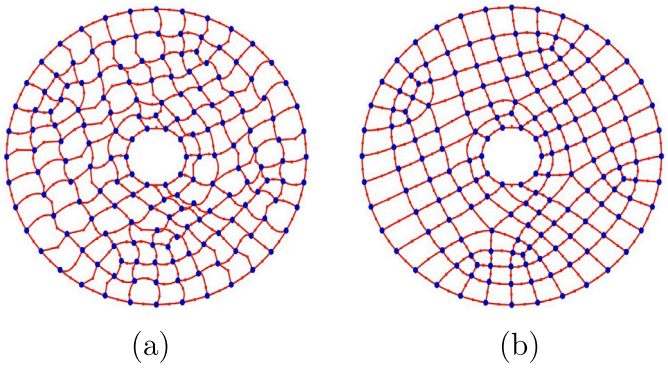

Fig. 23. Ring shape: (a) Initial distorted mesh and (b) Optimised mesh.

by comparing the values of the area of the domain calculated as an integral over the boundary sides of the high-order mesh and the same are calculated as the sum of the elemental areas. The evaluation of these areas follows the methodology described in Section 6.4. Significant differences in the values of these areas indicate the mesh is tangled and thus invalid for simulation.

One of the mesh quality measures discussed in Section 6.6, namely the convexity metric Q_E^C given by Eq. (6), turns to be very effective at untangling meshes for large values of N. The other two are ineffective for untangling purposes. We illustrate the suitability of the convexity metric by optimising a very distorted mesh of a ring that self-intersects using the value of this metric for $N \to \infty$, i.e. Eq. (8), that achieves a maximum value $Q_E^C = 1$ for a circle and thus encourages elements to untangle by deforming towards a circle. Fig. 22(a) depicts the initial invalid mesh, which is untangled after only 10 iterations of the optimisation procedure as shown in Fig. 22(b). Fig. 22(c) shows the mesh at the end of the optimisation process.

6.6.2. Effect of mesh distortion on solution accuracy

To verify the effectiveness of the mesh optimisation, we analyse how the quality of the elements affects the accuracy of the solution. We consider two high-order meshes for the ring geometry depicted in Fig. 23. A distorted mesh ($k = 3$), shown in Fig. 23(a), is first generated following the method described in Section 3, and its quality then degraded by randomly displacing the interior edge nodes to obtain a "bad" initial mesh. The mesh in Fig. 23(b) is the optimised mesh obtained using $(a, b) = (0.8, 0.1)$.

We consider the model problem given by Eq. (1) and set the right-hand side and the boundary conditions so that the exact solution is

$$u(x, y) = \sin(8\pi x)\,\sin(8\pi y)\,. \tag{20}$$

We solve this problem on both meshes using the VEM approach described in Section 2, and in more detail along with alternative formulations for curvilinear VEM in Refs. [7,39], and compute the L^2 error

$$\max_{E\in\Omega_h} \int_E (u - \hat{u})^2 dE \tag{21}$$

of the numerical solution $\hat{u}$ with respect to the exact solution u. The calculated errors are 1.17×10^{-11} and 4.29×10^{-12} for the distorted and optimised meshes, respectively. Mesh optimisation leads to a reduction of the solution error, and the values of both errors are low indicating that the VEM is robust against mesh distortion, as also shown elsewhere [40].

6.6.3. Mesh optimisation of an aerofoil computational domain

This section describes the application of the methodology to a more complex geometry representing the computational domain for an aerofoil.

We consider the optimisation of a high-order mesh ($k = 3$), shown in Fig. 24, that has been deliberately distorted to reduce its quality by introducing random nodal displacements to a straight-sided high-order mesh. Fig. 24(b), which display an enlargement of the mesh near the aerofoil, gives a good indication of the quality of the mesh. The colour map shows a sizeable number of "bad" elements and large disparity in element quality across the mesh, and the circles indicate where edges intersect leading to invalid elements.

The quality of this initial mesh is optimised to achieve a maximum of the averaged elemental quality measure (19). The main contributor to the quality measure is the convexity metric (6) where N is take to be the number of edges in a curvilinear polygonal element with a view to encourage them to be close to straight-edged polygons. The optimised high-order mesh is shown in Fig. 25 that does not contain any self-intersections, thus reinforcing the suitability of the convexity metric to untangle meshes as discussed in Section 6.6.1. The colouring of the elements show the quality of the majority of interior elements is significantly improved with the lowest values of elemental quality occurring near the boundary. This is partly due to the fixed position of the boundary nodes limiting the degrees of freedom in the elements adjacent to the boundary. However, the choice of a target metric that aims at improving the average mesh quality results in an evenly distributed elemental mesh quality with a large number of elements of medium to low quality.

To reduce large differences of quality in adjacent elements and achieve a shift towards higher mesh quality across the domain, we propose an alternative target quality measure that aims at increasing the number of elements close to best quality, i.e. $Q_E = 1$, and is given by

$$Q = \sum_{E=1}^{N_E} (Q_E - 1)^2 \tag{22}$$

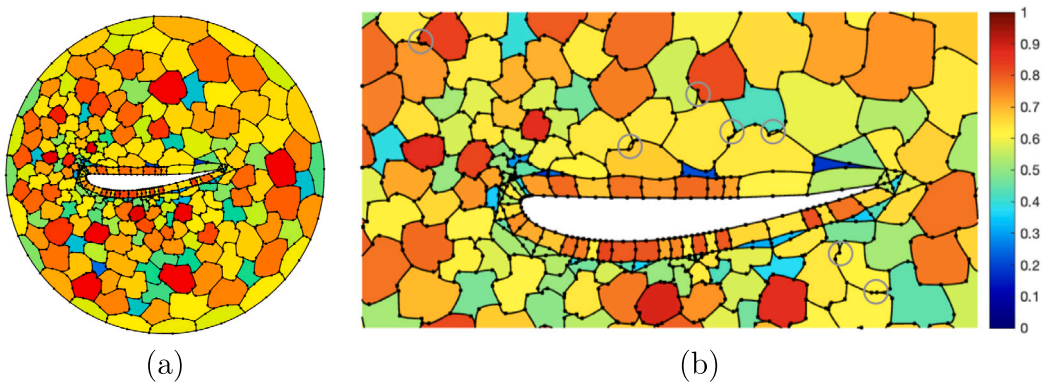

Fig. 24. Initial distorted high-order mesh for an aerofoil geometry: (a) full mesh, and (b) enlargement near the aerofoil with circles indicating the location of tangled elements. The colour map legend gives the values of the elemental mesh quality metric (18). (For interpretation of the references to colour in this figure legend, the reader is referred to the web version of this article.)

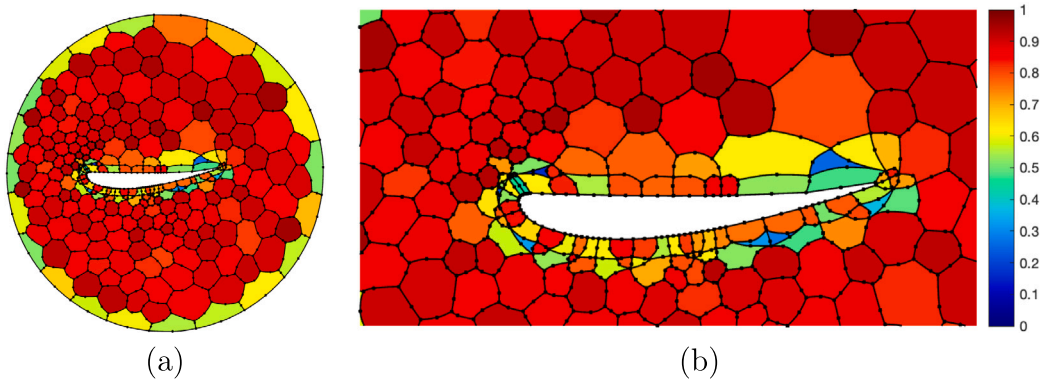

Fig. 25. Optimised high-order mesh for an aerofoil geometry obtained using the averaged elemental quality (19) as a target: (a) full mesh, and (b) enlargement near the aerofoil.

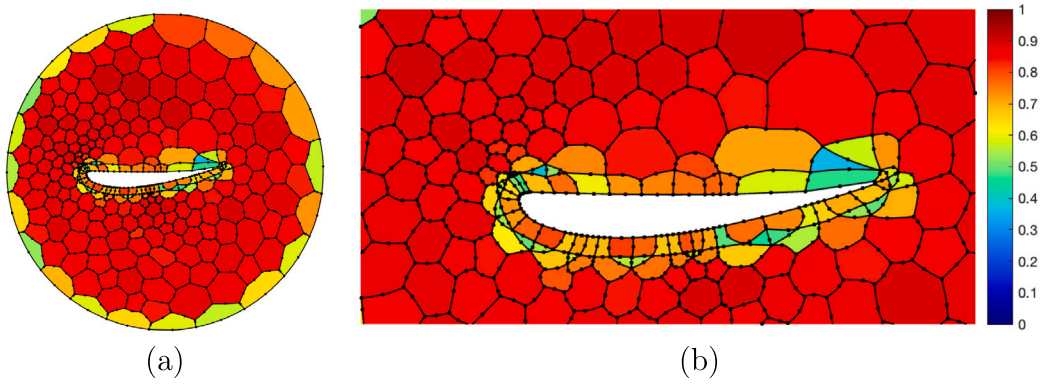

Fig. 26. Optimised high-order mesh for an aerofoil geometry obtained using the elemental quality (22), that favours an evenly distribution, as a target: (a) full mesh, and (b) enlargement near the aerofoil.

The resulting optimised mesh is shown in Fig. 26. The quality of the interior elements is again nearly optimal, but now we have achieved an improved of elemental quality near the boundary and a shift towards higher values of elemental mesh quality across the domain.

The optimisation method is not very efficient but converges monotonically towards the "optimal" mesh. Fig. 27 shows the evolution of the quality metric Q, evaluated via Eq. (22), versus computer time. The large number of iterations required and associated computer cost is impractical and calls for the use of alternative optimisers that are more efficient PRAXIS, but the version in NLOPT [37] employed here is suitable for the purposes of this proof-of-concept work due to its modularity, ease of implementation, and ability to deal with non-smooth target functions.

To get a better comparison of the performance of the quality measures (19) and (22) we compare their histograms of mesh quality with that of the initial mesh in Fig. 28. The histogram of the initial distorted mesh in Fig. 28(a) shows a large spread of values of elemental mesh quality with an average quality $Q \approx 0.65$. The histogram for the

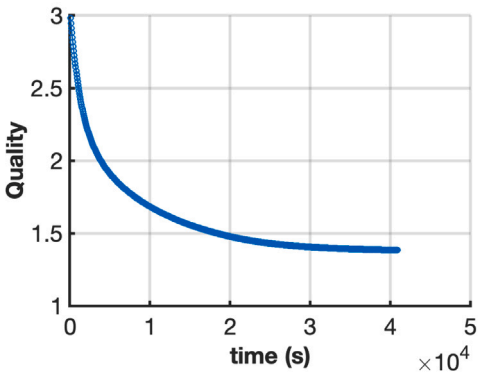

Fig. 27. Evolution of the mesh quality metric Q given by Eq. (22) versus computing time.

mesh optimised according to the averaged quality target (19), shown in Fig. 28(b), indicates a clear shift towards higher values of quality, but still retains a significant proportion of low-quality elements. The histogram of Fig. 28(c) shows that the target quality measure (22) is more effective since the number of elements of low quality is reduce whilst that of high quality is increased.

7. Conclusions and further work

We have proposed a proof-of-concept for a polygonal high-order curvilinear mesh generator for the Virtual Element Method (VEM) for arbitrary geometries defined through a standard CAD B-Rep of the domain. The realisation is that the interpolation and integration of functions within a VEM discretisation only requires the definition of a set of boundary, interface and internal points. This allows us to interpret the problem of finding the location of these interpolation and integration points as geometrical inquiries to a B-Rep of the computational domain, performed via an interface to CAD libraries.

As a consequence, we can adopt an *a posteriori* approach to high-order curvilinear mesh generation for the VEM. The starting point of the process is the generation of a straight-sided mesh in STAR-CCM+. The next step is to use the CAD API implemented in *NekMesh* to reconstruct the CAD information from STEP files and, according to the user-defined polynomial order and quadrature rules, to parametrically project the quadrature nodes onto the curved geometry. Finally, *NekMesh* communicates with the VEM solver through a two-file interface system: one for the linear mesh with its connectivity and a second for the geometrical information at the quadrature nodes. Using an exact solution of a linear elliptic PDE on a ring geometry we show that the generated high-order curvilinear meshes for the VEM are valid, accurately reproduce the analytical solution, and converge at the expected rate as the mesh size is decreased.

We have appraised three geometrical metrics of curvilinear polygonal mesh quality, namely convexity, regularity and isotropy, highlighted some of the difficulties associated in their use for mesh quality optimisation, and proposed a suitable combination of these quality metrics for mesh enhancement. We concluded that the convexity metric should be the principal contributor not only because it is better at improving mesh quality than the other two, but also due its ability to untangle invalid meshes. The role of the other two metrics, regularity and isotropy, is to compensate for some of the slight deficiencies of the convexity metric.

In assessing the effect of mesh distortion on simulation accuracy we observed that the simulation errors calculated in the non-optimised and optimised meshes were low indicating that VEM simulations are robust and relatively insensitive to mesh distortion, and they showed that the use of mesh optimisation leads to a reduction of the error.

An example of a more complex geometry representing the computational domain for an aerofoil provided further evidence of the untangling ability of the convexity metric. We also used this benchmark to appraise two alternative ways of constructing the target mesh quality measure for optimisation, leading to the choice of a measure that increases the number of elements of high elemental quality.

Since the ability to improve the quality of the mesh near the boundary is restricted at present because the boundary nodes are fixed, allowing these nodes to move on the boundary as described in Refs. [10,11] would be a desirable modification to incorporate in the proposed methodology.

The performance of the derivative-free optimiser employed here is poor and thus not suitable for large 2D meshes, let alone for use in 3D. Employing more efficient optimisers or elasticity-based methodologies to minimise deformation from the linear element are options to be considered for future work.

Further enhancements in mesh quality could be achieved by incorporating mesh modification techniques, such as mesh refinement, as proposed for instance in Ref. [41] for polytopal meshes.

We believe that this work has laid a strong foundation for the extension of the methodology to the generation of 3D curvilinear polyhedral meshes. However, in addition to the optimisation performance issues highlighted above, devising quadrature rules suitable for curvilinear polyhedral is an area of current active development. This represents a technical bottleneck that must be addressed before we can extend the proposed methodology to 3D geometries.

CRediT authorship contribution statement

Kaloyan S. Kirilov: Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Conceptualization. **Jingtian Zhou:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Conceptualization. **Joaquim Peiró:** Writing – review & editing, Writing – original draft, Methodology, Funding acquisition, Conceptualization. **Mashy Green:** Writing – review & editing, Software, Methodology. **David Moxey:** Writing – review & editing, Software, Methodology, Funding acquisition, Conceptualization. **Lourenço Beirão da Veiga:** Writing – review & editing, Methodology. **Alessandro Russo:** Writing – review & editing, Methodology. **Franco Dassi:** Writing – review & editing, Writing – original draft, Validation, Software, Methodology, Conceptualization.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgements

This project has received funding from the European Union's Horizon 2020 research and innovation programme under the Marie Skłodowska-Curie grant agreement No 955923. David Moxey, Joaquim Peiró and Mashy Green acknowledge funding from EPSRC, United Kingdom under grant EP/R029423/1. David Moxey acknowledges support from the Royal Academy of Engineering under their Research Chair scheme.

Data availability

Data will be made available on request.

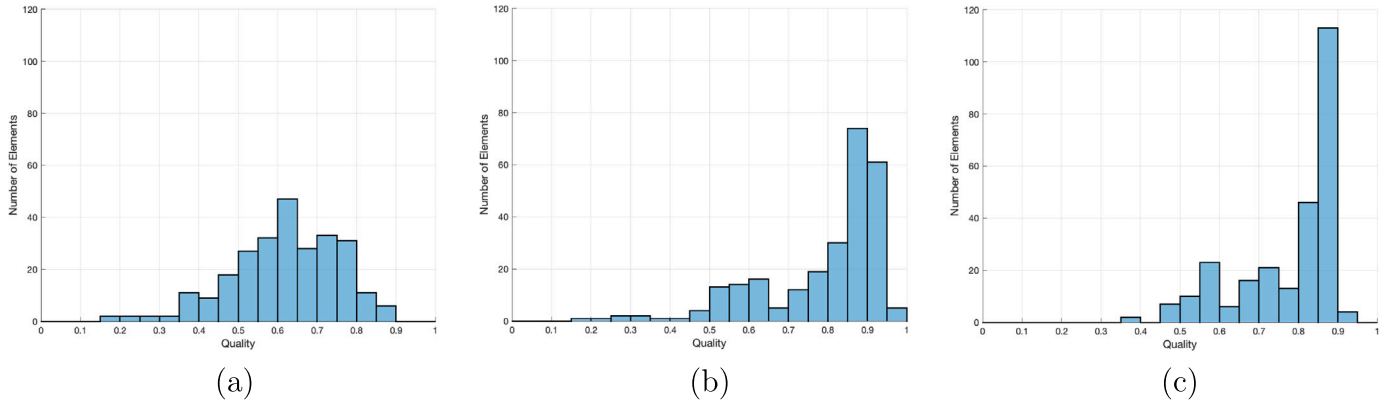

Fig. 28. Histograms of mesh quality: (a) Initial distorted mesh, (b) mesh optimised using (19), and (b) mesh optimised using (22).

References

[1] Versteeg H, Malalasekera W. An Introduction to Computational Fluid Dynamics: The Finite Volume Method. second ed.. Pearson; 2007.

[2] Di Pietro D, Droniou J. The Hybrid High-Order Method for Polytopal Meshes. Springer; 2020, http://dx.doi.org/10.1007/978-3-030-37203-3.

[3] Moxey D, Green M, Sherwin S, Peiró J. An isoparametric approach to high-order curvilinear boundary-layer meshing. Comput Methods Appl Mech Engrg 2015;283:636–50. http://dx.doi.org/10.1016/j.cma.2014.09.019.

[4] Cottrell JA, Hughes TJR, Bazilevs Y. Isogeometric Analysis: Toward Integration of CAD and FEA. Wiley; 2009.

[5] Cangiani A, Dong Z, Georgoulis EH, Houston P. hp-Version Discontinuous Galerkin Methods on Polygonal and Polyhedral Meshes. Springer; 2017, http://dx.doi.org/10.1007/978-3-319-67673-9.

[6] Cicuttin M, Ern A, Pignet N. Hybrid High-Order Methods – A Primer with Applications to Solid Mechanics. Springer; 2021, http://dx.doi.org/10.1007/978-3-030-81477-9.

[7] Beirão da Veiga L, Russo A, Vacca G. The virtual element method with curved edges. ESAIM Math Model Numer Anal 2019;53(2):375–404.

[8] Dassi F, Fumagalli A, Mazzieri I, Scotti A, Vacca G. A virtual element method for the wave equation on curved edges in two dimensions. J Sci Comput 2022;90(50).

[9] Ferguson JA, Kópházi J, Eaton MD. NURBS enhanced virtual element methods for the spatial discretization of the multigroup neutron diffusion equation on curvilinear polygonal meshes. J Comput Theor Transp 2022;51(4):145–204. http://dx.doi.org/10.1080/23324309.2022.2103150.

[10] Turner M, Peiró J, Moxey D. Curvilinear mesh generation using a variational framework. Computer-Aided Des 2018;103:73–91.

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

[12] Beirão da Veiga L, Brezzi F, Cangiani A, Manzini G, Marini LD, Russo A. Basic principles of virtual element methods. Math Models Methods Appl Sci 2013;23(1):199–214.

[13] Beirão da Veiga L, Brezzi F, Marini LD, Russo A. The Hitchhiker's guide to the virtual element method. Math Models Methods Appl Sci 2014;24(08):1541–73.

[14] Chin EB, Sukumar N. Scaled boundary cubature scheme for numerical integration over planar regions with affine and curved boundaries. Comput Methods Appl Mech Engrg 2021;380:113796.

[15] Sommariva A, Vianello M. Gauss–Green cubature and moment computation over arbitrary geometries. J Comput Appl Math 2009;231(2):886–96.

[16] Stroud I. Boundary Representation Modelling Techniques. Springer; 2006.

[17] Green M, Kirilov K, Turner M, Marcon J, Eichstädt J, Laughton E, Cantwell C, Sherwin S, Peiró J, Moxey D. NekMesh: An open-source high-order mesh generation framework. Comput Phys Comm 2024;298:109089. http://dx.doi.org/10.1016/j.cpc.2024.109089.

[18] Capgemini Engineering. Open cascade. 2025, URL https://www.opencascade.com. [Last Accessed July 2025].

[19] ITI Global. CADfix. 2025, URL https://www.iti-global.com/cadfix. [Last Accessed July 2025].

[20] Siemens. STAR-CCM+. Last accessed July 2025, URL https://www.plm.automation.siemens.com/global/en/products/simcenter/STAR-CCM.html.

[21] Dassi F. Vem++, a C++ library to handle and play with the virtual element method. Numer Algorithms 2025. http://dx.doi.org/10.1007/s11075-025-02059-z.

[22] OpenFOAM. API guide: ccmToFoam.C file reference. Last accessed July 2025, URL https://www.openfoam.com/documentation/guides/latest/api/ccmToFoam_8C.html.

[23] ISO. ISO 10303-21:2016 Industrial automation systems and integration – Product data representation and exchange – Part 21: Implementation methods: Clear text encoding of the exchange structure. International Organization for Standardization; 2016, The STEP standard.

[24] Bentley JL. Multidimensional binary search trees used for associative searching. Commun ACM 1975;18(9):509–17.

[25] Karniadakis G, Sherwin S. Spectral/hp element methods for computational fluid dynamics. second ed.. Oxford University Press; 2013.

[26] Knupp P. Introducing the target-matrix paradigm for mesh optimization via node-movement. Eng Comput 2012;28(4):419–29. http://dx.doi.org/10.1007/s00366-011-0230-1.

[27] Gargallo-Peiró A, Roca X, Peraire J, Sarrate J. Distortion and quality measures for validating and generating high-order tetrahedral meshes. Eng Comput 2015;31(3):423–37. http://dx.doi.org/10.1007/s00366-014-0370-1.

[28] Dobrev V, Knupp P, Kolev T, Mittal K, Tomov V. The target-matrix optimization paradigm for high-order meshes. SIAM J Sci Comput 2019;41(1):B50–68. http://dx.doi.org/10.1137/18M1167206.

[29] Floater M. Generalized barycentric coordinates and applications. Acta Numer 2015;24:161–214. http://dx.doi.org/10.1017/S0962492914000129.

[30] Huang W, Wang Y. Anisotropic mesh quality measures and adaptation for polygonal meshes. J Comput Phys 2020;410:109368. http://dx.doi.org/10.1016/j.jcp.2020.109368.

[31] Attene M, Biasotti S, Bertoluzza S, Cabiddu D, Livesu M, Patanè G, Pennacchio M, Prada D, Spagnuolo M. Benchmarking the geometrical robustness of a virtual element Poisson solver. Math Comput Simulation 2021;190:1392–414. http://dx.doi.org/10.1016/j.matcom.2021.07.018.

[32] Sorgente T, Prada D, Cabiddu D, Biasotti S, Patanè G, Pennacchio M, Bertoluzza S, Manzini G, Spagnuolo M. VEM and the mesh. In: Antonietti PF, Beirão da Veiga Lc, Manzini G, editors. The virtual element method and its applications. Springer International Publishing; 2022, p. 1–57. http://dx.doi.org/10.1007/978-3-030-95319-5_1.

[33] Sorgente T, Biasotti S, Manzini G, Spagnuolo M. A survey of indicators for mesh quality assessment. Comput Graph Forum 2023;42(2):461–83. http://dx.doi.org/10.1111/cgf.14779.

[34] Weisser S. Anisotropic polygonal and polyhedral discretizations in finite element analysis. ESAIM: M2AN 2019;53(2):475–501. http://dx.doi.org/10.1051/m2an/2018066.

[35] Berrone S, D'Auria A. A new quality preserving polygonal mesh refinement algorithm for polygonal element methods. Finite Elem Anal Des 2022;207(103770).

[36] Sorgente T, Biasotti S, Spagnuolo M. Polyhedron kernel computation using a geometric approach. Comput Graph 2022;105:94–104. http://dx.doi.org/10.1016/j.cag.2022.05.001.

[37] Johnson SG. The NLopt nonlinear-optimization package. 2007, https://github.com/stevengj/nlopt.

[38] Brent R. Algorithms for minimization without derivatives. Prentice-Hall; 1972.

[39] Beirão da Veiga L, Brezzi F, Marini L, Russo A. Polynomial preserving virtual elements with curved edges. Math Models Methods Appl Sci 2020;30(08):1555–90.

[40] Beirão da Veiga L, Dassi F, Russo A. High-order virtual element method on polyhedral meshes. Comput Math Appl 2017;74(5):1110–22.

[41] Sorgente T, Vicini F, Berrone S, Biasotti S, Manzini G, Spagnuolo M. Mesh optimization for the virtual element method: How small can an agglomerated mesh become? J Comput Phys 2025;521:113552. http://dx.doi.org/10.1016/j.jcp.2024.113552.