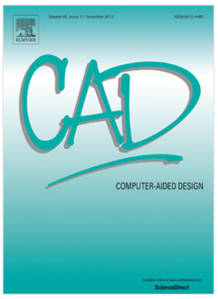
Journal Pre-proof



The Generation of 3D Surface Meshes for NURBS-Enhanced FEM

Xi Zou, Sui Bun Lo, Ruben Sevilla, Oubay Hassan, Kenneth Morgan

PII: S0010-4485(23)00185-9
DOI: https://doi.org/10.1016/j.cad.2023.103653
Reference: JCAD 103653

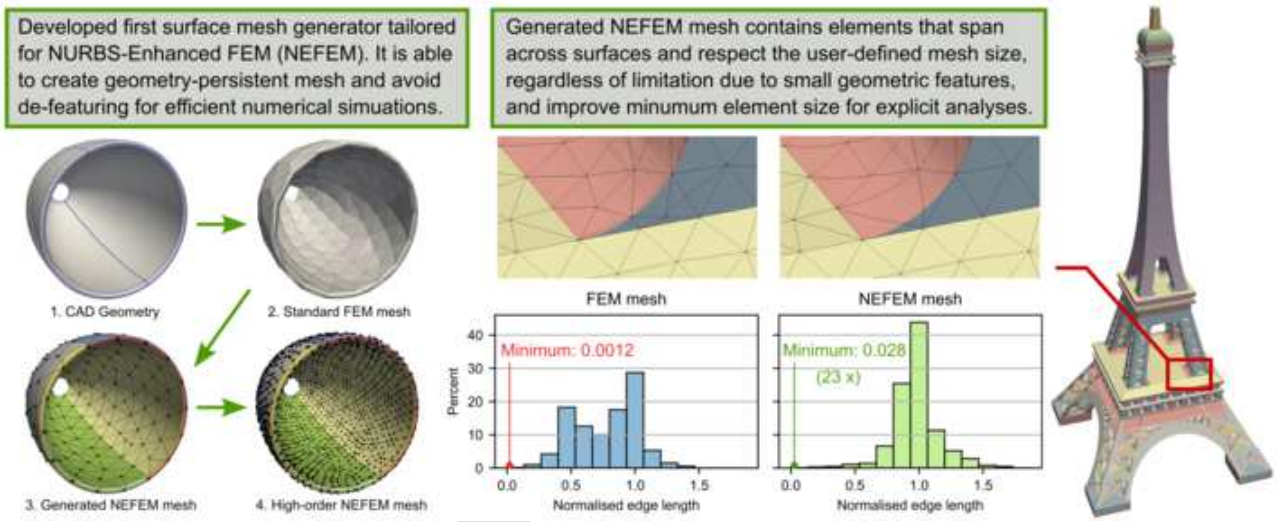
To appear in: *Computer-Aided Design*

Received date : 24 July 2022
Revised date : 6 September 2023
Accepted date : 16 November 2023

Please cite this article as: X. Zou, S.B. Lo, R. Sevilla et al., The Generation of 3D Surface Meshes for NURBS-Enhanced FEM. *Computer-Aided Design* (2023), doi: https://doi.org/10.1016/j.cad.2023.103653.

Graphical Abstract

Click here to access/download;Graphical Abstract;Graphical_Abstract.png



The generation of 3D surface meshes for NURBS–enhanced FEM

Xi Zou*, Sui Bun Lo, Ruben Sevilla, Oubay Hassan, Kenneth Morgan

Zienkiewicz Centre for Computational Engineering, Faculty of Science and Engineering, Swansea University, Swansea SA1 8EN, Wales, United Kingdom

Abstract

This work presents the first method for generating triangular surface meshes in three dimensions for the NURBS–enhanced finite element method. The generated meshes may contain triangular elements that span across multiple NURBS surfaces, whilst maintaining the exact representation of the CAD geometry. This strategy completely eliminates the need for de–featuring complex watertight CAD models and, at the same time, eliminates any uncertainty associated with the simplification of CAD models. In addition, the ability to create elements that span across multiple surfaces ensures that the generated meshes are highly compliant with the requirements of the user–specified spacing function, even if the CAD model contains very small geometric features. To demonstrate the capability, the proposed strategy is applied to a variety of CAD geometries, taken from areas such as solid/structural mechanics, fluid dynamics and wave propagation.

Keywords: NURBS–enhanced finite element method (NEFEM), mesh generation, de–featuring, exact geometry, high–order approximation

1. Introduction

The preparation of computer aided design (CAD) models for computational simulations remains one of the most time consuming parts of the whole simulation process. One aspect that requires a significant amount of human hours and expert decision making is the de–featuring of complex geometric models [1, 2]. CAD models often contain multi–scale geometric features that might, or might not, be relevant to a particular simulation. Using a standard mesh generation algorithm, with a CAD containing such features, usually leads to several issues, such as the generation of badly shaped elements and excessive and unnecessary local mesh refinement. Highly distorted elements can have an important impact in the quality of the simulations [3], whereas unnecessary mesh refinement can pose severe restrictions in the simulation of transient phenomena using explicit time marching algorithms. The later is of particular importance in a high–order setting, where coarse elements are preferred, to exploit the full advantage of high–order approximations. In this context, the presence of a few small elements can make a simulation unaffordable.

Although some semi–automatic tools for de–featuring CAD models exist [4], it is not easy to know if de–featuring a certain CAD model will induce significant changes in the engineering quantities of interest, introducing a level of uncertainty into the simulation. In addition, the de–featuring process cannot be fully automatised, as it is dependent on the physics to be simulated and even on specific parameters of a simulation. For instance, the level of de–featuring required in heat transfer, solid mechanics, electromagnetics or fluid mechanics simulations is completely different. This does not only mean that a different de–featuring is to be performed for the solution of each physical problem, but it poses a more profound issue. When different physics is to be considered, either the same geometric model is considered for all the physics, with a non–optimal de–featuring, or different geometric models will be utilised for different physics. This prevents, for instance, a multi–objective optimisation based on different physics. Even if a single physical problem

*Corresponding author

is considered, problem parameters, such as the frequency in a wave propagation problem or the Reynolds number in a fluid mechanics problem, usually induce different requirements in terms of de–featuring.

In addition to the uncertainty caused by de–featuring CAD models, the generation of meshes for computational simulations induces an extra level of geometric uncertainty. The CAD model, even if de–featured, is approximated by a surface discretisation, usually employing triangular or quadrilateral elements. The accuracy of the geometric approximation can be improved by using mesh refinement based on curvature, mesh sources or high–order elements. However, a factor often not given enough importance is that the resulting mesh only provides a piecewise $\mathcal{C}^0$ description of the original CAD model. When employing high–order elements, where coarse meshes are preferred, the discontinuous derivative of the normal between elements can induce non–physical effects, such as diffraction in wave propagation problems, concentration of stresses in a solid mechanics problem or entropy production in fluid mechanics problems. This effect can even drive a degree adaptive process towards an incorrect solution [5, 6, 7].

The NURBS–enhanced finite element method (NEFEM), originally proposed in [8], provides a simple and efficient approach to ensure that the geometry of the CAD model is exactly preserved during the simulation. The method was extended to three dimensional domains in [9] and it has been applied in a variety of problems involving heat transfer, electromagnetics, fluid mechanics and solid mechanics [10]. The NEFEM rationale also provides a powerful strategy for completely avoiding the de–featuring of complex models, while removing the requirement for small elements in regions where small geometric features are present. The main idea, similar to the virtual topology framework [11], is to consider elements that span across multiple surfaces. In contrast to the virtual topology, the elements can include non–smooth variations of the normal to the geometry and still maintain the exact representation of the geometry.

Despite these advantages, its applicability to problems involving complex geometries has been hampered by the lack of an automatic mesh generation algorithm. In fact, the lack of automatic mesh generators has led researchers to apply the NEFEM on unfitted meshes [12, 13, 14] or by using meshless methods [15, 16]. To date, a two dimensional NEFEM mesh generation approach has been available [17] and has demonstrated the benefit of using such elements. For a two dimensional electromagnetic scattering example, it was shown in [17] that the use of NEFEM can speed up a simulation by a factor of 140. This speed up is the result of using large elements, not restricted by the presence of small geometric features, making the use of time marching algorithms affordable.

This work presents the first three dimensional triangular surface mesh generation strategy for NEFEM. The proposed approach is capable of producing elements that span across different NURBS surfaces, maintaining the exact boundary representation and completely removing the need of de–featuring CAD models. By extending operations such as edge collapse and edge split, an initial finite element mesh is modified to offer better compliance with the user–defined spacing function. The concept of geometric supporting points, used to ensure the exact NURBS representation of elements spanning across multiple surfaces, is introduced and the strategy to compute these points is detailed. In the presence of trimmed NURBS surfaces, a validity check is performed to ensure that edges do not intersect trimming curves and, when these intersections are found, an edge curving strategy is proposed to alleviate the problem. Finally, two simple operations are employed to redefine badly shaped elements. The first is an extension of the traditional swap for edges that span across multiple surfaces. The second is completely novel and deals with the placement of the newly introduced geometric supporting points. The work also considers the construction of high–order nodal distributions on NEFEM triangular surface elements. Although the geometry of the elements is completely independent of the degree of the approximation used by the solver, this extension is a basic requirement to ensure that a NEFEM solver can utilise such meshes for an arbitrary order of approximation. The proposed approach is finally applied to generate meshes for a series of CAD models that contain multi–scale geometric features. The examples demonstrate the ability of the developed strategy to generate meshes that are valid, capture the exact geometry and comply with the user–defined spacing function, even in the presence of geometric features that are much smaller than the required spacing.

The outline of the paper is as follows. In Section 2, the definition of NEFEM entities is extended to account for the possibility of triangular elements spanning across multiple surfaces. Section 3 summarises the mesh requirements and presents the proposed technique to generate surface NEFEM meshes. In Section 4, the generation of high–order nodal distributions on NEFEM surface elements is detailed. Several mesh

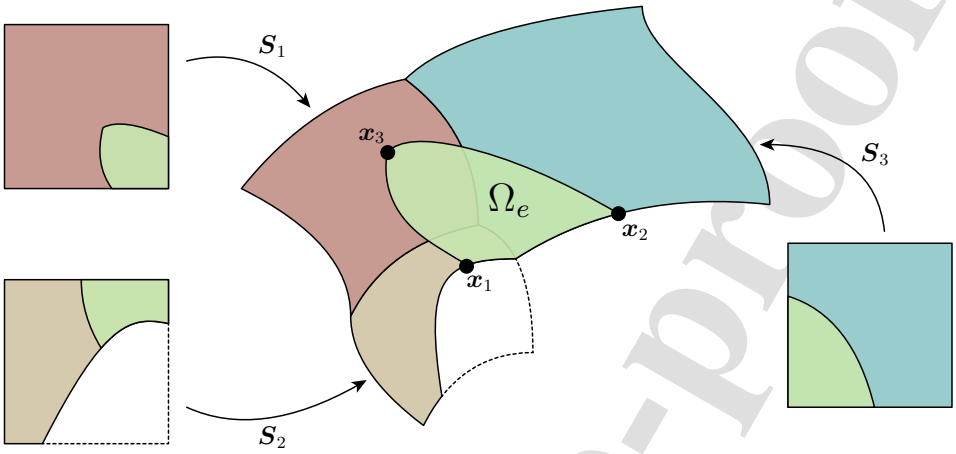


Figure 1: A NEFEM triangular surface element spanning across three different NURBS surfaces.

examples, of increasing complexity, are presented in Section 5, to illustrate the potential of the proposed technique. Finally, Section 6 summarises the main conclusions.

2. NEFEM geometric entities

In the standard finite element framework, the CAD model is only used at the mesh generation stage, to define the nodal distribution and the element connectivity. When the surface mesh has been created, finite element solvers employ an isoparametric formulation, in which the geometry of a surface element is defined in terms of a polynomial interpolation of the points provided by the mesh generator.

In the NEFEM approach, the exact CAD boundary representation is used to define the curved surface elements. This guarantees that the resulting elements introduce no geometric error and the geometry is made persistent throughout the whole simulation process [18].

A new definition of NEFEM surface triangular elements is proposed in this work, generalising the original definition [9, 19] which assumes that a surface element:

- belongs to a unique NURBS surface,
- is the image, through the NURBS surface parametrisation, of a straight sided triangle in the parametric space.

In this work, a NEFEM surface triangle is defined as a collection of trimmed NURBS surfaces. This, more general, definition, allows for a surface element to traverse several NURBS surfaces. In addition, the edges of the triangle in the parametric space will be allowed to be curved, introducing more flexibility to guarantee the validity of NEFEM triangles.

Figure 1 shows a general NEFEM triangular element, Ω_e, with vertices $\boldsymbol{x}_1$, $\boldsymbol{x}_2$ and $\boldsymbol{x}_3$. The triangle spans across three different NURBS surfaces, parametrised by $\boldsymbol{S}_1$, $\boldsymbol{S}_2$ and $\boldsymbol{S}_3$, with $\boldsymbol{S}_2$ parametrising a trimmed surface. The triangular element Ω_e is formally defined as the union of three different trimmed NURBS surfaces, referred to as *physical subdomains*. In general, the element is expressed as

$$\Omega_e = \bigcup_{j=1}^{\mathtt{n_{sdo}}} \Omega_{e,j}, \tag{1}$$

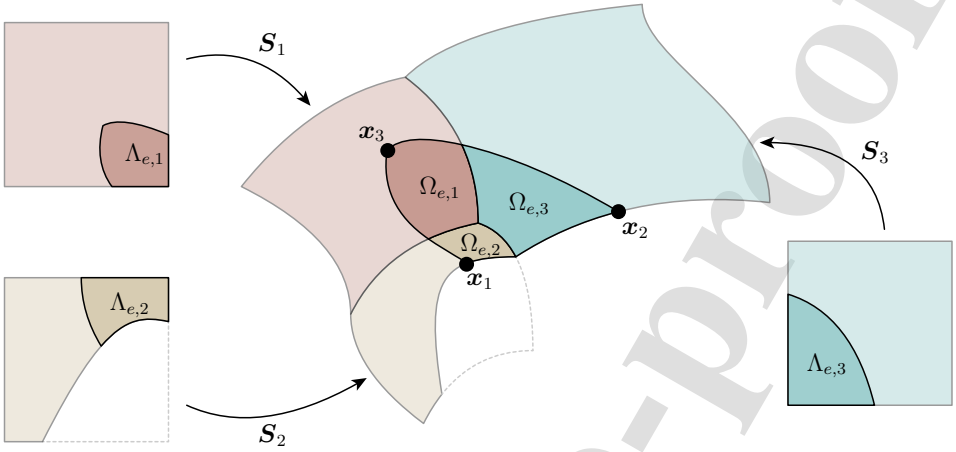


Figure 2: A NEFEM triangular surface element spanning across three different NURBS surfaces, showing the parametric and physical subdomains that form the element.

where $\mathtt{n_{sdo}}$ is the number of physical subdomains that form Ω_e. Each physical subdomain is defined as the image by the NURBS surface parametrisation of a *parametric subdomain*, namely

$$\Omega_{e,j} = \boldsymbol{S}_j(\Lambda_{e,j}), \quad \text{for } j = 1, \dots, \mathtt{n_{sdo}}. \tag{2}$$

The parametric and physical subdomains that form the element depicted in Figure 1 are shown in Figure 2. To simplify the data structure that will be utilised to store the NEFEM surface element information, parametric/physical subdomains are further divided into *parametric/physical subelements*, which are assumed to be triangular. The parametric and physical subelements that form the element depicted in Figure 1 are shown in Figure 3.

To complete the definition of a curved element, it is necessary to specify how the subelement edges are defined and the extra information required to allow elements spanning across several surfaces. These two aspects are detailed next.

2.1. Geometric definition of subelement edges

The edges of a parametric subelement, referred to as *parametric subedges* can be interior to the parametric space (i.e., with at most one vertex on a p–curve) or boundary edges (i.e., with both vertices on a p–curve). Parametric subedges with at most one vertex on a p–curve are defined, using a cubic isoparametric mapping, as

$$\begin{aligned} &\boldsymbol{\Psi} : [0,1] \to \Gamma_\lambda \\ &\xi \mapsto \boldsymbol{\Psi}(\xi) := \sum_{k=1}^{4} \boldsymbol{\lambda}_k N_k(\xi), \end{aligned} \tag{3}$$

where $\boldsymbol{\lambda}_k = (\lambda_k, \kappa_k)$ for $k = 1, \dots, 4$ are the four points in the parametric space of the NURBS that define the edge Γ_λ and N_i are the one dimensional Lagrange shape functions in the reference interval $[0, 1]$. Subedges with both vertices on a p–curve are simply defined, by trimming the p–curve, as

$$\begin{aligned} &\boldsymbol{\Psi} : [0,1] \to \Gamma_\lambda \\ &\xi \mapsto \boldsymbol{\Psi}(\xi) := \boldsymbol{C}_\lambda\big((1-\xi)\lambda_1 + \xi\lambda_2\big), \end{aligned} \tag{4}$$

where $\boldsymbol{C}_\lambda$ is the p–curve to which the subedge belongs and $[\lambda_1, \lambda_2]$ is the parametric interval for the trimming.

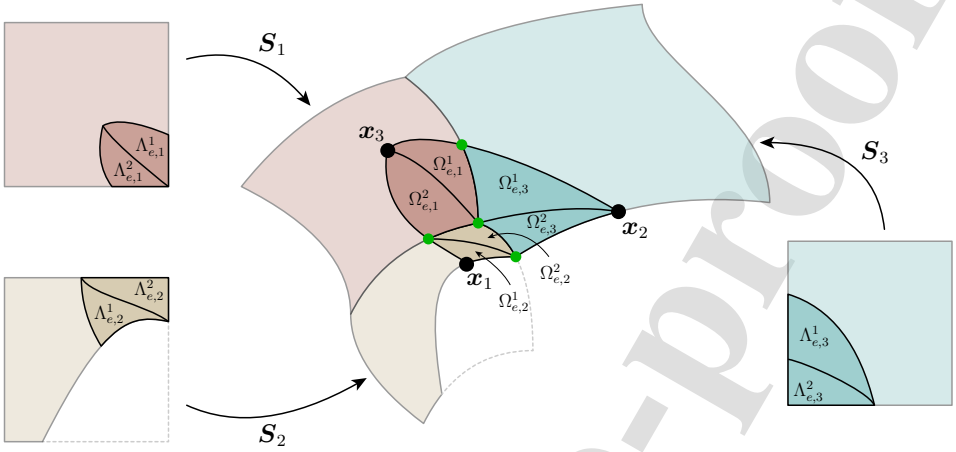


Figure 3: A NEFEM triangular surface element spanning three different NURBS surfaces, showing the parametric and physical triangular subelements that form the element.

2.2. Geometric definition of subelements

A parametric subelement $\Lambda^i_{e,j}$ is defined using the mapping

$$\begin{aligned} \boldsymbol{\Theta} \; &: [0,1]^2 \longrightarrow \Lambda^i_{e,j} \\ (\xi,\eta) \longmapsto \boldsymbol{\Theta}(\xi,\eta) := \; & (1-\eta)\boldsymbol{\Psi}_1(\xi) + \xi\boldsymbol{\Psi}_2(\eta) + (1-\xi)\boldsymbol{\Psi}_3(\eta) \\ & -(1-\xi)(1-\eta)\boldsymbol{x}_1 - \xi(1-\eta)\boldsymbol{x}_2, \end{aligned} \tag{5}$$

where $\boldsymbol{\Psi}_1$ is the parametrisation of the first subedge, connecting nodes 1 and 2 of the subelement, $\boldsymbol{\Psi}_2$ is the parametrisation of the second subedge, connecting nodes 2 and 3 of the subelement and $\boldsymbol{\Psi}_3$ is the parametrisation of the third subedge, connecting nodes 1 and 3 of the subelement. This mapping can be seen as a particular case of the blending function method [20]. Physical subelements are simply defined as $\Omega^i_{e,j} = \boldsymbol{S}(\Lambda^i_{e,j})$.

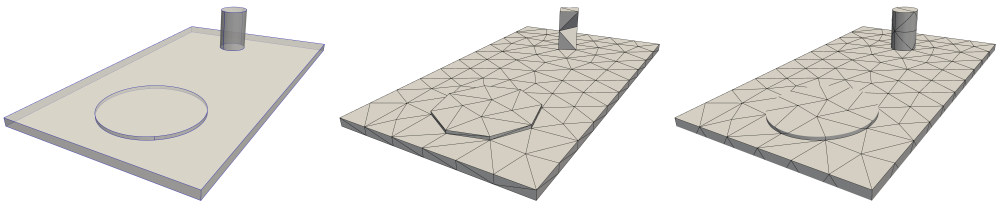
2.3. Geometric supporting points

In the original definition of NEFEM surface elements, an edge of a triangular element is fully described by specifying the two end nodes. This is not sufficient here because an edge can traverse physical p–curves, as shown in Figure 1. An enhanced edge description is proposed by introducing the concept of *geometric supporting points* (GS–points). The set of GS–points associated to one element Ω_e is given by the intersections of $\partial\Omega_e$ with the physical p–curves plus the intersections between physical p–curves that are inside Ω_e. The set of GS–points for the element depicted in Figure 1 are shown in Figure 3 as green dots.

It is worth emphasising that GS–points are only used to formally define a NEFEM surface element. They do not introduce new degrees of freedom in a solver that considers such meshes. Similarly, subelements are only introduced due to the piecewise nature of a NEFEM surface element, but the only element that is used in the solver is Ω_e.

3. Generation of NEFEM surface meshes

This section introduces the generation of the NEFEM triangular surface meshes suitable for low order approximations. It is worth emphasising that, in a NEFEM solver, the geometric approximation is completely decoupled from the solution approximation, i.e. no isoparametric concept is used. Therefore, NEFEM



(a) CAD model (b) Standard FEM surface mesh (c) Desired NEFEM surface mesh

Figure 4: Example illustrating the NEFEM surface mesh generation process.

enables the exact representation of complex geometries even when a low–order approximation of the solution is employed. Although the same geometry may be variously represented in different CAD models, modern CAD tools such as [21, 22] provide the functionality to simplify the data and generate standardised CAD files for the downstream mesh generators. In this paper, it is assumed that the upstream CAD geometry is watertight.

3.1. Mesh requirements

The requirements for a desired NEFEM surface mesh are:

1. The characteristic element size at a point, $h(\boldsymbol{x})$, must be dictated by the user and not be restricted by the presence of small geometric features, i.e. features with characteristic length $\ell \ll h$.
2. The surface discretisation must introduces zero geometric error. This means that all the points of an element are exactly located on the NURBS surfaces, not only the approximation points, as is usually the case with traditional mesh generators. This means that the whole surface element coincides with the NURBS and not only the element nodes are located on the surface, which is what happens in an isoparametric context.

The first requirement is the most challenging to fulfil and, at the same time, has the most important implications in terms of developing efficient solvers. The requirement ensures that no de–featuring of complex geometries is required, as small features will no longer induce undesired small elements. This will mean that the use of explicit time marching solvers for transient problems will be affordable, as the time step will not be massively restricted by the presence of a few undesired small elements. However, the requirement implies that a completely new mesh generation approach must be adopted, enabling elements to traverse through different NURBS surfaces.

The second requirement introduces the need for a completely new data structure to store the element information, but it also provides two important desired features. Firstly, it ensures that the solution error is free of any uncertainty induced by geometric errors, as is the case for standard FEM solvers. Secondly, it guarantees that the geometry is persistent throughout the whole simulation process, facilitating the implementation of degree adaptive approaches for high order methods [5, 6].

To illustrate the proposed mesh generation approach, an example is shown in Figure 4(a). The NURBS surfaces consists of a trimmed flat plate and two cylinders with significantly different radii and height. The thickness of the plate is much smaller than its length and width. One cylinder has a diameter much larger than its height, while, for the other cylinder the height is larger than the diameter. The desired spacing function, $h(\boldsymbol{x})$, is defined to be constant, with a value much larger than the thickness of the plate and the height of the flat cylinder. Figure 4(b) shows a triangular surface mesh generated using a standard mesh generator. It can be clearly observed that small and badly shaped triangles are present in regions where the desired element size is much larger than a geometric feature. The aim of this section is to explain the generation of the NEFEM surface mesh shown in Figure 4(c).

3.2. Initial mesh

Consider a surface manifold $\Omega \subset \mathbb{R}^3$ and a user–specified spacing function $h : \Omega \to \mathbb{R}$, defining the spacing at a given location $\boldsymbol{x} \in \Omega$. In this work, the spacing function is defined by using a background mesh and a set of point, line and triangular sources which implement the mesh control in Chapter 17 of [23]. The first stage of the proposed NEFEM mesh generation approach is to generate an initial FEM mesh, as shown in Figure 4(b), using a standard surface mesh generator. It is anticipated that the initial mesh will exhibit the following characteristics:

1. The mesh is watertight, as it discretises the original watertight CAD geometry.
2. The user specified spacing function $h(\boldsymbol{x})$ is generally respected, but, in regions where geometric features are smaller than the desired element size, the spacing function will not necessarily be respected.

Figure 4(b) shows an initial triangular mesh for the geometry of Figure 4(a). The mesh has been generated using a standard FEM mesh generator with uniform spacing.

3.3. Remeshing procedure

Starting from this initial mesh, a remeshing procedure is employed, to create the desired NEFEM elements, by using local mesh modification operations. The main idea is to loop over the nodes on physical p–curves, to identify the connected element edges with length smaller than the user–defined spacing. When these edges have been identified, they are collapsed recursively. This procedure creates the NEFEM elements. These may span over multiple surfaces to satisfy the requirement of the spacing function.

For the initial surface triangular mesh, $\mathcal{T}_h$, the set of edges is denoted by $\mathcal{E}_h$. Each edge is referenced in terms of the two nodes that it connects, e.g. $E_{a,b} \in \mathcal{E}_h$ denotes the edge connecting nodes $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$. To decide if an edge $E_{a,b}$ of the initial mesh is a valid NEFEM edge, its length, $|E_{a,b}|$, computed using the appropriate NURBS surface or p–curve parametrisation, is compared to the user–defined spacing function $h(\boldsymbol{x})$. If the length of the edge is such that

$$|E_{a,b}| \leq h/\sqrt{2}, \tag{6}$$

the edge is considered too short and, therefore, not compliant with the desired spacing. Similarly, if the length of the edge is such that

$$|E_{a,b}| \geq h\sqrt{2}, \tag{7}$$

the edge is considered too long and, again, not compliant with the desired spacing. Finally, if

$$h/\sqrt{2} \leq |E_{a,b}| \leq h\sqrt{2}, \tag{8}$$

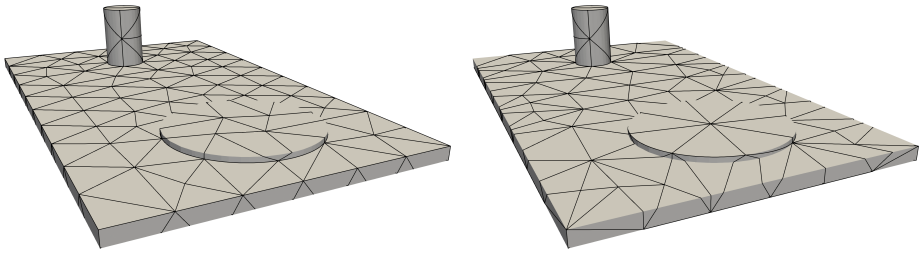
the edge is considered as compliant with the required spacing and it will be accepted as a NEFEM edge [24].

According to the criteria specified by (6)-(8), three disjoint sets of edges are constructed: the set of short edges, $\mathcal{E}_h^s$, the set of long edges, $\mathcal{E}_h^l$ and the set of compliant edges, $\mathcal{E}_h^c$, with $\mathcal{E}_h = \mathcal{E}_h^s \cup \mathcal{E}_h^l \cup \mathcal{E}_h^c$.

It is worth noting that non–compliant edges in the initial mesh could be both edges considered to be too short or too long. Edges to be considered too short are mainly due to the presence of geometric features that are much smaller than the required spacing. However, edges that are too long could be present in the initial mesh, as the initial finite element mesh does not consider any smoothing to strictly satisfy that no long edges are present. The reason for the factor $\sqrt{2}$ in (8) is to ensure that when an edge is regarded as being too long, i.e. satisfying (7), and it is split into two edges, the resulting edges will both be compliant, according to (8). It is important to note that the proposed algorithm targets the elimination of edges considered too short, but this work does not focus on the application of smoothing to strictly ensure that the all the elements satisfy (8).

The proposed remeshing approach is based on an extension of procedures usually found in standard mesh generators, such as edge collapse and edge split. The main idea is to identify edges connected to nodes on physical p–curves that can be collapsed to ensure that the spacing function is respected as much as possible. If collapsing an edge that was considered too short as per (6), results in the appearance of an edge that is considered too long as per (7), an edge split is applied. The strategy introduced here is novel, as both the edge collapse and split are devised to work with edges that traverse physical p–curves. This is not an operation that is available in standard mesh generators.

Remark 1. *By looping through the physical p–curves in the CAD model, the surface remeshing algorithm proposed in this section is applied sequentially to nodes on the curves. Alternating the numbering of the physical p–curves may result in different pattern as shown in Figure 5. All those patterns are valid NEFEM meshes.*



(a) A NEFEM mesh generated with intersection curves of the bottom surface numbered first.

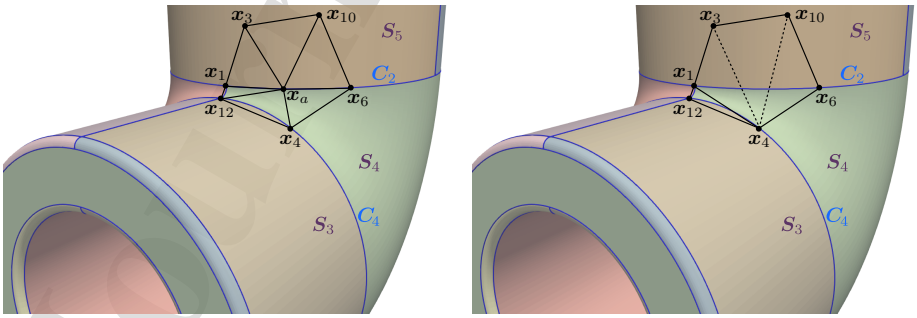
(b) NEFEM mesh generated with an alternative numbering of intersection curves.

Figure 5: NEFEM mesh patterns may change due to different numbering of physical p–curves.

3.3.1. Edge collapse

Consider an element patch $\mathcal{T}_{h,a} \subset \mathcal{T}_h$, with the centre node $\boldsymbol{x}_a$ being on a physical p–curve. The set of edges connected to $\boldsymbol{x}_a$ is denoted by $\mathcal{E}_{h,a}$. The edges on $\mathcal{E}_{h,a}$ are denoted by E_{ai}, for $i \in \mathcal{I}_a$, where $\mathcal{I}_a$ is the set of indices corresponding to nodes connected to $\boldsymbol{x}_a$. The number of edges in $\mathcal{I}_a$ is denoted by $\mathtt{n_E}$. The edges of the element patch $\mathcal{T}_{h,a}$ are classified into three subsets of short, long and compliant edges, viz. $\mathcal{E}^s_{h,a} = \mathcal{E}_{h,a} \cap \mathcal{E}^s_h$, $\mathcal{E}^l_{h,a} = \mathcal{E}_{h,a} \cap \mathcal{E}^l_h$ and $\mathcal{E}^c_{h,a} = \mathcal{E}_{h,a} \cap \mathcal{E}^c_h$. Figure 6(a) shows an example of an element patch centred at $\boldsymbol{x}_a$. The set of indices corresponding to nodes connected to $\boldsymbol{x}_a$ is $\mathcal{I}_a = \{1, 3, 4, 6, 10, 12\}$.

If $\mathcal{E}^s_{h,a} \neq \emptyset$, the edges in $\mathcal{E}^s_{h,a}$ are collapsed sequentially, until the set of short edges is empty. Each time an edge is collapsed, the sets $\mathcal{E}^s_h$, $\mathcal{E}^l_h$ and $\mathcal{E}^c_h$ are updated. The sets $\mathcal{E}^s_{h,a}$, $\mathcal{E}^l_{h,a}$ and $\mathcal{E}^c_{h,a}$ are updated accordingly. The proposed edge collapse implies deleting a point that was on a physical p–curve and creating edges that traverse physical p–curves. In the example of Figure 6(a), the edge $E_{a,4}$ is considered a short edge and is collapsed, as shown in Figure 6(b). As a result, the updated edges $E_{3,4}$ and $E_{10,4}$ traverse the physical p–curve parametrised by $\boldsymbol{C}_2$.



(a) Element patch before edge collapse

(b) Triangle strip after edge collapse

Figure 6: Illustration of the edge collapse strategy allowing new edges to traverse physical p–curves.

To describe an edge connecting two nodes on different surfaces, apart from the two end points, GS–points need to be created to account for the intersecting location of the edge and the physical p-curve. The proposed strategy to create the GS–points will be detailed in the next section. Nevertheless, the exact length of such edges cannot be computed before creating the GS–points. Therefore, an estimation is made to assess the length of new edges that will traverse physical p–curves before deciding if an edge needs to collapse. This is done using the previously computed lengths of existing edges in the element patch. This estimation is also used to decide if the edge collapse is performed by eliminating either the centre node $\boldsymbol{x}_a$ or the node connected to $\boldsymbol{x}_a$. Owning to this estimation, the resulting element edge length may slightly violate the imposed compliance criterion. In this process, preference is given to the collapse that maintains the centre node of the element patch, as this minimises the number of GS–points that needs to be created. However, it is common to encounter small edges where the two nodes belong to different physical p–curves and, therefore, the two possible edge collapse operations will induce the need of creating GS–points. In the example of Figure 6(a), the edge $E_{a,4}$ is made of two nodes on different physical p–curves, parametrised by $\boldsymbol{C}_2$ and $\boldsymbol{C}_4$. The edge collapse is made by deleting the central node $\boldsymbol{x}_a$, as shown in Figure 6(b). Before deciding to collapse the edge $E_{a,4}$, the length of the newly created edge $E_{10,4}$ is estimated as $|E_{10,4}| \approx |E_{10,a}| + |E_{a,4}|$.

3.3.2. Creation of GS–points

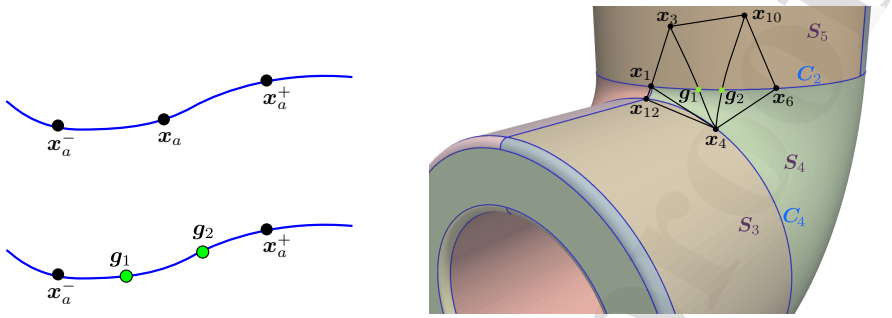
The creation of new edges that traverse physical p–curves requires the creation of GS–points, to ensure that such edges exactly lie on the surfaces given in the CAD model. The number of GS–points to be created, $\mathtt{n_{GS}}$, after performing the edge collapse is simply given by the number of edges that traverse a physical p–curve, i.e. edges where the two end nodes belong to two different NURBS surfaces. For each edge, the index of the physical p–curve that needs to be traversed is readily available, as it is the curve that contained the centre node of the element patch, $\boldsymbol{x}_a$. In fact, not only the physical p–curve is known, but also its parametric coordinate, i.e. λ_a and j such that $\boldsymbol{x}_a = \boldsymbol{C}_j(\lambda_a)$, are both known. In the example of Figure 6(b), edges $E_{3,4}$ and $E_{10,4}$ require the creation of one GS–point each and these GS–points will belong to the physical p–curve parametrised by $\boldsymbol{C}_2$.

The process begins by creating an ordered list, with the three nodes that belonged to the physical p–curve parametrised by $\boldsymbol{C}_j$ in the original element patch. The order is assigned by using the orientation of the physical p–curve and the ordered list of nodes is denoted by $\mathcal{I}_{a,j} = \{\boldsymbol{x}_a^-, \boldsymbol{x}_a, \boldsymbol{x}_a^+\} \subset \mathcal{I}_a$. The corresponding parametric coordinates of the three nodes are denoted by λ_a^-, λ_a and λ_a^+. To avoid self–intersection with edges from other element patches, the interval defining the GS–points is taken as

$$I_a := \left[\frac{\lambda_a^- + \lambda_a}{2}, \frac{\lambda_a + \lambda_a^+}{2}\right]. \tag{9}$$

A simple equally spaced distribution of nodes is initially placed in I_a, viz. $\lambda_{a,k}$ for $k = 1, \ldots, \mathtt{n_{GS}}$. The GS–points are obtained by mapping this distribution to the physical p–curve, i.e. $\boldsymbol{g}_k = \boldsymbol{C}_j(\lambda_{a,k})$ for $k = 1, \ldots, \mathtt{n_{GS}}$. Each GS–point is then assigned to an existing edge of the triangle strip, created by the edge collapse. This operation is easily performed after the triangles of the strip are ordered according to the orientation of the physical p–curve $\boldsymbol{C}_j$. Algorithm 1 lists the steps involved in the process of creating the GS–points. This approach can easily lead to badly shaped elements, but their validity is ensured. A repositioning strategy to guarantee better shaped elements will be described in Section 3.4. The process of creating the GS–points, for the example of Figure 6, is illustrated in Figure 7. Two GS–points are created in the physical p–curve and then associated to the edges $E_{3,4}$ and $E_{10,4}$. This allows for the NURBS–enhanced edges to traverse the physical p–curve parametrised by $\boldsymbol{C}_2$ and lie exactly on the NURBS surfaces $\boldsymbol{S}_5$ and $\boldsymbol{S}_4$.

Remark 2. *More complex scenarios, that involve an edge traversing several physical p–curves, are common, when a CAD model contains very small features, compared to the local requirements of the user–defined spacing function. Such cases require the creation of multiple GS–points for edges that traverse multiple physical p–curves. This situation is easily handled by creating the GS–points after a collapse operation is completed at an element patch level. This means that an edge that traverses multiple physical p–curves is*



(a) Creation of GS–points (b) Association of GS–points to edges

Figure 7: Illustration of the process to create the GS–points and to associate them to the edges that traverse a physical p–curve.

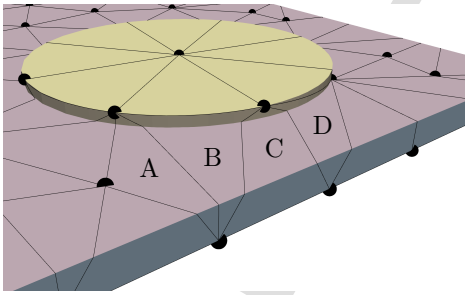


Figure 8: An example of NEFEM surface mesh including four NEFEM elements (labelled A, B, C and D) traversing three surfaces (coloured in brown, pink and blue, respectively), the geometry is based on Figure 4(a).

constructed recursively, by performing several edge collapse operations, one at a time, and creating a GS–point every time an edge collapse operation is performed. An example of such scenarios is shown in Figure 8, where four elements are traversing three surfaces after several edge collapse operations.

After the edge collapse strategy is performed sequentially, on all element patches with a node on a physical p–curve and with at least one short edge, according to (6), the set of short edges will be empty. However, as a result of the collapse, some newly created edges might become too long. A new edge split strategy is desired that is capable of handling edges that traverse multiple physical p–curves.

3.3.3. Edge split

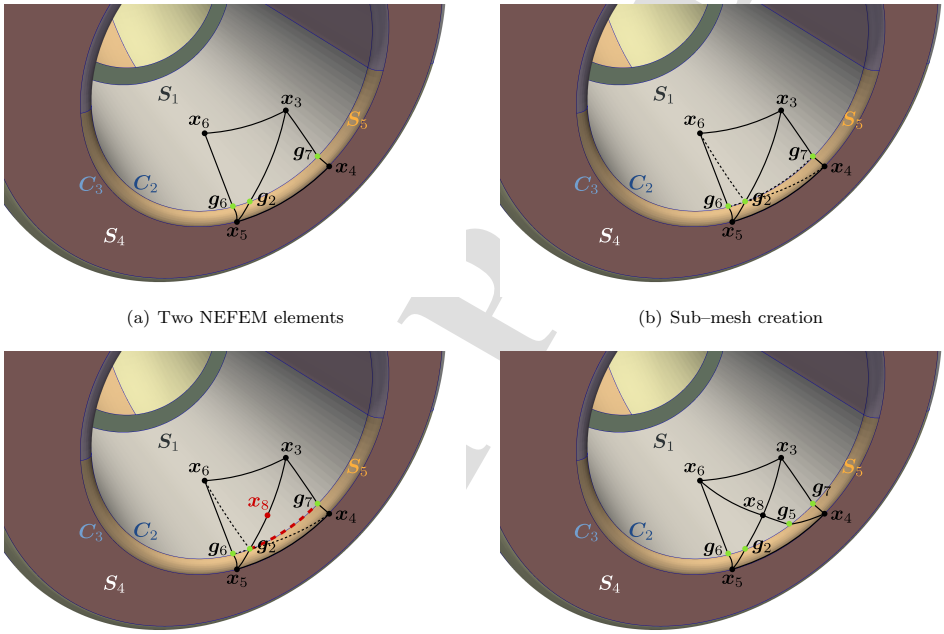
When a long edge is created by edge collapse, a new edge split process is performed to ensure compliance with the requirements of the user specified spacing function. The difficulty in splitting an edge of a NEFEM surface element lies in the fact that the newly created edges can traverse intersection curves. This situation is not encountered in the usual edge split utilised in standard mesh generators.

Consider two elements, Ω_1 and Ω_2 with nodes $\{\boldsymbol{x}_5, \boldsymbol{x}_3, \boldsymbol{x}_6\}$ and $\{\boldsymbol{x}_3, \boldsymbol{x}_5, \boldsymbol{x}_4\}$ respectively, that share the edge $E_{5,3}$, which traverses at least one physical p–curve and it is regarded as too long. To illustrate the process, the example shown in Figure 9(a) is considered. The two elements span across two different surfaces, parametrised by $\boldsymbol{S}_1$ and $\boldsymbol{S}_5$, and traversing a physical p–curve parametrised by $\boldsymbol{C}_2$.

Firstly, a triangular sub–mesh is created, with the advancing front method, by using the element nodes and the GS–points. The sub–edges of the sub–mesh are depicted with dashed lines in Figure 9(b). Each

Algorithm 1: Process for GS–point creation during edge collapse.

```
Collect x_a, x_a^- and x_a^+;
Collect the parameters λ_a, λ_a^- and λ_a^+;
Calculate the interval I_a according to (9);
Extract the number of GS–points to create n_GS;
Identify the edges {E_{a,k}} to add GS–points;
for k ← 1 to n_GS do
    Calculate λ_{a,k} ∈ I_a;
    Create GS–point g_k = C_j(λ_{a,k});
end for
Perform the edge collapse and update connectivity;
for k ← 1 to n_GS do
    Associate GS–point g_k to edge E_{a,k};
end for
```



(a) Two NEFEM elements

(b) Sub–mesh creation

(c) Mid–point creation

(d) Result of the edge split

Figure 9: Illustration of the edge split approach for an edge that traverses two physical p–curves.

sub–edge is associated to a geometric entity, which can be a physical p–curve, if both nodes belong to the same curve, or a surface, if at least one node does not belong to a physical p–curve. For instance, the sub–edge connecting $\boldsymbol{g}_2$ and $\boldsymbol{x}_6$ is associated to surface $\boldsymbol{S}_1$, whereas the edge connecting $\boldsymbol{g}_2$ and $\boldsymbol{g}_7$ is associated to the physical p–curve $\boldsymbol{C}_2$. Next, the mid point of the edge to be split, namely $\boldsymbol{x}_8$ on $E_{5,3}$, is computed using the appropriate NURBS surface parametrisation. The objective is to find if new GS–points need to be created when splitting the edge, i.e. when creating the two new edges connecting nodes $\boldsymbol{x}_6$ and $\boldsymbol{x}_8$ and nodes $\boldsymbol{x}_4$ and $\boldsymbol{x}_8$. To this end, sub–edges of the sub–mesh are marked if they will be traversed by the new edges to be created. These sub–edges, such as that connecting $\boldsymbol{g}_2$ and $\boldsymbol{g}_7$, highlighted in red in Figure 9(c), are

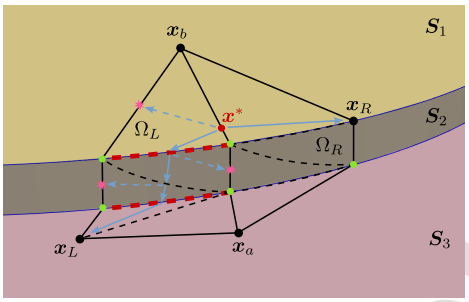
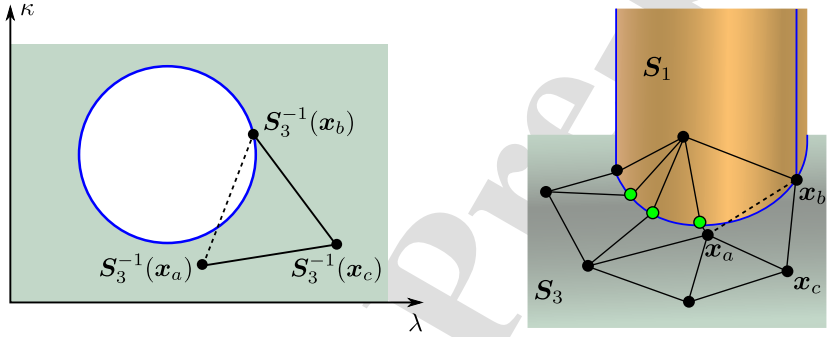


Figure 10: Illustration of the pathfinding results for edge split.



(a) Parametric space (b) Physical space

Figure 11: Illustration of the validity check for an edge that intersects a physical p–curve used to trim a NURBS surface.

identified using a combination of a modified Dijkstra's Algorithm [25] and an approximation of the geodesic described in [26]. If the marked edges to be traversed are associated to a physical p–curve, a new GS–point will be created along this edge, otherwise the new edge can be created by simply joining the two nodes. The result of the edge split is illustrated in Figure 9(d). A new GS–point $\boldsymbol{g}_5$ is created on the physical p–curve $\boldsymbol{C}_2$ and is associated to the new edge $E_{8,4}$.

The detailed steps in the process for edge splitting are listed in Algorithm 2. The Dijkstra–like pathfinding procedure is listed separately in Algorithm 3. A representative pathfinding scenario is also presented in Figure 10 to show the resulting paths. In this scenario, two elements Ω_L and Ω_R traverse three surfaces. Here, Ω_L is subdivided into 5 sub–elements and Ω_R into 4 sub–elements. The algorithm utilises two stacks, of sub–edges and sub–elements, to dynamically store the path in a recursive implementation. The solid arrows in Figure 10 indicate the path successfully found, while the dashed arrows represent the discarded testing paths when they hit a boundary sub–edge. The sub–edge rendered in red implies a GS–point will be created.

3.3.4. Validity check

This procedure for creating NEFEM surface elements might lead to non–valid elements in the presence of trimmed surfaces. A validity check and a mesh local modification is adopted to alleviate this problem. To illustrate the problem and the approach adopted, a representative scenario illustrated in Figure 11(b) is considered. The edge connecting nodes $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$, $E_{a,b}$, is associated to surface $\boldsymbol{S}_3$. However $\boldsymbol{S}_3$ is trimmed by the physical p–curve depicted in blue, which intersects the edge $E_{a,b}$, leading to a non–valid

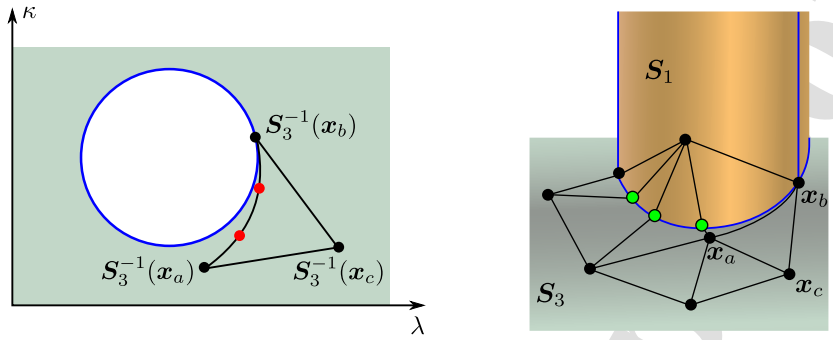
Algorithm 2: Process for splitting an edge traversing multiple surfaces.

Collect the vertices for the element pair Ω_L and Ω_R: $\boldsymbol{x}_a, \boldsymbol{x}_b, \boldsymbol{x}_L, \boldsymbol{x}_R$;
Collect the edge to be split $E_{a,b}$;
Collect the involved surfaces and curves: $\{\boldsymbol{S}_j\}$ and $\{\boldsymbol{C}_k\}$;
Build the sub–mesh $\mathcal{M}$ within elements Ω_L and Ω_R;
Extract from $\mathcal{M}$ the sets of boundary sub–edges for the two sides $\bar{\epsilon}_L$, $\bar{\epsilon}_R$;
Create the midpoint $\boldsymbol{x}^* = \texttt{BisectEdge}(E(\boldsymbol{x}_a, \boldsymbol{x}_b), \{\boldsymbol{S}_j\})$;
Find the sub–elements $\Lambda_L \in \Omega_L$ and $\Lambda_R \in \Omega_R$ such that $\boldsymbol{x}^* \in \partial\Lambda_L \cap \partial\Lambda_R$;
Initialise the sub–element stacks $\mathcal{L}_L = \{\Lambda_L\}$ and $\mathcal{L}_R = \{\Lambda_R\}$;
Find the sub–edges e^* such that $\boldsymbol{x}^* \in e^*$;
Initialise the sub–edge stacks $\epsilon_L = \{e^*\}$ and $\epsilon_R = \{e^*\}$;
Find sub–edges traversed by the new edge $E(\boldsymbol{x}^*, \boldsymbol{x}_L)$: $\epsilon_L = \texttt{FindPath}(\bar{\epsilon}_L, \epsilon_L, \mathcal{L}_L, \boldsymbol{x}^*, \boldsymbol{x}_L)$;
Find sub–edges traversed by the new edge $E(\boldsymbol{x}^*, \boldsymbol{x}_R)$: $\epsilon_R = \texttt{FindPath}(\bar{\epsilon}_R, \epsilon_R, \mathcal{L}_R, \boldsymbol{x}^*, \boldsymbol{x}_R)$;
Perform the split, update element connectivity;
for $e_l \in \epsilon_L$ **do**
  for $\boldsymbol{C}_k \in \{\boldsymbol{C}_k\}$ **do**
    if $e_l \in \boldsymbol{C}_k$ **then**
      Create a GS–point $\boldsymbol{g}_l \in e_l$;
      Associate $\boldsymbol{g}_l$ to $E(\boldsymbol{x}^*, \boldsymbol{x}_L)$;
    end if
  end for
end for
for $e_m \in \epsilon_R$ **do**
  for $\boldsymbol{C}_k \in \{\boldsymbol{C}_k\}$ **do**
    if $e_m \in \boldsymbol{C}_k$ **then**
      Create a GS–point $\boldsymbol{g}_m \in e_m$;
      Associate $\boldsymbol{g}_m$ to $E(\boldsymbol{x}^*, \boldsymbol{x}_R)$;
    end if
  end for
end for

Algorithm 3: Process for pathfinding during edge split: `FindPath`.

global: The sub–mesh $\mathcal{M}$
input: Boundary sub–edges $\bar{\epsilon}$, sub–edge stack ϵ_a, sub–element stack $\mathcal{L}_a$;
input: Midpoint $\boldsymbol{x}^*$ and goal vertex x_a;
Retrieve current sub–edge e_k from top of stack ϵ_a;
Retrieve current sub–element Λ_k from top of stack $\mathcal{L}_a$;
if $x_a \in \Lambda_k$ **then**
  return ϵ_a
else
  Identify the two sub–edges $e_i \in \partial\Lambda_k$ and $e_j \in \partial\Lambda_k$ other than e_k;
  Identify the corresponding neighbour sub–elements Λ_i and Λ_j;
  if $e_i \notin \bar{\epsilon}$ **then**
    Push e_i into stack ϵ_a;
    Push Λ_i into stack $\mathcal{L}_a$;
  else if $e_j \notin \bar{\epsilon}$ **then**
    Push e_j into stack ϵ_a;
    Push Λ_j into stack $\mathcal{L}_a$;
  else
    Add e_k into $\bar{\epsilon}$;
    Pop e_k out of stack ϵ_a;
    Pop Λ_k out of stack $\mathcal{L}_a$;
  end if
end if
Recursively update the sub–edge stack: $\epsilon_a = \texttt{FindPath}(\bar{\epsilon}_a, \epsilon_a, \mathcal{L}_a, \boldsymbol{x}^*, \boldsymbol{x}_a)$;

element. Figure 11(a) shows the problem in the parametric space of the surface parametrised by $\boldsymbol{S}_3$. For each edge that contains one node on a trimming intersection curve, e.g. $E_{a,b}$ in the example of Figure 11(b), a validity check is performed. The check simply involves computing the normal to the trimming curve in the



(a) Parametric space (b) Physical space

Figure 12: Illustration of the validity fix for an edge that is curved to avoid intersection with a physical p–curve used to trim a NURBS surface.

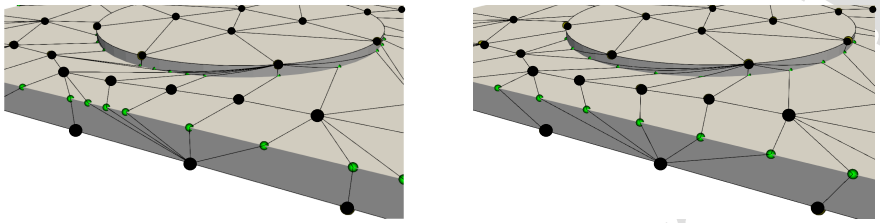
parametric space and a scalar product. More precisely, if the outward unit normal vector to the parametric space of $\boldsymbol{S}_3$ is denoted by $\boldsymbol{n}$ and the unit vector defined by the two end points of the edge in the parametric space is defined by $\boldsymbol{\lambda}_{a,b}$, the edge is considered valid if $\boldsymbol{n} \cdot \boldsymbol{\lambda}_{a,b} < 0$. If a non–valid edge is identified, a simple strategy is proposed, involving redefining the edge as a cubic curve in the parametric space. To illustrate the process, consider the cubic specified by the two end points, $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ in the example of Figure 11, and the tangent vectors at the two end points. The tangent vector at the node that belongs to the trimming curve, $\boldsymbol{x}_b$, is defined as the bisector of the angle formed by the tangent to the trimming curve and the vector $\boldsymbol{\lambda}_{a,c}$. The tangent vector to the cubic at the other end, $\boldsymbol{x}_a$, can be adjusted by the user. In the current implementation it is selected as the bisector of the angle formed by $\boldsymbol{\lambda}_{a,b}$ and $\boldsymbol{\lambda}_{a,c}$. When the cubic curve is defined in the parametric space, two additional points are used to store its geometric definition in the parametric space, as shown in Figure 12(a). In this way, the data structure only contains nodal coordinates and connectivities but not tangent vectors. The resulting physical curved edge is defined as the image of the cubic curve by the NURBS surface parametrisation, as depicted in Figure 12(b).

3.4. Mesh enhancements

Standard mesh generators employ a variety of local and global operations to enhance the quality of the generated elements. These operations include procedures such as edge swap and Laplacian smoothing. This work is not aimed at defining and improving quality measures for NEFEM surface elements, but at generating valid surface NEFEM meshes. As quality measures devised for traditional FEs are not suitable for NEFEM, due to the non–isoparametric nature of the NEFEM rationale, improved quality procedures will be the subject of future research. However, two local operations are introduced here to redefine badly shaped elements that are the result of the edge collapse, generation of GS–points and the edge split. The first operation is unique to the current mesh generation technique and involves sliding GS–points on physical p–curves. The second operation is the edge swap, which is extended here to consider elements that span across multiple NURBS surfaces.

3.4.1. Sliding of GS–points

The GS–points used to define the NURBS–enhanced edges can be moved along the corresponding physical p–curve to improve the shape of the NEFEM surface elements. When generated by mapping an equally–spaced nodal distribution in the parametric space of the p–curve, as described in Section 3.3.2, the resulting elements might be largely distorted, depending on the derivative of the NURBS curve parametrisation. An example of largely distorted elements that may be created by a naive construction of the GS–points is shown in Figure 13(a).



(a) Before sliding the GS–points (b) After sliding the GS–points

Figure 13: Detail of a NEFEM surface mesh corresponding to the geometry of Figure 4(a) before and after sliding the GS–points.

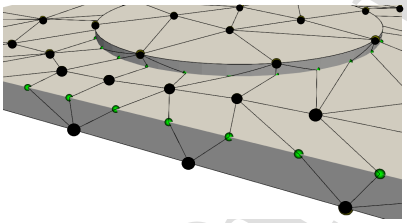


Figure 14: Detail of a NEFEM surface mesh after performing multiple edge swaps to the mesh of Figure 13(b).

The proposed approach to produce better shaped elements consists of sliding the GS–points to guarantee that they conform with the spacing function $h(\boldsymbol{x})$ along the physical p–curve. The iterative process for placing the nodes on the physical p–-curves proposed in [26] is utilised here. Figure 13(b) shows the result of sliding the GS–points of the mesh in Figure 13(a). In this example, the user–specified spacing function corresponds to a requirement for uniform spacing.

3.4.2. Edge swap

The strategy for performing an edge swap for an element that spans across multiple NURBS surfaces is closely related to the edge split process presented in Section 3.3.3. The only difference is that the edge to be swapped is deleted and no mid–node needs to be created. Using the general example in Figure 9, if an edge swap is to be performed by replacing edge $E_{5,3}$ by edge $E_{6,4}$, a sub–mesh is again used to identify the path that the new edge is to follow and to also identify if new GS–points need to be created as a result of the edge swap. After the path is identified, the new edge connection $E_{6,4}$ is created and, when needed, new GS–points are located and associated to this edge. The criteria used to decide if an edge swap is performed is, as usual, based on the angles of the triangle. However, in a NEFEM context, the angles of an enhanced triangle must be computed using the tangent to the enhanced edges by employing the NURBS description. In addition, the current implementation also computes the length of the proposed new edge before performing the swap. The edge is swapped only if the length of the resulting edge does not substantially differ from the user defined spacing. It is worth recalling that the length of an enhanced edge is evaluated using the NURBS entities and not just defined as the distance between the vertices.

4. Extension to high–order approximations

The strategy described in the previous section enables the generation of NEFEM surface meshes where the elements are allowed to span across multiple NURBS surfaces. This results in a better compliance with

the requirements of the user defined spacing function, even when very small geometric features are present in the CAD model. From the point of view of a NEFEM solver, the generated elements only support a linear approximation of the solution, as the only degrees of freedom of the triangle correspond to the three vertices. In this section, a novel strategy to generate high–order nodal distributions in NEFEM surface elements is presented.

4.1. Distribution of high–order edge nodes

The distribution of high–order nodes on NURBS–enhanced edges is similar to the strategy presented in [17] when generating two dimensional triangular NEFEM meshes. The main difference is that in the two dimensional case, NURBS–enhanced edges are always defined by NURBS curves, whereas in the current three dimensional setting, edges could be on NURBS curves and/or on NURBS surfaces. For a NURBS–enhanced generic edge, connecting nodes $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$, $E_{a,b}$, the set of $\mathtt{n_{se}}$ sub–edges that form $E_{a,b}$ is denoted by $\epsilon_{a,b}$. The sub–edges in $\epsilon_{a,b}$ are assumed ordered, such that the first sub–edge e_1 starts at $\boldsymbol{x}_a$ and the last sub–edge $e_{\mathtt{n_{se}}}$ ends at $\boldsymbol{x_b}$. Each sub–edge, connects a node of the original edge and a GS–point or two GS–points. The length of the sub–edges, $l_{r,s}$ is already available as it is computed during the mesh generation process. Therefore, the length $|E_{a,b}|$ of the edge $E_{a,b}$ is available.

Consider a p–th degree nodal distribution on the reference interval $I = [0, 1]$, namely $\{\xi\}_k \in I$ for $k = 1, \ldots, p+1$. The nodal distribution utilised is specified by the user, depending on the type of elements used by the NEFEM solver, e.g. an equally–spaced nodal distribution or a Fekete nodal distribution [27]. The high order nodes on the NURBS–enhanced edge $E_{a,b}$ are found in two stages. The sub–edge e_l that must contain the k–th high order node, for $k = 2, \ldots, p$, is easily identified by comparing the length of the sub–edges to the position of the k–th high order node in the reference interval, namely ξ_k. More precisely, the l–th sub–edge is such that

$$\frac{1}{|E_{a,b}|}\sum_{i=1}^{l-1}|e_i| \leq \xi_k \leq \frac{1}{|E_{a,b}|}\sum_{i=1}^{l}|e_i|. \tag{10}$$

Once the subedge is identified, the exact position of the high–order node is computed by iteratively solving a one–dimensional root finding problem. If the sub–edge is on a physical p–curve, parametrised by $\boldsymbol{C}_j$, the position of the k–th high order node is first computed in the parametric space of the p–curve as the root of the function

$$G(\eta) = \frac{1}{|E_{a,b}|}\left(\sum_{i=1}^{l-1}|e_i| + \int_{\lambda_l}^{\eta}\|\boldsymbol{C}_j'(\lambda)\|d\lambda\right), \tag{11}$$

where λ_l is the parametric coordinate of the first vertex of the subedge e_l. When the root $\eta^\star$ of G is obtained, the physical position of the high–order node is computed as $\boldsymbol{C}_j(\eta^\star)$. If the sub–edge that must contain the k–th node is not on an intersection curve but on a surface, parametrised by $\boldsymbol{S}_j$, the solution of a slightly different one dimensional root finding problem is required. This is due to the different definition of the sub–edge, as described in Section 2.1. The position of the node is obtained by computing the root of the function

$$G(\eta) = \frac{1}{|E_{a,b}|}\left(\sum_{i=1}^{l-1}|e_i| + \int_{0}^{\eta}\left\|\frac{d\boldsymbol{S}_j(\boldsymbol{\Psi}(\xi))}{d\xi}\right\|d\xi\right), \tag{12}$$

where $\boldsymbol{\Psi}$, given in (4), is the isoparametric mapping used to describe a cubic curve in the parametric space of $\boldsymbol{S}_j$. Once the root $\eta^\star$ of G is obtained, the physical position of the high–order node is computed as $\boldsymbol{S}_j\big(\boldsymbol{\Psi}(\eta^\star)\big)$. A simple bisection method is employed and the integrals appearing in (11) and (12) are evaluated using an adaptive Gauss–Legendre quadrature. The placement of high–order nodes does not need to be done sequentially, as the positions of the high–order nodes are independent.

To illustrate this approach to locating high–order nodes on NURBS–enhanced edges, Figure 15, shows the strategy followed when employing a third order Fekete nodal distribution on a NURBS–enhanced edge $E_{a,b}$. The edge contains a GS–point, $\boldsymbol{g}_1$, and is made of two sub–edges. Sub–edge e_1 contains the node $\boldsymbol{x}_a$ and the GS–point $\boldsymbol{g}_1$, while sub–edge e_2 contains the GS–point $\boldsymbol{g}_1$ and the node $\boldsymbol{x}_b$. The first high–order node placed on $E_{a,b}$ is identified to be placed on the first sub–edge, while the second high–order node must

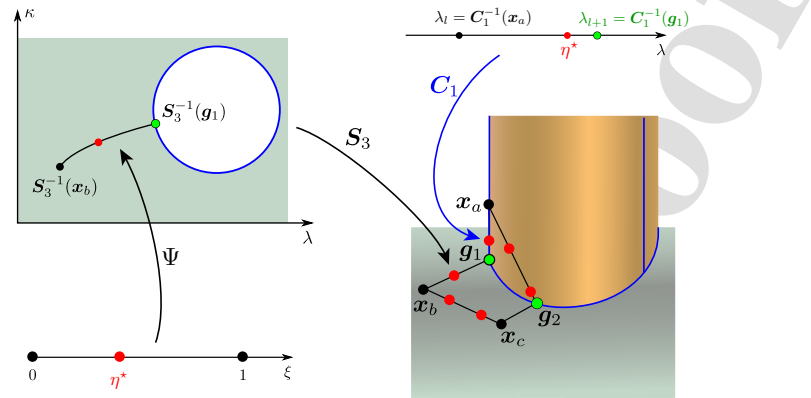
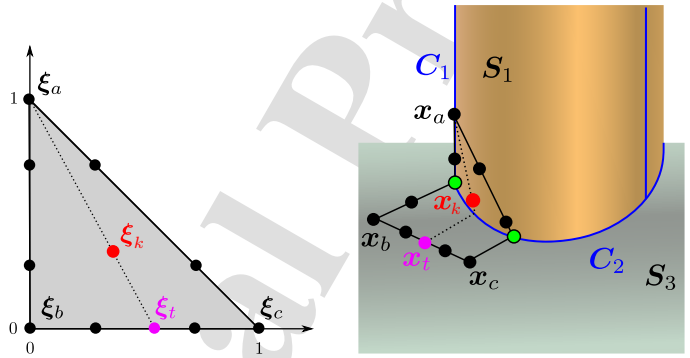


Figure 15: Illustration of the procedure to define a third order nodal distribution on an edge that is made of two sub–edges. The first sub–edge belongs to the physical p–curve parametrised by $\boldsymbol{C}_1$, whereas the second sub–edge is interior to NURBS surface parametrised by $\boldsymbol{S}_3$.



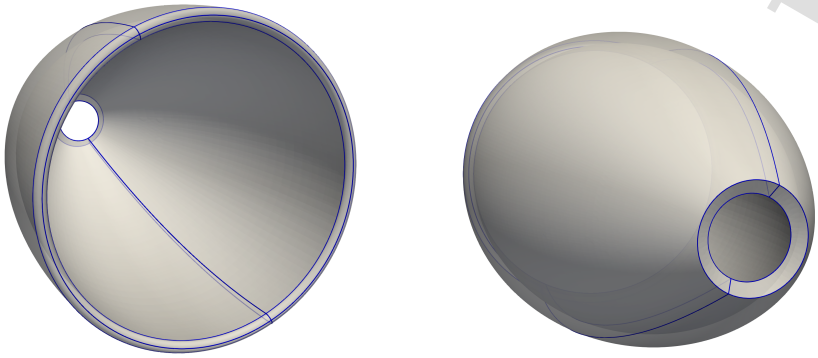
(a) Reference triangle (b) Physical NEFEM element

Figure 16: Illustration of the procedure used to place a high–order node that is interior to a NEFEM element. The example corresponds to a high–order approximation with degree $p = 3$.

be placed on the second sub–edge. The figure shows the parametric space of the physical p–curve that is used to solve the non–linear problem of (11). The figure also shows the parametric space of the NURBS surface and the mapping $\boldsymbol{\Psi}$ to the reference interval that is used to define a sub–edge on a NURBS surface.

4.2. Distribution of high–order interior nodes

For NEFEM elements with $p > 2$, the final step in the process consists of placing the high–order nodes that are interior to the element. For a desired degree of approximation p, an equally–spaced or Fekete nodal distribution is considered on a reference triangle. This is illustrated in Figure 16(a) for $p = 3$. For each interior node, the coordinates in the reference triangle are denoted by $\boldsymbol{\xi}_k = (\xi_k, \eta_k)$. The intersection between the line connecting the vertex of the reference triangle $(0, 1)$ with $\boldsymbol{\xi}_k$ and the horizontal axis $\eta = 0$, is given by $\boldsymbol{\xi}_t = (\xi_k/(1-\eta_k), 0)$. The point $\boldsymbol{x}_t$ is defined, over the physical edge $E_{b,c}$, such that the distance



(a) Leading edge view (b) Trailing edge view

Figure 17: CAD geometry of the hollow fairing for a turbine engine.

Table 1: Geometric data of the hollow fairing model.

Number of NURBS Surfaces	10
Number of NURBS Curves	22
Minimum curve length	15.62
Maximum curve length	850.74

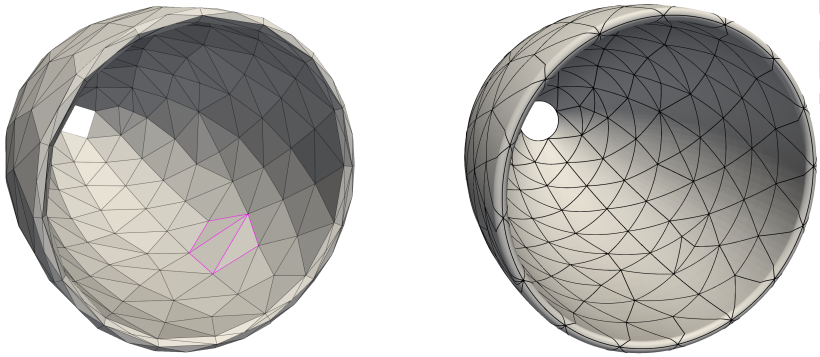
from $\boldsymbol{x}_b$ to $\boldsymbol{x}_t$, measured over $E_{b,c}$, is equal to $|E_{b,c}|\xi_k/(1-\eta_k)$. This is illustrated in Figure 16(b). The path between $\boldsymbol{x}_a$ and $\boldsymbol{x}_t$ in the physical space is then found by using an approximation of the geodesic [26]. This path is shown with a dashed line in the example of Figure 16(b). The final step consists of finding the position of the high–order node, $\boldsymbol{x}_k$, in the physical space. This is done by ensuring that the distance from $\boldsymbol{x}_a$ to $\boldsymbol{x}_k$, measured over the approximate geodesic that joins $\boldsymbol{x}_a$ and $\boldsymbol{x}_t$, is equal to $d_{a,t}\hat{d}_{a,t}$. Here $d_{a,t}$ denotes the arc length of the approximate geodesic joining $\boldsymbol{x}_a$ and $\boldsymbol{x}_t$ and $\hat{d}_{a,t}$ is the distance from $(0,1)$ to $\boldsymbol{\xi}_t$ in the reference space. The computation of the position of internal nodes utilises the algorithms, that have already described in detail, for building a NURBS–enhanced edge, performing an edge split and placing high–order nodes on element edges. The final position of the high–order node is illustrated in the example of Figure 16(b).

5. Examples

A number of examples have been included to illustrate the capability of the procedure for generating surface NEFEM meshes for geometries that contain small geometric features. The selected examples include a wide range of geometries relevant to different areas of computational engineering, such as solid/structural mechanics, fluid dynamics and wave propagation. In each example mesh, the edges adjacent to at least one intersection curve are particularly defined as the *edges of interest*. These edges of interest in a FEM mesh are analysed and modified during the generation of the corresponding NEFEM surface mesh.

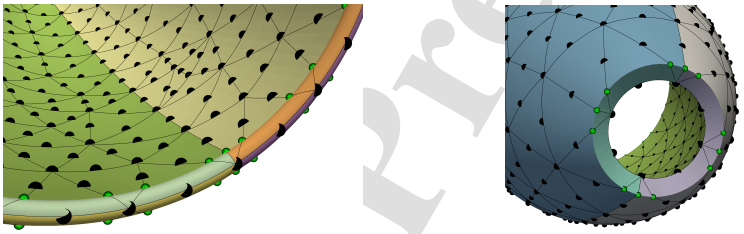
5.1. Hollow fairing for a turbine engine

The first example considers the generation of a NEFEM surface mesh for a turbine engine fairing with a uniform spacing function. The CAD model, shown in Figure 17, contains four large surfaces representing the outer and inner shells, and six narrow and thin surfaces representing the leading and trailing edges that connect these two shells. The representative dimensions of the model are listed in Table 1. Note that the lengths of the curves in the CAD model vary significantly, e.g. the length of the longest curve is



(a) Standard FEM mesh (b) NEFEM mesh

Figure 18: Meshes for the hollow fairing for a turbine engine. Elements highlighted in pink are split in the NEFEM mesh due to change of length metrics.



(a) Leading edge zoomed in (b) Trailing edge zoomed in

Figure 19: Detailed views of the generated high–order NEFEM mesh with $p = 2$ for the hollow fairing model. Surfaces are rendered in distinguishable colours. GS–points are rendered with green dots.

about 54 times larger than that of the shortest curve. More importantly, the user specified spacing in this example, $h(\boldsymbol{x}) = 100$, requires elements that have a representative length more than six times larger than the shortest curve in the CAD model. For this reason, the initial FEM mesh contains a number of small elements that clearly violate the desired spacing, as depicted in Figure 18(a). The condition number of the global finite element matrices will be adversely affected by the presence of these small elements. The presence of a single element with a very short edge length will impose severe restrictions on an explicit time marching algorithm, if numerical stability is to be achieved. The generated NEFEM surface mesh is shown in Figure 18(b). Details of the generated NEFEM mesh near the trailing and leading edges, where small surfaces are present, are shown in Figure 19(a) and (b), respectively. In this model, all surfaces are curved and, therefore, all surface elements are considered NEFEM elements to ensure that the exact representation of the domain is maintained. The elements highlighted with pink edges in Figure 18(a) have been split in the NEFEM mesh. This is because the the edge length of NEFEM elements is evaluated with the arc length of the approximated geodesic, which is typically longer than the Cartesian distance, thus they would trigger the edge split process as per (7). The statistics for both meshes are listed in Table 2. The generated NEFEM mesh has a slightly lower number of nodes and elements, as a result of the edge collapse performed on the original FEM mesh. Considering the edges of interest, the most significant difference is that the minimum edge length in the NEFEM mesh is more than nine times larger than the minimum edge length in the original FEM mesh and more than four times larger than the smallest geometric feature present in the

Table 2: Surface mesh statistics for the hollow fairing model.

Surface mesh $p = 1$	FEM	NEFEM
Number of nodes	345	278
Number of elements	690	556
Number of edges of interest	483	386
Minimum normalised edge length	0.0761	0.7340
Average normalised edge length	0.7617	1.1230

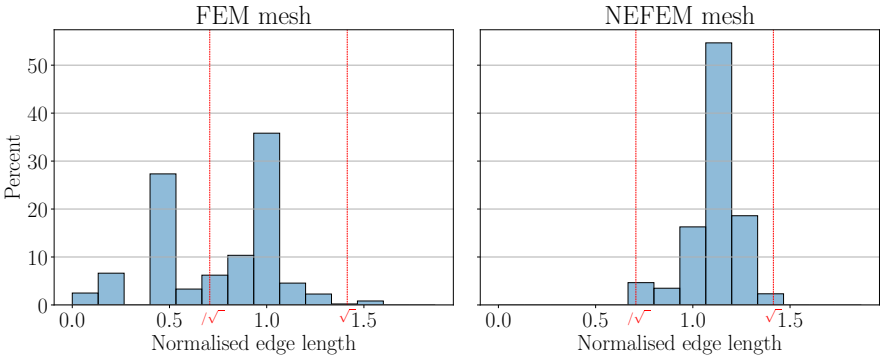
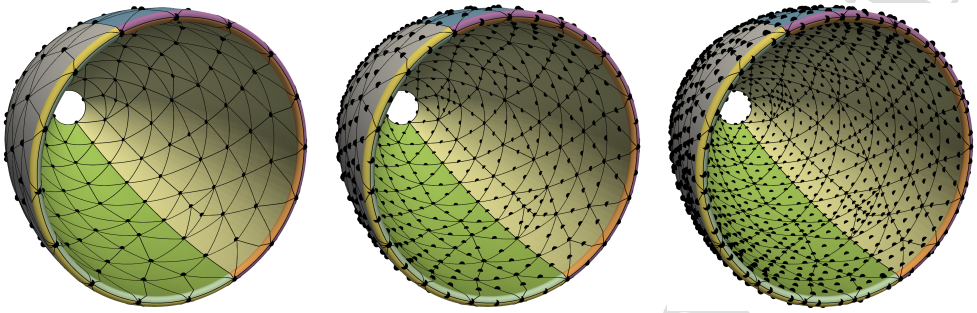


Figure 20: Histograms of edges of interest in FEM and NEFEM meshes for the hollow fairing.

CAD model. This example shows the ability to produce surface meshes with triangular elements spanning across multiple NURBS surfaces, while retaining the exact geometric representation. To further illustrate the potential of the approach, histograms of the normalised edge length for both the original FEM mesh and the resulting NEFEM mesh are displayed in Figure 20. A comparison clearly shows the ability of the method to create a mesh in which the majority of elements have an edge length very close to the requirements of the user–specified spacing function, even in the presence of small geometric features. To conclude this example, the technique described in Section 4 is applied to construct high order nodal distributions on the surface NEFEM elements. Figure 21 shows the surface NEFEM meshes obtained for linear, quadratic and cubic approximation. Note that, in all cases, the same, exact representation of the geometry is guaranteed. The only use of the high–order nodal distributions is to define an approximation of the solution in a NEFEM solver.

It is worth noting that, in this example, the virtual topology paradigm [11], would also allow meshing across different surfaces, as there is a smooth transition of the normal between the surfaces. However, as has been shown in [7], utilising such meshes in a traditional finite element context will lead to a non–exact approximation of the geometry. Further, as isoparametric elements only provide a piecewise $\mathcal{C}^0$ approximation of the geometry, even if high–order approximations are used, the resulting simulations might show non–physical singularities in the solution induced by the piecewise $\mathcal{C}^0$ geometric approximation. This is relevant in different applications. In stress analysis, boundaries with $\mathcal{C}^0$ continuity might lead to a stress concentration. In fluid mechanics, corners are known to introduce non–physical entropy. In electromagnetics, corners can lead to strong singularities of the electromagnetic field. Therefore, the persistence of the true CAD model in the solver, via the NEFEM approach, is expected to bring several advantages, not only in terms of efficiency, but also in terms of reliability of the results.

To further illustrate this issue, Figure 22 shows the effect of using elements that traverse surfaces without a smooth transition of the normal in a FE context. As it can be observed, the approximation of the geometry



(a) $p = 1$ (b) $p = 2$ (c) $p = 3$

Figure 21: NEFEM surface mesh of the hollow fairing with linear, quadratic and cubic nodal distributions.

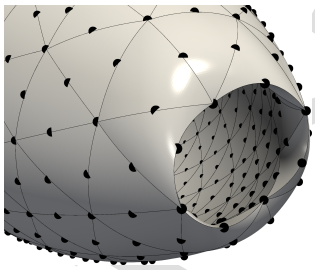


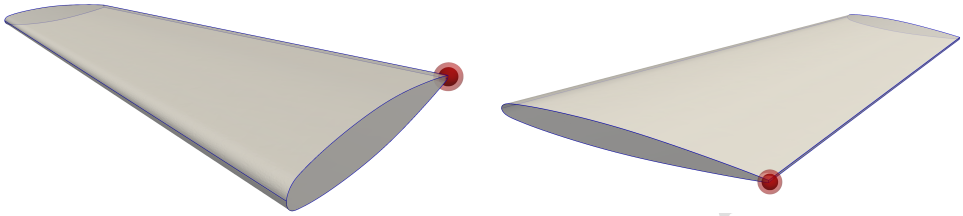
Figure 22: Virtual topology enabled FEM surface mesh of the hollow fairing with quadratic isoparametric elements, viewing from trailing edge. After generating elements that traverse multiple surfaces with no smooth transition of the normal, the geometry is approximated using quadratic polynomials, losing the exact geometric definition and leading to different physics.

with a high–order element would not reproduce the exact geometric features. In addition, the blunt trailing edge will not be captured and this is known to lead to significant differences in the physics that can be reproduced [28].

5.2. Wing with a blunt trailing edge

The next example considers a wing with a blunt trailing edge and is intended to show an ability to generate a NEFEM surface mesh with a prescribed non–uniform spacing function $h(\boldsymbol{x})$. The example also demonstrates the ability of the approach to generate elements that span across multiple surfaces, even when there is a non–smooth transition of the normal across the surfaces. This is a feature that cannot be achieved with the virtual topology approach.

Figure 23 shows the CAD geometry of the wing. The model consists of five NURBS surfaces, viz. the top and bottom surfaces, the tip and root of the wing and the blunt trailing edge. Compared to the previous example, the ratio between the maximum and minimum curve lengths of the CAD model is even more extreme and is almost 190 in this example. Representative dimensions of the model are listed in Table 3. The non–uniform spacing function is defined using a combination of point and line sources [23]. A point source, shown in Figure 23, is introduced near the trailing edge of the root and two line sources are introduced near the leading and trailing edges of the wing. The initial FEM and the generated NEFEM meshes are shown in Figure 24 and Figure 25, respectively. The local refinement induced by the point sources can be clearly observed in the rear view of both the FEM and NEFEM meshes. However, the element size on the NEFEM mesh grows rapidly, as the influence of the point source disappears, while the FEM mesh is refined

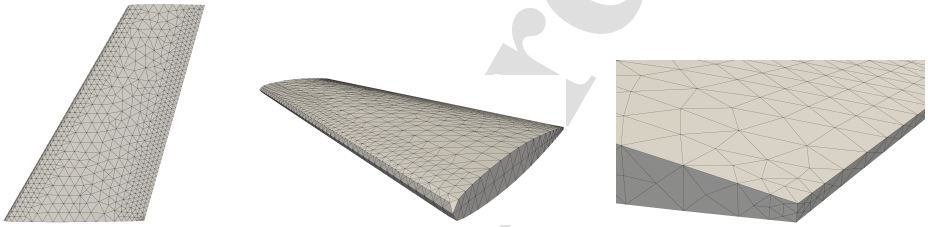


(a) Leading edge view (b) Trailing edge view

Figure 23: CAD model of the wing with a blunt trailing edge, a point source is prescribed and illustrated in red spheres.

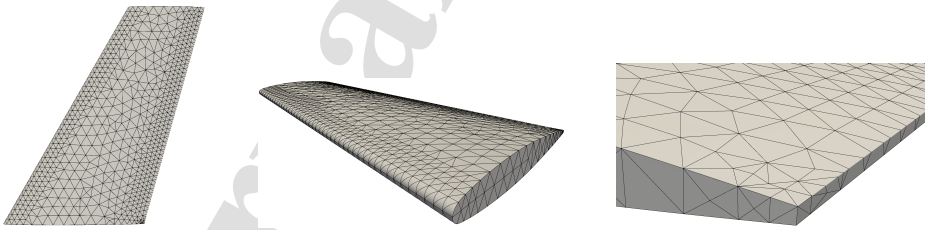
Table 3: Geometric data of the wing model.

Number of NURBS Surfaces	5
Number of NURBS Curves	9
Minimum curve length	7.27
Maximum curve length	1,381.12



(a) Top view (b) Leading edge view (c) Trailing edge view

Figure 24: FEM mesh of the wing model with views from different aspects.

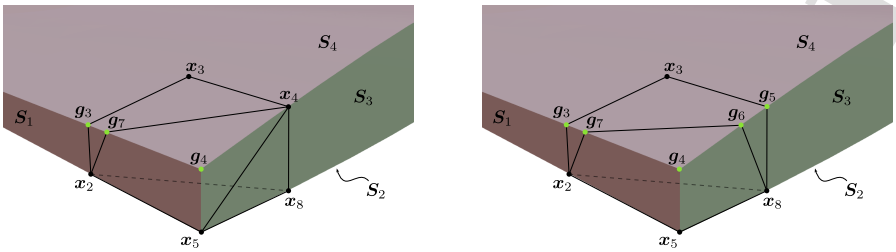


(a) Top view (b) Leading edge view (c) Trailing edge view

Figure 25: NEFEM mesh of the wing model with views from different aspects.

due to the small thickness of the blunt trailing edge. When the defined spacing is larger than the thickness of the blunt trailing edge, the generated NEFEM elements span across multiple surfaces, even when there is a non–smooth transition of the normal between the surfaces.

Remark 3. *The scenario in Figure 26(a) presents a state before collapsing edge* $E_{4,8}$ *from* $\boldsymbol{x}_4$ *to* $\boldsymbol{x}_8$*. This would typically be prevented by standard mesh generators, due to the creation of two triangular elements sharing all three nodes, viz.* $\{\boldsymbol{x}_2, \boldsymbol{x}_5, \boldsymbol{x}_8\}$*, and the geometric information that surfaces* $\{\boldsymbol{S}_1, \boldsymbol{S}_3, \boldsymbol{S}_4\}$ *intersect*



(a) Scenario before collapsing edge $E_{4,8}$ (b) Result after collapse

Figure 26: A detailed view of the NEFEM mesh at a corner of the blunt trailing edge.

Table 4: Surface mesh statistics for the wing model.

Surface mesh $p = 1$	FEM	NEFEM
Number of nodes	1,606	1,273
Number of elements	3,208	2,542
Number of edges of interest	1,596	1,357
Minimum normalised edge length	0.1455	0.7214
Average normalised edge length	0.7614	1.2810

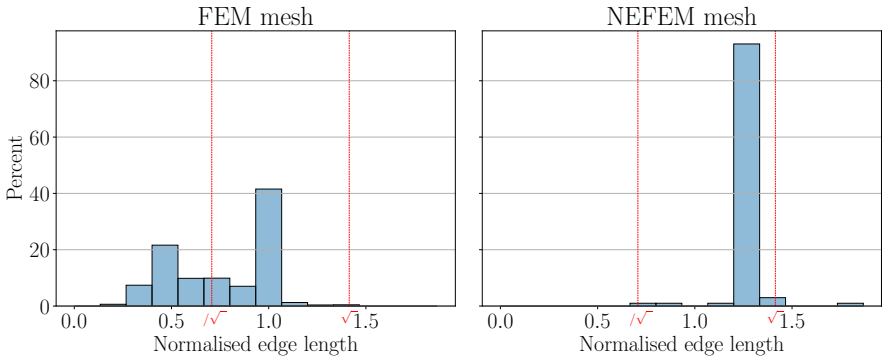
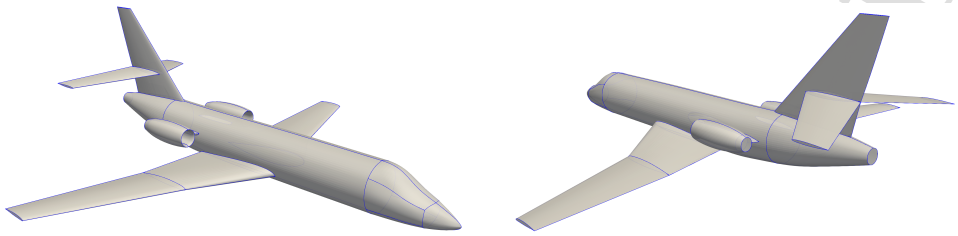


Figure 27: Histograms of edges of interest in FEM and NEFEM meshes for the wing.

at the location of $\boldsymbol{g}_4$ would be discarded. In NEFEM, this collapse is permitted due to the introduction of the GS–points. As shown in Figure 26(b), the geometric information is preserved by $\{\boldsymbol{g}_4, \boldsymbol{g}_6, \boldsymbol{g}_7\}$ in the upper element after the collapse.

The mesh statistics are listed in Table 4, where the element edge length is normalised with respect to the requirements of the user–specified spacing function. The NEFEM mesh again involves a lower number of nodes and elements, but the most significant difference is that, considering the edges of interest only, the minimum edge length in the NEFEM mesh is almost five times larger than the smallest edge in the FEM mesh. Figure 27 shows the histograms of the normalised edge length of interest for both the original FEM mesh and the resulting NEFEM mesh. The histogram of the NEFEM mesh shows that the majority of the



(a) Front view (b) Rear view

Figure 28: CAD model for the falcon aircraft model.

Table 5: Geometric data of the full aircraft model.

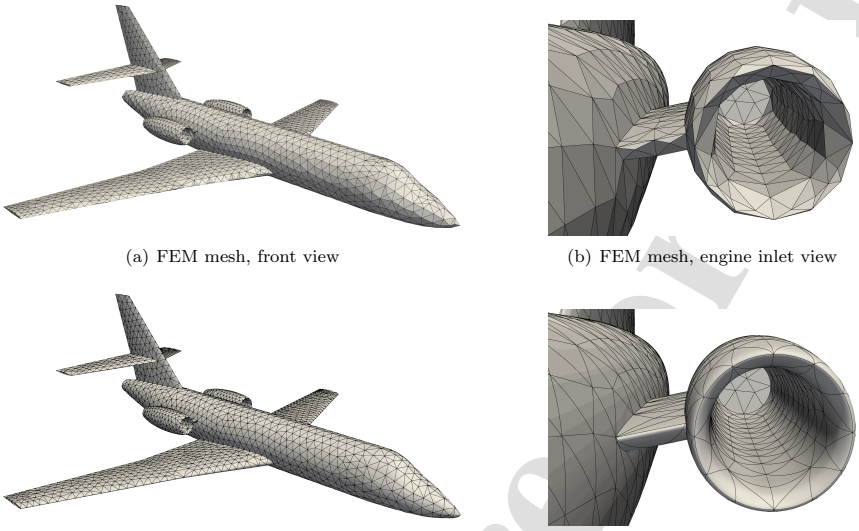
Number of NURBS Surfaces	48
Number of NURBS Curves	100
Minimum curve length	0.37
Maximum curve length	10.61

elements comply with the requirements of the spacing function.

It is worth noting that in computational fluid dynamics simulations, preserving the blunt trailing edge of a wing is crucial to reproduce the correct physics. Numerical simulations and experiments [28] have confirmed that collapsing the blunt trailing edge might lead to a steady flow, whereas preserving the blunt trailing edge triggers an unsteady behaviour that induces vibrations and noise. The main problem with preserving blunt trailing edges is the small thickness (2 mm to 3 mm), which naturally induces very small elements when using traditional methods. In turn, this leads to restrictions to the time stepping when using explicit time marching. This is particularly problematic when using high order methods where the objective is to use very large elements with high order approximations. The objective here is to preserve the geometric feature exactly and rely on high order approximations to capture the flow physics.

5.3. Complete aircraft

This next example involves a full aircraft model and it is designed to show the ability of the NEFEM surface mesh generator to handle complex geometries. The CAD model, shown in Figure 28, contains 48 NURBS surfaces and 100 NURBS curves. The geometric data is presented in Table 3. The model contains a variety of features, including very short curves and small surfaces, smooth transitions between different surfaces and sharp transitions with a non–smooth normal between surfaces. The minimum curve length is only 0.37 while the maximum is 10.61, which is nearly 30 times larger. The specified global mesh size is 0.35. The characteristic thickness of the wing is about 0.2, which poses a major limitation on the element size at the wing tip when using standard mesh generators. In this case, the minimum curve length is larger than the specific spacing, but compliance with the desired spacing is limited by the thickness of some NURBS surfaces, rather than a NURBS curve. The initial FEM and the resulting NEFEM meshes are shown in Figure 29. Two detailed views, near a wing tip and the tip of the tail, of the generated NEFEM mesh are shown in Figure 30. The figure clearly demonstrates the ability of the method to comply with the spacing function by creating elements that span across multiple surfaces. The mesh statistics are listed in Table 6. In the standard FEM mesh, when only the edges of interest are considered, the minimum edge length could only reach 8.3% of the desired mesh size, owing to the presence of small geometric features. This is significantly increased to 74.6% in the NEFEM mesh. The histograms, presented in Figure 31, clearly demonstrate the potential of the approach to guarantee a better compliance with the requirements of the user–specified spacing function. It is worth noting that edges of interest in the NEFEM mesh do not strictly follow the desired spacing, even an edge longer than the desired spacing range has been created in



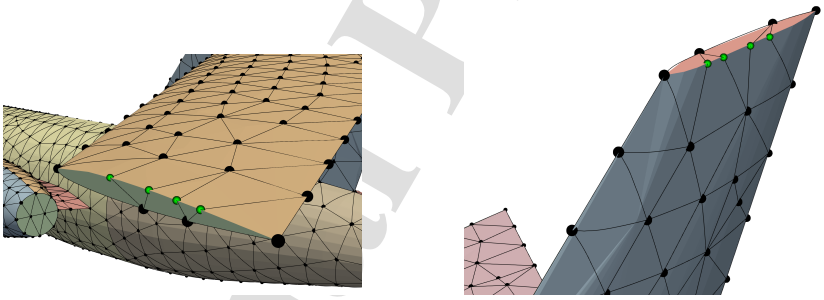
(a) FEM mesh, front view

(b) FEM mesh, engine inlet view

(c) NEFEM mesh, front view

(d) NEFEM mesh, engine inlet view

Figure 29: Meshes for the full aircraft model.



(a) Local view near the horizontal tail tip

(b) Local view near the vertical tail tip

Figure 30: Detailed views of NEFEM mesh for the falcon aircraft model.

Table 6: Surface mesh statistics for the full aircraft model.

Surface mesh $p = 1$	FEM	NEFEM
Number of nodes	3,464	3,393
Number of elements	6,924	6,782
Number of edges of interest	3,999	3,908
Minimum normalised edge length	0.0833	0.7461
Average normalised edge length	0.9423	1.0820

the NEFEM mesh. This can be easily corrected by applying mesh improving techniques such as edge flip and mesh smoothing.

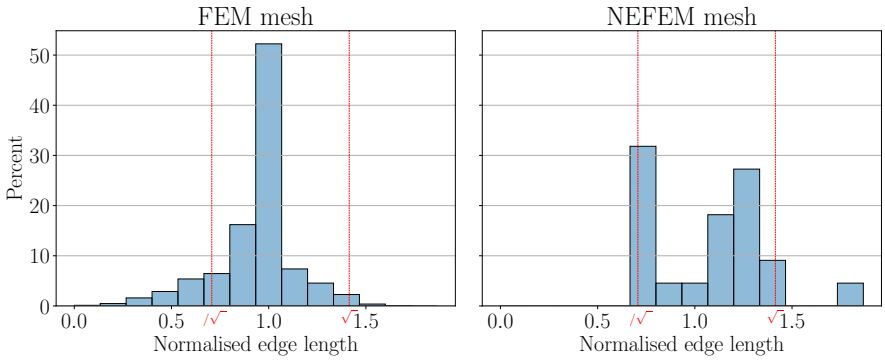


Figure 31: Histograms of edges of interest in FEM and NEFEM meshes for the full aircraft model.

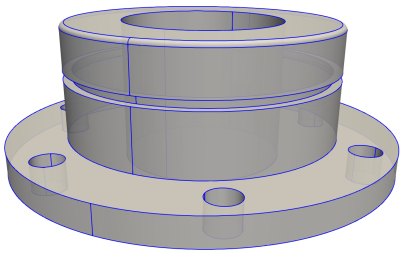


Figure 32: CAD geometry of a flange model.

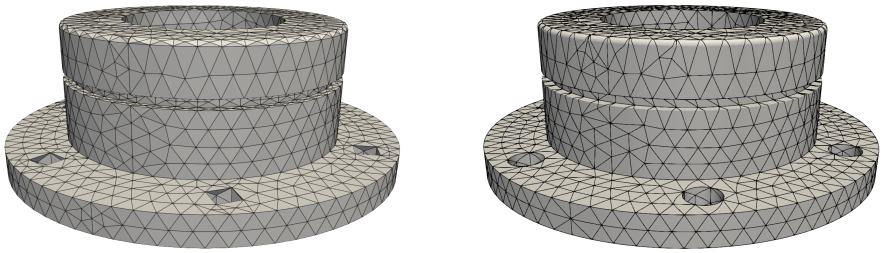
Table 7: Geometric data of the flange model.

Number of NURBS Surfaces	29
Number of NURBS Curves	70
Minimum curve length	1.57
Maximum curve length	138.13

5.4. Flange

The next example considers a mechanical component with multiple geometric features. This example aims to demonstrate the ability to handle a model with multiple trimmed surfaces by curving NEFEM edges as described in Section 3.3.4.

The CAD model, shown in Figure 32, depicts a flange, with a large centre hole and a skirt, containing six satellite holes for fastener installation. A U–shaped channel, at the mid–height of the body, creates three sliver surfaces in the shape of a ring or a cylinder. The round fillet, at the outer edge of the top surface, also introduces a curved sliver surface. The geometric data is detailed in Table 7. The geometric characteristics of the model results in the creation of a large number of small elements, with large aspect ratio, when using a standard FEM mesh generator, as presented in Figure 33(a). In this example, the minimum edge length in the initial FEM mesh is 19% of the desired element size. A view of the NEFEM mesh is shown in Figure 33(b). Again, a reduction in the number of nodes and elements is obtained, as detailed in Table 8, and, more significantly, the minimum edge length is more than double that for the

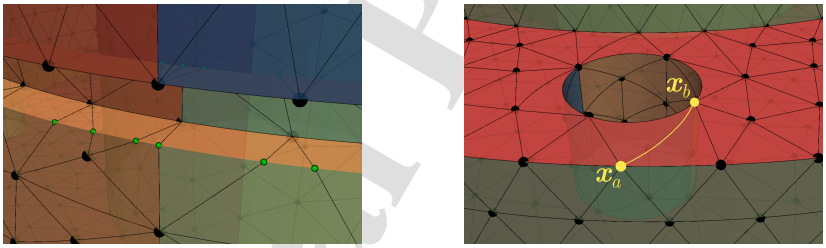


(a) FEM mesh (b) NEFEM mesh

Figure 33: Meshes of the flange model.

Table 8: Surface mesh statistics for the flange model.

Surface mesh $p = 1$	FEM	NEFEM
Number of nodes	2,139	1,724
Number of elements	4,302	3,472
Number of edges of interest	3,635	2,823
Minimum normalised edge length	0.1913	0.7083
Average normalised edge length	0.7988	1.0971



(a) Local view near the U–shaped channel (b) Local view near a satellite hole

Figure 34: Detailed views of the NEFEM mesh for the flange model. The highlighted edge in (b) is curved to ensure validity.

original mesh. Two detailed views of the NEFEM mesh are presented in Figure 34. The view, near the U–channel, shows elements that cross multiple intersections. The view, near one of the satellite holes, illustrates that the technique introduced in Section 3.3.4 is used to avoid the intersection of edges with trimming curves. The highlighted edge in Figure 34(b) between nodes $\boldsymbol{x}_a$ and $\boldsymbol{x}_b$ is intentionally curved so that it would not intersect with the other edge on the circular hole near $\boldsymbol{x}_b$. Figure 35 shows the histograms for the two meshes, with he NEFEM mesh clearly improving the compliance of the mesh size and increasing the minimum element size.

5.5. The Eiffel Tower

The final example involves a model of the Eiffel Tower and demonstrates the ability to handle large and complex geometric models. The CAD model, illustrated in Figure 36, contains 12,034 NURBS surfaces and 3,139 NURBS curves. Numerous features, including common geometric issues reported in [29] such as sliver surfaces, narrow regions, sharp angles, short edges and fillets are included in this complex model. The

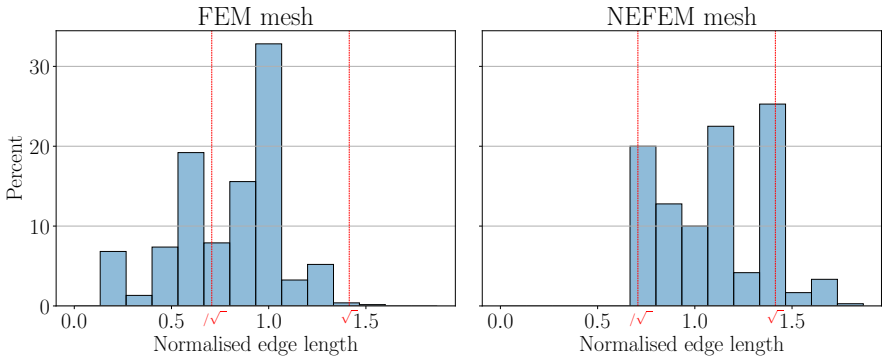


Figure 35: Histograms of edges of interest in FEM and NEFEM meshes for the flange model.

Table 9: Geometric data of the Eiffel Tower model.

Number of NURBS Surfaces	3,139
Number of NURBS Curves	12,034
Minimum curve length	0.23
Maximum curve length	452.61

Table 10: Surface mesh statistics for the Eiffel Tower 1/8 model.

Surface mesh $p = 1$	FEM	NEFEM
Number of nodes	17,519	12,164
Number of elements	33,776	23,112
Number of edge of interest	16,006	12,475
Minimum normalised edge length	0.0229	0.5118
Average normalised edge length	0.7816	1.0311

characteristic dimensions of the model are listed in Table 9. It can be seen that,in this model, the maximum curve length is 452.61, which is almost 2,000 times larger than the minimum curve length of 0.23. This challenging ratio results in a large number of non–compliant elements in the initial FEM mesh.

Taking advantage of the symmetry of the geometry, a one–eighth model is considered for the mesh generation. In this sub–model, the number of NURBS surfaces and curves have been reduced to 460 and 1,828, respectively. The large size of the model means that the global view cannot clearly illustrate the mesh, at an affordable resolution. Instead, the detailed views at the featured locations labelled from A to D in Figure 36(a) are presented in Figure 37 to 40, respectively. Figure 37 demonstrates that the curved surfaces at the tower base do not limit the NEFEM element size, as they do in the FEM mesh. Figure 38 shows that the sliver fillet surfaces are traversed by NEFEM elements, ensuring a better compliance with the user–defined spacing function. The sharp angle at the tangent point between the cylindrical surface and the bottom plane, as shown in Figure 39, induces the creation of small elements with large aspect ratio in the FEM mesh. This is avoided in the NEFEM mesh. Figure 40 shows the FEM and NEFEM meshes at location D, where multiple geometric features are present. The statistics of both the FEM and NEFEM meshes are listed in Table 10. Along with the reduction in the number of elements and nodes, taking into account the edges of interest, the NEFEM mesh has a minimum edge length which is about 23 times larger than the minimum edge length of the original FEM mesh. The histograms, shown in Figure 41, demonstrate

(a) Semi–transparent surfaces and opaque curves

(b) Coloured surfaces

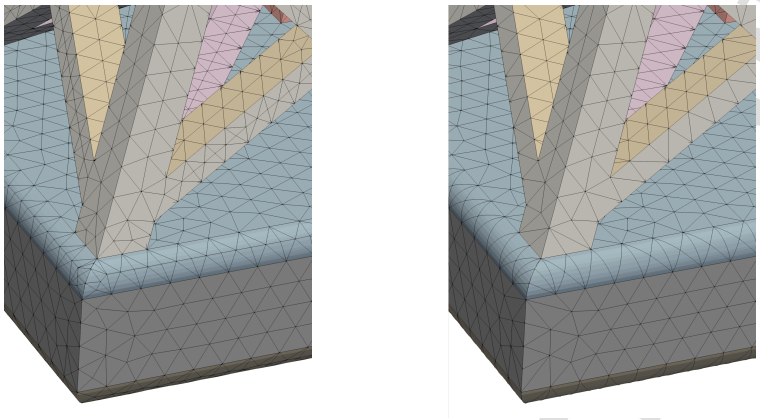
Figure 36: CAD model for the Eiffel Tower. Featured locations are labelled to show mesh details.

the improved element size in the NEFEM mesh. In addition, the percentage of edges with non–compliant length is significantly decreased in the NEFEM mesh, where only about 20% edges achieve half of the desired edge length. Again, a few edges of interest in the NEFEM mesh fall out of the desired spacing range, and this can be improved by further processing such as smoothing the mesh.

6. Concluding remarks

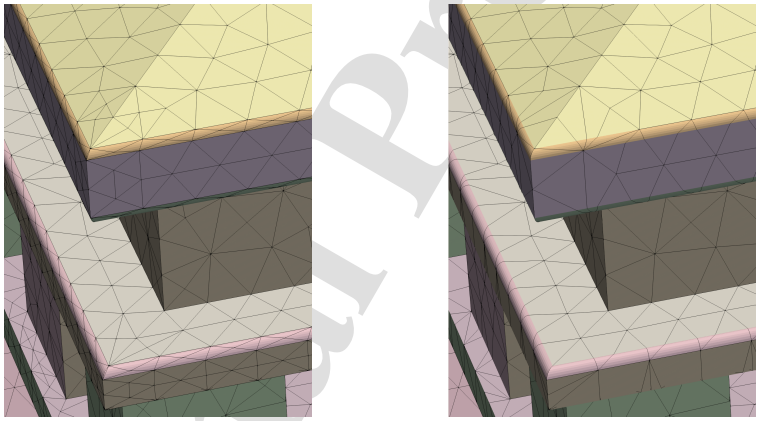
This work presents a novel surface mesh generation technique tailored to NEFEM. The generated meshes contain elements that span across multiple surfaces and demonstrate much better compliance with the user–defined spacing, even in the presence of very small geometric features. The resulting meshes completely avoid the need for de–featuring complex geometric models that contain multi–scale geometric features and, at the same time, preserve the exact representation of the original CAD model. These features are unique to the present technique and enable the geometry to be persistent throughout the whole simulation process.

A new geometric definition of curved edges and curved surface elements is introduced, extending the definitions employed in the original NEFEM approach. With the requirements for a NEFEM surface mesh identified, a novel approach is developed. This relies on an extension of operations, commonly found in mesh



(a) FEM mesh (b) NEFEM mesh

Figure 37: Meshes for the Eiffel Tower with zoomed in views at featured location A.

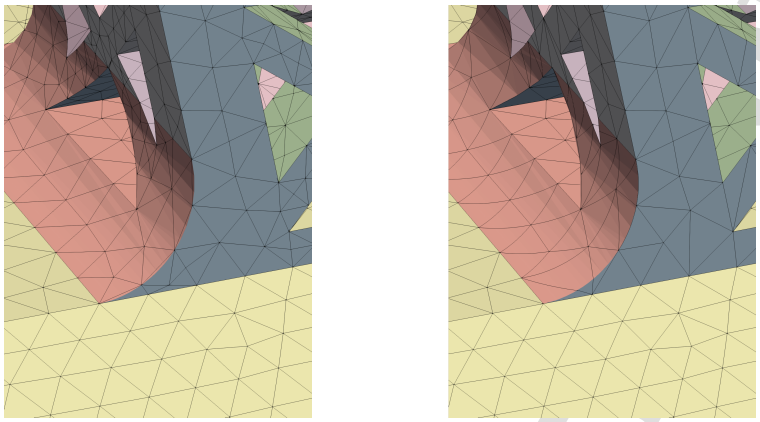


(a) FEM mesh (b) NEFEM mesh

Figure 38: Meshes for the Eiffel Tower with zoomed in views at featured location B.

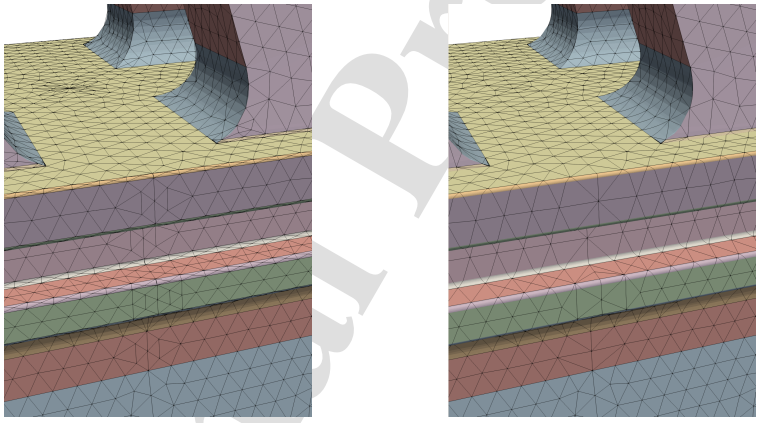
generators, such as edge collapse and edge split. The extension is required to ensure that edges can traverse multiple physical p–curves, and the concept of GS–points is introduced. A simple check for the validity of the surface meshes is introduced and a simple fix that consists of curving internal edges is performed, when problematic elements appear due to the presence of trimmed NURBS surfaces in the CAD model. Although this work does not specifically address mesh quality measures, two basic operations are devised to redefine badly shaped elements.

The generation of high–order nodal distributions on NEFEM surface elements is also addressed. More precisely, novel strategies are devised to define an arbitrary high–order nodal distribution in elements that span across multiple surfaces. It is worth noting, that contrary to mesh generators suitable for isoparametric finite elements, the exact NURBS description is considered for any order of approximation.



(a) FEM mesh (b) NEFEM mesh

Figure 39: Meshes for the Eiffel Tower with zoomed in views at featured location C.



(a) FEM mesh (b) NEFEM mesh

Figure 40: Meshes for the Eiffel Tower with zoomed in views at featured location D.

A set of numerical examples has been presented to demonstrate the potential of the surface mesh generator. The examples include geometries relevant in different engineering applications and show the possibility of creating elements spanning multiple surfaces, even when the normal changes abruptly between the surfaces. This is in contrast to existing approaches based on the virtual topology paradigm. In all the examples, the resulting NEFEM meshes contain a slightly lower number of nodes and elements, but, most importantly, the minimum element size is significantly increased with respect to the original FEM meshes. This is expected to provide significant advantages for the NEFEM solver, as it will alleviate the severe restriction in the time step size that the small elements in the FEM mesh impose when attempting to guarantee numerical stability in an explicit time marching algorithm.

The surface mesh generator can be directly used within a NURBS–enhanced boundary element frame-

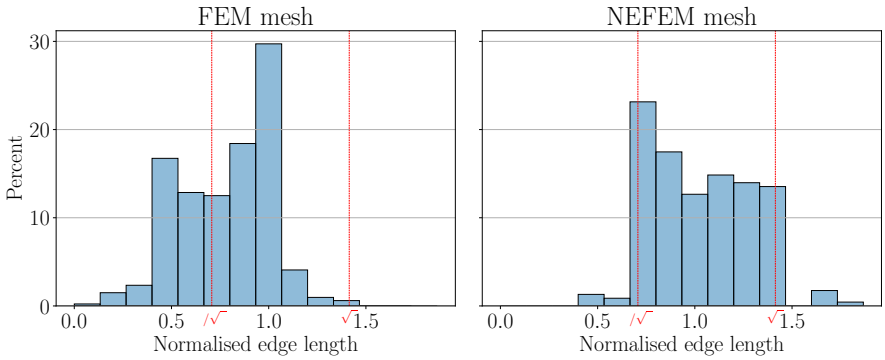


Figure 41: Histograms of edges of interest in FEM and NEFEM meshes for the Eiffel Tower model.

work, but it also serves as the main building block for the development of a NEFEM volume mesh generator, which is the object of current research.

Acknowledgements

The authors gratefully acknowledge the financial support of the EPSRC project *A feature–independent mesh generation and integrated solution framework* with reference EP/T009071/1.

References

[1] A. Thakur, A. G. Banerjee, S. K. Gupta, A survey of CAD model simplification techniques for physics-based simulation applications, Computer-Aided Design 41 (2) (2009) 65–80. doi:10.1016/j.cad.2008.11.009.

[2] V. Shapiro, I. Tsukanov, A. Grishin, Geometric issues in computer aided design/computer aided engineering integration, Journal of Computing and Information Science in Engineering 11 (2) (2011).

[3] A. Katz, V. Sankaran, Mesh quality effects on the accuracy of cfd solutions on unstructured meshes, Journal of Computational Physics 230 (20) (2011) 7670–7686.

[4] A. V. Mobley, M. P. Carroll, S. A. Canann, An object oriented approach to geometry defeaturing for finite element meshing., in: IMR, Citeseer, 1998, pp. 547–563.

[5] R. Sevilla, A. Huerta, HDG-NEFEM with degree adaptivity for Stokes flows, Journal of Scientific Computing 77 (3) (2018) 1953–1980.

[6] R. Sevilla, HDG-NEFEM for two dimensional linear elasticity, Computers & Structures 220 (2019) 69–80. doi:10.1016/j.compstruc.2019.05.005.

[7] M. Giacomini, R. Sevilla, Discontinuous Galerkin approximations in computational mechanics: hybridization, exact geometry and degree adaptivity, SN Applied Sciences 1 (9) (2019) 1–15.

[8] R. Sevilla, S. Fernández-Méndez, A. Huerta, NURBS-enhanced finite element method (NEFEM), International Journal for Numerical Methods in Engineering 76 (1) (2008) 56–83. doi:10.1002/nme.3164.

[9] R. Sevilla, S. Fernández-Méndez, A. Huerta, 3D-NURBS-enhanced finite element method (NEFEM), International Journal for Numerical Methods in Engineering 88 (2) (2011) 103–125. doi:10.1002/nme.3164.

[10] M. Dawson, R. Sevilla, K. Morgan, The application of a high-order discontinuous Galerkin time-domain method for the computation of electromagnetic resonant modes, Applied Mathematical Modelling 55 (2018) 94–108. doi:10.1016/j.apm.2017.10.030.

[11] A. Sheffer, M. Bercovier, T. Blacker, J. Clements, Virtual topology operators for meshing, International Journal of Computational Geometry & Applications 10 (03) (2000) 309–331. doi:10.1142/s0218195900000188.

[12] G. Legrain, A NURBS enhanced extended finite element approach for unfitted CAD analysis, Computational Mechanics 52 (4) (2013) 913–929.

[13] O. Marco, R. Sevilla, Y. Zhang, J. J. Ródenas, M. Tur, Exact 3D boundary representation in finite element analysis based on Cartesian grids independent of the geometry, International Journal for Numerical Methods in Engineering 103 (6) (2015) 445–468. doi:10.1002/nme.4914.

[14] H. Navarro-García, R. Sevilla, E. Nadal, J. J. Ródenas, High-order discontinuous Galerkin method for time-domain electromagnetics on geometry-independent Cartesian meshes, International Journal for Numerical Methods in Engineering 122 (24) (2021) 7632–7663.
[15] R. Sevilla, E. Barbieri, NURBS distance fields for extremely curved cracks, Computational Mechanics 54 (6) (2014) 1431–1446. doi:10.1007/s00466-014-1067-4.
[16] F. Greco, L. Coox, F. Maurin, W. Desmet, NURBS-enhanced maximum-entropy schemes, Computer Methods in Applied Mechanics and Engineering 317 (2017) 580–597.
[17] R. Sevilla, L. Rees, O. Hassan, The generation of triangular meshes for NURBS-enhanced FEM, International Journal for Numerical Methods in Engineering 108 (8) (2016) 941–968. doi:10.1002/nme.5247.
[18] R. Sevilla, S. Fernández-Méndez, A. Huerta, Comparison of high-order curved finite elements, International Journal for Numerical Methods in Engineering 87 (8) (2011) 719–734. doi:10.1002/nme.3129.
[19] R. Sevilla, S. Fernández-Méndez, A. Huerta, NURBS-enhanced finite element method (NEFEM): A seamless bridge between CAD and FEM, Archives of Computational Methods in Engineering 18 (4) (2011) 441–484. doi:10.1007/s11831-011-9066-5.
[20] W. J. Gordon, C. A. Hall, Transfinite element methods: Blending-function interpolation over arbitrary curved element domains, Numerische Mathematik 21 (2) (1973) 109–129. doi:10.1007/bf01436298.
[21] A. Coll, R. Ribó, M. Pasenau, E. Escolano, J. Perez, A. Melendo, A. Monros, J. Gárate, GiD v.14 User Manual (2018).
[22] N. Bergemann, C. Pollard, H. Bucklow, M. Gammon, CADfix API for meshing research, Proceedings of the 27th International Meshing Roundtable (2018).
[23] J. F. Thompson, B. K. Soni, N. P. Weatherill, Handbook of grid generation, CRC press, 1998.
[24] P. J. Frey, P.-L. George, Mesh generation. Application to finite elements., 2nd Edition, London: ISTE; Hoboken, NJ: John Wiley, 2008. doi:10.1002/9780470611166.
[25] E. W. Dijkstra, A note on two problems in connexion with graphs, Numerische Mathematik 1 (1) (1959) 269–271. doi:10.1007/BF01386390.
[26] 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. doi:10.1007/s00466-012-0736-4.
[27] M. A. Taylor, B. A. Wingate, R. E. Vincent, An algorithm for computing fekete points in the triangle, SIAM Journal on Numerical Analysis 38 (5) (2000) 1707–1720.
[28] T. Do, L. Chen, J. Tu, Numerical study of the effect of trailing edge bluntness on highly turbulent hydrofoil flow, ANZIAM Journal 47 (2005) C822–C839.
[29] M. Gammon, H. Bucklow, R. Fairey, A review of common geometry issues affecting mesh generation, AIAA Aerospace Sciences Meeting, 2018 (2018). doi:10.2514/6.2018-1402.

Highlights for the submitted paper

The generation of 3D surface meshes for NURBS–enhanced FEM

- The first method to create 3D surface meshes tailored for NEFEM
- Avoiding de-featuring before numerical simulation and eliminating geometric discretisation error
- Elements can traverse multiple surfaces to respect user-specified mesh size
- Significantly improving minimum element size and relaxing time-step limit for explicit analyses

Declaration of interests

☒ 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.

☐ The authors declare the following financial interests/personal relationships which may be considered as potential competing interests: