This is an author produced version of a paper published in:
*International Journal for Numerical Methods in Engineering*

Cronfa URL for this paper:
http://cronfa.swan.ac.uk/Record/cronfa28843

**Paper:**
Sevilla, R., Rees, L. & Hassan, O. (in press). The generation of triangular meshes for NURBS-enhanced FEM.
*International Journal for Numerical Methods in Engineering, 108*(8), 941-968.
http://dx.doi.org/10.1002/nme.5247

This article is brought to you by Swansea University. Any person downloading material is agreeing to abide by the terms of the repository licence. Authors are personally responsible for adhering to publisher restrictions or conditions. When uploading content they are required to comply with their publisher agreement and the SHERPA RoMEO database to judge whether or not it is copyright safe to add this version of the paper to this repository.
http://www.swansea.ac.uk/iss/researchsupport/cronfa-support/
The generation of triangular meshes for NURBS-enhanced FEM

Ruben Sevilla, Luke Rees, Oubay Hassan

Zienkiewicz Centre for Computational Engineering, College of Engineering, Swansea University, Swansea, SA1 8EN, Wales, UK.

Abstract

This paper presents the first method that enables the fully automatic generation of triangular meshes suitable for the so-called NURBS-enhanced finite element method (NEFEM). The meshes generated with the proposed approach account for the CAD boundary representation of the domain given by NURBS curves. The characteristic element size is completely independent on the geometric complexity and on the presence of very small geometric features. The proposed strategy allows to circumvent the time consuming process of de-feature complex geometric models before a finite element mesh suitable for the analysis can be produced. A generalisation of the original definition of a NEFEM element is also proposed, enabling to treat more complicated elements with an edge defined by several NURBS curves or more than one edge defined by different NURBS. Three examples of increasing difficulty demonstrate the applicability of the proposed approach and illustrate the advantages compared to traditional finite element mesh generators. Finally, a simulation of an electromagnetic scattering problem is considered to show the applicability of the generated meshes for finite element analysis.

Keywords: triangular mesh, NEFEM, geometric features, high-order finite elements, NURBS

Email addresses: R.Sevilla@swansea.ac.uk (Ruben Sevilla), L.A.REES.635876@swansea.ac.uk (Luke Rees), O.Hassan@swansea.ac.uk (Oubay Hassan)
1. Introduction

High-order discretisation methods have gained an increased popularity during the last decade due to the potential of providing higher accuracy with a reduced computational cost compared to traditional low-order methods [1, 2, 3, 4, 5]. The advantages of high-order methods are particularly important in wave propagation problems, where low-order methods are known to suffer from excessive dissipation and/or dispersion when propagating waves over long distances [6, 7, 8, 9, 10, 11]. These methods have also attracted significant attention within the computational fluid dynamics community due the ability to propagate vortices over long distances [12, 13, 14, 15].

The use of curved elements is crucial in order to fully exploit the benefits of high-order methods [16, 17, 18, 19, 20]. This has prompted a great interest in the research community on the generation of high-order curvilinear meshes and nowadays there are some approaches to automatically generate such meshes [21, 22, 23, 24, 25, 26]. To fully exploit the potential advantages of high-order methods, very coarse curvilinear meshes and very high-order approximations are preferred, but as the size of the elements increases, the effect of geometric inaccuracies induced by the traditional polynomial approximation of curved boundaries inherent to the isoparametric formulation becomes evident. The error induced by the isoparametric formulation can be up to one order of magnitude higher than a formulation with an exact boundary representation of the computational domain [27]. In addition, complex geometries of large scale objects often contain very small geometric features (e.g. holes and fillets) making necessary to produce extremely refined meshes in some regions of the domain in order to properly represent the geometry under consideration. In many occasions, it is necessary to invest a non-negligible amount of time removing these small features present in the computer aided design (CAD) model before attempting to produce a mesh suitable for finite element analysis [28]. Although the feature removal can be assisted by some semi-automatic tools [29, 30, 31, 32], the problem is not just the huge amount of time that is invested in removing geometric features but the uncertainty that the geometric simplification can generate in the finite element solution obtained in the simplified model. The problem becomes particularly dramatic when different simulations are required in the same geometry. For instance, a geometric feature that is not relevant in a fluid mechanics problem could be extremely relevant when an acoustic problem is solved. Furthermore, features not relevant in electromagnetic simulations at
low frequency can become extremely influential at higher frequencies. Although CAD and numerical simulation are ubiquitous in modern product development, these two technologies are still far from being integrated in a seamless manner. Automatic methods have been developed in the last decade that are capable of ensuring water tightness with minimum user intervention [33]. However, when dealing with complex geometries that contain multi-scale features, it is often necessary to manually remove small geometric features that induce excessive and unnecessary mesh refinement. NURBS-enhanced finite element method (NEFEM) was proposed to bridge the gap between CAD and finite element analysis [34]. The main idea is to define the curved elements in contact with the CAD boundary in terms of the exact boundary description and not using the traditional isoparametric approach. The advantages of NEFEM compared to other curved finite element methods have been studied in detail from a theoretical and practical points of view both in two and three dimensions [35, 27, 36, 37].

Despite all the benefits reported in the literature, the widespread application of NEFEM has been hampered by the lack of an automatic mesh generator able to generate the meshes that will allow to fully exploit its potential. In particular, a mesh generator able to generate elements which size is independent on the complexity of the boundary and that can be used for finite element analysis is not available. In fact, this lack of such an automatic mesh generator for NEFEM has motivated the development of methods that do not require the generation of fitted NEFEM meshes but still use the NEFEM rationale [38, 39, 40].

This paper proposes a novel mesh generation technique that allows to produce triangular meshes where the elements account for the exact boundary representation of the domain irrespectively of the desired element size and the geometrical complexity. In particular the produced meshes contain elements with edges defined by more than one curve, avoiding small elements when very small NURBS curves are used to represent the boundary of the domain. Furthermore, the meshes generated with this technique can produce elements where an edge contains corners of the boundary representation of the domain, enabling to encapsulate small and complex geometric features within coarse triangular elements. The produced meshes allow to extend the NEFEM formalism introduced in [34], where it was assumed that an edge of an element must be defined by at most one curve. This paper also proposes a technique to extend the meshes to higher order by using a solid mechanics analogy only for the elements in contact with the NURBS bound-
ary. Several examples demonstrate not only the applicability and potential of the proposed mesh generation technique but also demonstrate the applicability of the produced meshes by presenting a simulation of the scattering of electromagnetic waves by complex geometric objects using NEFEM.

The outline of the paper is as follows. In Section 2 the NEFEM formulation is recalled and the concept of a NEFEM element is extended with respect to its original definition. Section 3 summarises the mesh requirements and presents the proposed technique to generate linear meshes suitable for NEFEM. The generation of high-order meshes for NEFEM is presented in Section 4. Three numerical examples of increasing difficulty are used in Section 5 to illustrate the potential of the proposed technique and the proposed meshes are used to perform a simulation with NEFEM. Finally, Section 6 summarises the main conclusions of the work that has been presented.

2. NURBS-Enhanced FEM

This section introduces the fundamental concepts of NEFEM in two-dimensional domains and generalises the original definition of NEFEM elements [37].

2.1. NURBS curves

A $q$th-degree non-uniform rational B-spline (NURBS) curve is a piecewise rational function defined in parametric form as

$$C(\lambda) = \left( \sum_{i=0}^{n_{cp}} \nu_i B_i C_i^q(\lambda) \right) / \left( \sum_{i=0}^{n_{cp}} \nu_i C_i^q(\lambda) \right) \quad \lambda \in [0, 1]$$

where $\{B_i\}$ are the coordinates of the $n_{cp} + 1$ control points (forming the control polygon), $\{\nu_i\}$ are the control weights, and $\{C_i^q(\lambda)\}$ are the normalized B-spline basis functions of degree $q$, which are defined recursively by

$$C_i^0(\lambda) = \begin{cases} 1 & \text{if } \lambda \in [\lambda_i, \lambda_{i+1}) \\ 0 & \text{elsewhere} \end{cases}$$

$$C_i^k(\lambda) = \frac{\lambda - \lambda_i}{\lambda_{i+k} - \lambda_i} C_i^{k-1}(\lambda) + \frac{\lambda_{i+k+1} - \lambda}{\lambda_{i+k+1} - \lambda_{i+1}} C_{i+1}^{k-1}(\lambda)$$

for $k = 1 \ldots q$, where $\lambda_i$, for $i = 0, \ldots, n_k$, are the knots or breakpoints, which are assumed ordered $0 \leq \lambda_i \leq \lambda_{i+1} \leq 1$. They form the so-called knot vector

$$\Lambda = \{0, \ldots, 0, \lambda_{q+1}, \ldots, \lambda_{n_k-q-1}, 1, \ldots, 1\}$$
which uniquely describes the B-spline basis functions. The multiplicity of
a knot, when it is larger than one, determines the decrease in the number
of continuous derivatives. The number of control points, \( n_{cp} + 1 \), and knots,
\( n_k + 1 \), are related to the degree of the parametrization, \( q \), by the relation
\[ n_k = n_{cp} + q + 1 \] [41].

2.2. NEFEM triangular elements

An open bounded domain \( \Omega \in \mathbb{R}^2 \) is considered, where its boundary \( \partial \Omega \)
is described using NURBS curves \( C^k \) for \( k = 1, \ldots, M \), with \( M \) the total
number of boundary curves

\[ \partial \Omega = \bigcup_{j=1}^{M} C_j([0, 1]) \]

A regular partition of the domain \( \overline{\Omega} = \bigcup_e \overline{\Omega_e} \) in triangles is assumed, such
that \( \Omega_i \cap \Omega_j = \emptyset \), for \( i \neq j \).

2.2.1. Original definition of NEFEM triangular elements

The original definition of a NEFEM triangular element [34] assumes that

- curved elements have, at most, one edge on the boundary,
- every curved edge belongs to a unique NURBS curve and
- internal edges are straight.

If \( \Gamma \) represents the edge of the element \( \Omega_e \) on the NURBS boundary
parametrized by \( C \), and \( x_1, x_2 \in \partial \Omega \) the two vertices on the NURBS bound-
dary, see Figure 1, the curved edge is defined by

\[ \Gamma := C([\lambda_1, \lambda_2]) \]

where \( C(\lambda_1) = x_1 \) and \( C(\lambda_2) = x_2 \). Then, a curved triangular element is
defined as a convex linear combination of the curved edge and the interior
node as \( \Omega_e := \psi_{x_3}(R) \), where \( R = [\lambda_1, \lambda_2] \times [0, 1] \), \( x_3 \) is the internal vertex
of \( \Omega_e \) and

\[ \psi_{x_3}(\lambda, \vartheta) := (1 - \vartheta)C(\lambda) + \vartheta x_3 \] (1)

as illustrated in Figure 1.
Remark 1 (Visibility condition). The parametrisation (1) used to define curved elements with one edge defined by one NURBS curve implicitly assumes that the straight segments connecting the interior node $x_3$ with the points in $\Gamma$ are contained in the domain $\Omega$. This means that

$$\left(C(\lambda) - x_3\right) \cdot n \geq 0 \quad \forall \lambda \in [\lambda_1, \lambda_2]$$

where $n$ denotes the outward normal vector to $\Gamma$.

2.2.2. Extending the definition of NEFEM triangular elements

In this work, the original definition of a NEFEM triangular element is extended to include elements with more than one boundary edge, elements with boundary edges defined by more than one NURBS curve and elements that do not fulfil the so-called visibility condition detailed in Remark 1. As it will be shown, this extension facilitates the possibility of generating meshes with a characteristic element size completely independent of the complexity of the NURBS boundary. To simplify the presentation and without loss of generality, it is still assumed that internal edges are straight.

An edge $\Gamma$ on the boundary with vertices $x_1$ and $x_2$ is defined by

$$\Gamma := \bigcup_{j=1}^{n_c} C_j([\lambda_1^j, \lambda_2^j])$$

where $n_c$ denotes the total number of curves defining the edge $\Gamma$, $C_1(\lambda_1^1) = x_1$ and $C_{n_c}(\lambda_2^{n_c}) = x_2$. The boundary of the element $\partial \Omega_e$ is assumed to be a closed piecewise curve formed by trimmed NURBS curves and straight segments.
Formally, an element $\Omega_e$ can be defined using the Jordan’s curve theorem [42] as the subdomain

$$\Omega_e := \{ x \in \Omega \mid \text{ind}_{\partial\Omega_e}(x) \text{ is odd} \}$$

where $\text{ind}_{\partial\Omega_e}$ is the index or winding number of a point $x$ with respect to the closed curve $\partial\Omega_e$ formed by the edges of the element. Figure 2 shows a triangular element that satisfies the new definition of a NEFEM element but does not satisfy the original definition. It can be observed that the triangle has two edges defined by NURBS curves and one edge is defined by two NURBS curves. In addition, the element does not satisfy the visibility condition in Remark 1.

![Figure 2: A valid NEFEM element with the new definition that does not fulfil the original requirements stated in Section 2.2.1.](image)

2.3. NEFEM rationale

A distinctive feature of NEFEM is the definition of the polynomial approximation in the physical space, with Cartesian coordinates, rather than in the reference element with local coordinates. This choice ensures the reproducibility of polynomials in the physical space and has been shown to be advantageous compared to the traditional approximation with local coordinates [27]. For simplicity, Lagrange nodal basis are considered to define the polynomial approximation although other basis (e.g. hierarchical) can be considered.

NEFEM performs the computation of the integrals appearing in a weak variational formulation using the exact boundary description of the computational domain. This section describes the strategy to compute the boundary and element integrals for the new definition of NEFEM elements introduced in Section 2.2.2.
2.3.1. Boundary integrals

A boundary integral to be computed along an edge on the boundary given by (3) can be written as

\[ \int_{\Gamma} f(x, y) \, d\ell = \sum_{j=1}^{n_c} \int_{\lambda^j_1}^{\lambda^j_2} f(C_j(\lambda)) \| C'_j(\lambda) \| \, d\lambda \]

where \( f \) is a generic function (usually a polynomial). It is worth noting that the integral has been split as a summation of integrals over different NURBS curves as the new definition of NEFEM curved elements allow the possibility to have one edge defined by several NURBS. A detailed comparison of different numerical integration techniques to evaluate the integrals of polynomial functions along NURBS curves [43] showed that one dimensional Gaussian quadratures defined over the parametric space of the NURBS provide the most efficient option. Therefore, the boundary integrals are approximated as

\[ \int_{\Gamma} f(x, y) \, d\ell \approx \sum_{j=1}^{n_c} \left( \sum_{i=1}^{n_{ip}^j} f(C_j(\lambda^j_i)) \| C'_j(\lambda^j_i) \| \omega_i^j \right) \]

where \( \lambda^j_i \) and \( \omega_i^j \) are the coordinates and weights of the \( n_{ip}^j \) Gaussian integration points in \([\lambda^j_1, \lambda^j_2]\).

It is worth recalling that a NURBS parametrisation is a piecewise rational function whose definition changes at the breakpoints so it is mandatory to define composite quadratures that account for the discontinuous nature of the parametrisation [43].

2.3.2. Interior integrals

The proposed strategy to compute interior integrals according to the new definition of a NEFEM element is to build a composite two dimensional quadrature in elements affected by the NURBS boundary representation of the domain.

Elements satisfying the hypothesis stated in Section 2.2.1 and the visibility condition detailed in Remark 1 are parametrised using the mapping given in Equation (1), as usual in a NEFEM context. An element \( \Omega_e \) not satisfying any of the previous hypothesis, see an example in Figure 2, can be split into \( K \) cells with only one face described by a NURBS curve and \( L \) straight-sided
cells, namely
\[
\bar{\Omega}_e = \left( \bigcup_{k=1}^{K} \bar{\Omega}_c^k \right) \cup \left( \bigcup_{l=1}^{L} \bar{\Omega}_s^l \right)
\]  
(4)

where \(\Omega_c^k\) denotes a cell with only one edge on defined by one NURBS curve and satisfying the condition (2) and \(\Omega_s^l\) denotes a straight-sided cell. The partition satisfies that \(\Omega_c^i \cap \Omega_c^j = \emptyset\) and \(\Omega_s^i \cap \Omega_s^j = \emptyset\) for \(i \neq j\) and \(\Omega_c^i \cap \Omega_s^j = \emptyset\) for all \(i\) and \(j\).

Figure 3 shows a partition of the NEFEM element shown in Figure 2 with three integration cells that have only one edge defined by one NURBS curve.

Figure 3: Partition of the NEFEM element shown in Figure 2 with three integration cells that have only one edge defined by one NURBS curve.

An interior integral in an element \(\Omega_e\) that has been partitioned according to (4) is computed as
\[
\int_{\Omega_e} f(x, y) \, d\Omega = \sum_{k=1}^{K} \int_{\Omega_c^k} f(x, y) \, d\Omega + \sum_{l=1}^{L} \int_{\Omega_s^l} f(x, y) \, d\Omega
\]

Integrals on cells \(\Omega_c^k\), with at most one edge described by a NURBS curve and satisfying the visibility condition, are computed using the mapping given in Equation (1) whereas integrals in straight-sided cells are computed using standard triangle quadratures [44, 45, 46, 47].

The details of the strategy to perform the partition of elements into integration cells with at most one edge described by a NURBS curve and satisfying the visibility condition are presented in Section 3.5. It is important to emphasise that the subdivision proposed to compute interior integrals is
aimed at simplifying the implementation (i.e., to avoid the definition of a
different mapping for each type of element). It is also worth remarking that
this subdivision does not introduce new degrees of freedom, as the cells are
only used to perform the numerical integration.

3. Generation of low-order NEFEM meshes

A mesh is usually generated in a hierarchical manner. For a two dimen-
sional domain, vertices are meshed with nodes, curves forming the boundary
are discretised with edges and, finally, the domain is meshed using elements.

The technique presented here follows a different rationale and allows to
discretise the boundary with a required element size, independently of small
geometric features. Finally, a variant of the advancing front method is pro-
posed whereby mesh fronts coincident with the NURBS boundary match the
exact geometric definition from CAD independently on the element size.

3.1. Mesh requirements

The following requirements for a NEFEM element are considered:

1. the characteristic element size is not restricted by small geometric fea-
tures
2. element edges on the boundary are exactly described by the CAD
boundary representation of the domain using NURBS
3. the boundary of a NEFEM element is a simple closed or Jordan curve
   (i.e. there is an injective map from a circle to the boundary of the
   element)
4. element edges interior to the domain are straight segments

The first two requirements constitute the basis of NEFEM and ensure
that de-featuring of complex geometries is not required before a finite el-
ment mesh suitable for a NEFEM analysis can be generated. The third
requirement ensures that there are no self-intersections between the edges
of an element. Finally, the last requirement is optional and is only aimed
at improving the performance of the solver. For instance, a discontinuous
Galerkin algorithm for solving Maxwell’s equations in the time domain can
be accelerated 10 times for low-order approximations and up to 100 times for
high-order approximation by considering straight-sided triangles compared
to curved triangles [48].
3.2. Boundary sampling points

The first step consists on defining a set of sampling points for each NURBS curve on the boundary. The approach considered uses a distribution function [49] defined through a background mesh and a set of points and line sources. The sampling points are generated at a distance $\alpha$ times the minimum value of the spacing distribution function, where $\alpha$ is a user defined parameter that by default is taken as 0.1. In practice, the user can also specify the regions where an exact boundary representation is of interest and the set of sampling points will only be built on those boundaries. This avoid the generation of a large set of sampling points in problems such as external flow computations where the far field shape and geometric representation is not relevant.

The initial set of sampling points induces a polygonal description of curved boundaries. Extra sampling points are added in regions where the distance from the straight sided segment connecting two sampling points and the true CAD boundary is more than a specified tolerance. In the proposed implementation, extra points are added when the distance is larger than 10% of the minimum spacing. Next, the desired mesh spacing at the sampling points is computed from the spacing distribution function. Finally, the spacing of the sampling points is checked and corrected if it induces an element on the boundary that is $\varepsilon$ times smaller than the desired element size. Typically, a value of $\varepsilon \in [0.5, 0.75]$ is considered in the implementation of the proposed algorithm.

It is worth noting that the parameter $\alpha$ should be chosen to ensure that all geometric features are captured. Due to the marginal cost associated to the generation of sampling points, the use of small values of $\alpha$ is advocated here, but other techniques based on the generation of sampling points over the Bézier curves forming the NURBS with curvature control could be devised.

3.3. Boundary discretisation

The first mesh requirement described in Section 3.1 implies that the traditional, hierarchical, meshing paradigm cannot be adopted. It is clear that using a hierarchic approach, where the first step consists on adopting the vertices of the CAD model as mesh nodes, implies that small geometric features might dictate the element size. As an example, let us consider a domain $\Omega = [-25, 25]^2 \setminus [-5, 5] \times [-0.25, 0.25]$, representing a square plate with a rectangular inclusion of dimension $10 \times 0.5$ whose centre of gravity is located in the origin, as represented in Figure 4(a). If the desired element size is, for
instance, 3.5 and the traditional hierarchical approach is utilised, the minimum element size will be dictated by the thickness of the inclusion (i.e., 0.5) as illustrated in Figure 4(b).

![Figure 4: (a) Domain \( \Omega \) representing a square plate with a rectangular inclusion and (b) a standard FE mesh showing that the minimum element size is dictated by the thickness of the inclusion.](image)

The proposed approach starts by combining the boundary curves into loops. A loop is defined as a collection of boundary curves \( L := \{C_k\}_{k=1}^{N_L} \), where \( N_L \) is the total number of curves forming the loop, \( C_1(0) = C_{N_L}(1) \) and \( C_k(1) = C_{k+1}(0) \) for \( k = 1, \ldots, N_L - 1 \).

The curves forming the loop are ordered such that the first curve, \( C_1 \), is of minimum length, namely \( L_1([0,1]) \leq L_k([0,1]) \) for \( k = 2, \ldots, N_L \) where the length of a curve \( C_k \) trimmed to the subspace of the parametric space \([\lambda_i, \lambda_j] \) is

\[
L_k([\lambda_i, \lambda_j]) := \int_{C_k([\lambda_i,\lambda_j])} d\ell = \int_{\lambda_i}^{\lambda_j} \|C'_k(\lambda)\| \, d\lambda
\]

and the total length of the loop is denoted by \( L = \sum_{k=1}^{N_L} L_k([0,1]) \).

The proposed procedure to discretise boundary loops with edges of a desired size is detailed in Algorithm 1.
Algorithm 1 Discretisation of a closed loop formed by NURBS curves

Input: Spacing distribution function $h(x)$
Input: NURBS curves forming a closed loop $L := \{C_k\}_{k=1,...,N_L}$
Output: Mesh nodes $\{x_i\}_{i=1,...,M_L}$ discretising the loop $L$

Set $i = 1$, $\lambda_i = 0$, $r = 1$ and $s = 1$
Define the first node as $x_i = C_i(\lambda_i)$

for $i = 2, \ldots, M_L$ do
    $\lambda_i := \text{boundaryNodeAtGivenDistance}(L, r, \lambda_{i-1}, h(x_{i-1}), s)$
    Compute the candidate mesh node, $x_i = C_s(\lambda_i)$
    isValid = checkValidityOfMeshFront($x_{i-1}, x_i, L$)
    if not isValid then
        Get the sampling points within $x_{i-1}$ and $x_i$, $\{\lambda^l_{i-1}\}_{l=1,...,n_s}$, and the associated index
        of the curves where the points belong, $\{s^l\}_{l=1,...,n_s}$
        $\lambda_i := \text{boundaryNodeAtMaxDistance}(L, C_r(\lambda_{i-1}), s)$
        Compute the new mesh node, $x_i = C_s(\lambda_i)$
    end if
end for

A candidate boundary vertex $x_i$ at a distance $h(x_{i-1})$ of a given vertex $x_{i-1}$ is first identified. As the proposed methodology enables the creation of element edges across different boundary curves, it is necessary to identify the curve where the candidate vertex belongs, $C_s$, and the parametric coordinate of the candidate vertex, $\lambda_i$. The strategy to identify both the curve $C_s$ and the parametric coordinate $\lambda_i$ is detailed in Algorithm 2. It is worth noting that Algorithm 2 requires the solution of a one-dimensional non-linear problem. In practice this is achieved using a simple bisection approach as the curves are initially sampled and convergence is achieved in very few iterations.

Next, the mesh front formed by the given vertex $x_{i-1}$ and the candidate vertex $x_i = C_s(\lambda_i)$ is checked. The objective is to guarantee that, in the second stage of the proposed meshing technique described in Section 3.4 (i.e., domain discretisation), an element satisfying the third mesh requirement stated in Section 3.1 exists. The proposed strategy to check the validity of a mesh front is summarised in Algorithm 3.

For a given edge $\Gamma$, the horizon of an edge vertex $x_k$ of $\Gamma$ is defined as the most distant point $x_k^* \in \Gamma$ that can be connected to $x_k$ without intersecting
**Algorithm 2** boundaryNodeAtGivenDistance
Find boundary vertex $x_i$ at a distance $\delta$ from another boundary point $x_{i-1}$

**Input:** Loop $L = \{C_k\}_{k=1,\ldots,N_L}$

**Input:** Index $r$ of the curve to which the point $x_{i-1}$ belongs

**Input:** Parametric coordinate $\lambda_{i-1}$ of the point $x_{i-1}$

**Input:** Distance $\delta$

**Output:** Parametric coordinate $\lambda_i$ of node $x_i$ and index $s$ of the curve to which the point $x_i$ belongs

1. **Identify the curve to which the vertex $x_i$ must belong**
   
   $s = r$
   
   $l = L_s ([\lambda_{i-1}, 1])$
   
   while $l < d$
     
     $s = s + 1$
     
     $l = l + L_s ([0, 1])$
   
   end while

2. **Compute the parametric coordinate of the next boundary vertex**
   
   if $r = s$ then
     
     Find the parametric coordinate $\lambda_i \in [\lambda_{i-1}, 1]$ of the curve $C_s$ such that
     
     $L_s ([\lambda_{i-1}, \lambda_i]) = \delta$
   
   else
     
     Find the parametric coordinate $\lambda_i \in [0, 1]$ of the curve $C_s$ such that
     
     $L_r ([\lambda_{i-1}, 1]) + \sum_{k=r}^{s-1} L_k ([0, 1]) + L_s ([0, \lambda_i]) = \delta$
   
end if
Algorithm 3 `checkValidityOfMeshFront`
Check the validity of a mesh front with nodes $x_{i-1}$ and $x_i$

**Input:** Edge nodes of a mesh front $x_{i-1}$ and $x_i$

**Input:** NURBS curves, $\{C_i\}_{i=1,\ldots,n_c}$, describing the boundary edge $\Gamma$

**Input:** Desired element size $h$

**Output:** Boolean `isValid` denoting the validity of mesh front with nodes $x_{i-1}$ and $x_i$

1. **Compute the horizon of the boundary vertices**
   
   for $k = i-1, i$ do
   
   Find $x^*_k \in \Gamma$ such that $d(x_k, x^*_k) \geq d(x, x)$, for all $x \in \Gamma$ and $x_kx^*_k \subset \Omega$, where $d$ denotes
   
   the distance function and $xy$ is the straight segment connecting $x$ and $y$
   
   end for

2. **Check if a third node can be found to form a valid element**

   Find $P$ as the intersection between the line connecting $x_{i-1}$ and $x^*_i$ and the line connecting $x_i$ and $x^*_i$

   if $d(x_{i-1}, P) < h$ and $d(x_i, P) < h$ then
   
   isValid = true
   
   else
   
   isValid = false
   
   end if
Γ, as illustrated in Figure 5. The intersection of the lines connecting the boundary vertices and the corresponding horizons, denoted by \( P \) in Figure 5, is used to decide the validity of the mesh front. If the distance from \( P \) to the boundary vertices is less than the desired spacing \( h \) this implies that an interior node to form a triangle with the required spacing can be found. An example of a non-valid mesh front is shown in Figure 6.

As detailed in Algorithm 1, if the mesh front is valid, the second boundary vertex \( x_i \) is accepted. Otherwise, a more restrictive criteria is applied in order to find the boundary vertex \( x_i \) that will guarantee a valid mesh front. The procedure, summarised in Algorithm 4, uses the set of sampling points defined in Section 3.2 to find a boundary edge of maximum length that forms a valid mesh front.
Algorithm 4 boundaryNodeAtMaxDistance
Find the second vertex of a boundary edge $\Gamma$ using sampling points

**Input:** Loop $L = \{C_k\}_{k=1,\ldots,N_L}$

**Input:** Boundary vertex $x_{i-1}$

**Input:** Parametric coordinates of the sampling points $\{\lambda_{l-1}^i\}_{l=1,\ldots,n_s}$

**Input:** Index of the curve to which each sampling point belongs $\{s^l\}_{l=1,\ldots,n_s}$

**Output:** Parametric coordinate $\lambda_i$ of node $x_i$ and index $s$ of the curve to which the point $x_i$ belongs

Set isValid = false, $l = n_s$, $\lambda_i = \lambda_{l-1}^i$ and $s = s^l$

while isValid = false do
  Compute the candidate mesh node, $x_i = C_s(\lambda_i)$
  isValid = checkValidityOfMeshFront($x_{i-1}$, $x_i$, $L$)
  Update $l = l - 1$, $\lambda_i = \lambda_{l-1}^i$ and $s = s^l$
end while

Starting with the position of $x_i$ that produces a non-valid front (marked with a red dot in Figure 7), the sampling points, $\{x_{l-1}^i\}_{l=1,\ldots,n_s}$, are checked sequentially starting with the sampling point closest to the discarded vertex $x_{i-1}$. For each sampling point, the proposed algorithm checks the validity of the mesh front formed by $x_{i-1}$ and the current sampling point until a valid front is found. Figure 7 shows the sampling points that produced invalid mesh front with red crosses. The position of $x_i$ is adjusted to be the position of the first sampling point producing a valid front.

**Remark 2.** The check for the validity of a mesh front proposed in Algorithm 3 is only relevant when the produced mesh has to verify the fourth mesh requirement described in Section 3.1. When curved internal edges are allowed, it is possible to eliminate this step and build the boundary discretisation only based on the desired spacing.

**Remark 3.** In the implementation of the proposed method, the user can specify parts of the boundary where the curves will be grouped in loops and other parts of the domain where the standard, hierarchical, approach will be utilised. For instance, in the domain shown in Figure 4, only the curves defining the inclusion can be selected to form an inner loop and maintain the curves defining the outer part of the domain to be treated in a hierarchical manner. This is of particular importance when problems in an exterior domain are of interest (e.g., external flows, electromagnetic scattering) as the
so-called far field boundary is introduced for computational purposes and its shape and geometric representation is irrelevant.

3.4. Domain discretisation

The advancing front method (AFM) is one of the most popular unstructured mesh generation procedures [50]. The main idea is to use the concept of a generation front, initially a sequence of segments that connects the mesh nodes obtained during the discretisation of the boundary. The procedure used to build a triangular element consists on selecting a segment from the front and looking for the ideal point to form the element. The method is designed to try to form equilateral triangles whenever is possible and it is usually finalised by a mesh quality enhancement procedure [50].

The standard AFM is not applicable to generate meshes suitable for NEFEM because boundary edges are considered to be straight segments. Therefore, if the AFM is applied to the discretisation of the boundary obtained after following the strategy described in Section 3.3, the third mesh requirement stated in Section 3.1 is only guaranteed if the straight segment is inside the domain $\Omega$. This is illustrated in Figure 8. A zoom of the domain represented in Figure 4 is shown in Figure 8(a). Figure 8(b) shows the resulting domain to be discretised with the AFM if the boundary edges are transformed to straight-sided segments. A triangular element produced by the AFM is shown in Figure 8(c). It can be clearly observed in Figure 8(d) that
the elements generated with the standard AFM might be invalid if the exact boundary representation is considered.

This section proposes a variation of the standard AFM that accounts for the exact boundary representation of the domain and meets the mesh requirements stated in Section 3.1. The procedure is described in Algorithm 5.

The first step consists on computing the horizon of the boundary vertices, $x_1^*$ and $x_2^*$, as proposed in Algorithm 3. The intersection of the lines connecting the boundary vertices and their respective horizons is denoted by $P$ as shown in Figure 9. A line of search is defined as the angle bisector of the two lines connecting the boundary vertices and their respective horizons, denoted by a dashed red line in Figure 9. The third vertex of the element is defined as

$$x_3 := P + t n_P$$

where $t$ takes the maximum value that guarantees that the edges $x_1 x_3$ and $x_2 x_3$ do not intersect the boundary and have length not greater than the desired element size $h$.

\[ 19 \]
Algorithm 5 Build a NEFEM element from a mesh front with nodes $\mathbf{x}_1$ and $\mathbf{x}_2$

Input: Edge nodes of a mesh front $\mathbf{x}_1$ and $\mathbf{x}_2$

Input: NURBS curves, $\{C_i\}_{i=1,...,n_c}$, describing the boundary edge $\Gamma$

Input: Desired element size $h$

Output: Third node $\mathbf{x}_3$ to form a triangular NEFEM element $\Omega_e$

1. **Compute the horizon of the boundary vertices**
   
   for $k = 1, 2$ do
   
   Find $\mathbf{x}_k^* \in \Gamma$ such that $d(\mathbf{x}_k^*, \mathbf{x}_k) \geq d(\mathbf{x}_k, \mathbf{x})$, for all $\mathbf{x} \in \Gamma$ and $\overline{\mathbf{x}_k, \mathbf{x}_k^*} \subset \Omega$, where $d$ denotes the distance function and $\overline{\mathbf{x}, \mathbf{y}}$ is the straight segment connecting $\mathbf{x}$ and $\mathbf{y}$
   
   end for

2. **Compute the third node using the modified AFM**

   Find $\mathbf{P}$ as the intersection between the line connecting $\mathbf{x}_1$ and $\mathbf{x}_1^*$ and the line connecting $\mathbf{x}_2$ and $\mathbf{x}_2^*$

   Define $\mathbf{n}_P = ((\mathbf{x}_1^* - \mathbf{x}_1) + (\mathbf{x}_2^* - \mathbf{x}_2))$

   Find $\mathbf{x}_3 \in \Omega$ along the line defined by $\mathbf{P}$ and $\mathbf{n}_P$ such that $d(\mathbf{x}_i, \mathbf{x}_3) \leq h$ and $\overline{\mathbf{x}_i, \mathbf{x}_3} \subset \Omega$ for $i = 1, 2$. 

---

Preprint of Ruben Sevilla, Luke Rees and Oubay Hassa, "The generation of triangular meshes for NURBS-enhanced FEM" International Journal for Numerical Methods in Engineering, 2016, DOI: 10.1002/nme.5247
In some situations, using the bisector to obtain the position of the third node induces the generation of small internal edges, as illustrated in Figure 10(a). In such cases, it is advantageous to introduce a rotation of the line where the third vertex will be positioned. The following expression for the vertex $x_3$ is used when the expression in Equation (5) produces an internal edge much smaller than the desired spacing

$$x_3 := P + t \begin{pmatrix} \cos(\alpha) & -\sin(\alpha) \\ \sin(\alpha) & \cos(\alpha) \end{pmatrix} n_P$$

and the angle $\alpha$ is selected to minimise the difference between the length of both internal edges and the desired spacing.

Figure 10 illustrates the benefit of using this definition for the position of the third vertex. In this example the definition in Equation (5) produces an internal edge of size $0.6h$, being $h$ the desired element size, whereas the definition in Equation (6) produces an internal edge of size $0.9h$. In this figure, the shaded area in blue shows the region of possible positions for the third vertex of the element.

It is worth noting that the proposed modification of the AFM described in Algorithm 5 reduces to the standard AFM if the mesh front is a straight-sided segment.

### 3.5. Numerical integration cells

As mentioned earlier, in Section 2.3.2, the strategy to perform the interior integrals appearing in the weak formulation is to use the convex linear
mapping of Equation (1) [34]. The new definition of curved elements introduced in this work requires a partition of the elements with more than one edge on the NURBS boundary or with an edge defined by more than one NURBS curve. This section presents the proposed strategy to build the integration cells for those elements. It is worth emphasising that the cells are only used to perform the numerical integration, so no new degrees of freedom are introduced.

A recursive technique is proposed in Algorithm 6 to build integration cells that have, at most, one edge defined by a NURBS curve. To simplify the notation, it is assumed that when the element has two edges on the boundary, the internal edge connects the vertices $x_1$ and $x_3$, otherwise the node not on the boundary is $x_3$.

The proposed algorithm uses a point interior to the domain, $Q$ to check the visibility of the sampling boundary points. The point $Q$ is selected to be the third node if the element has only one edge defined by NURBS as illustrated in the example of Figure 11. Otherwise, if the element has two
Algorithm 6 integrationCellsOfAnElement
Compute the integration cells for a boundary element

Input: Vertices forming the element \(x_1, x_2\) and \(x_3\).
Input: NURBS curves, \(\{C_i\}_{i=1,\ldots,n_c}\), describing the boundary edge/s \(\Gamma\)
Input: Sampling points \(\{x_l^i\}_{l=1,\ldots,n_s}\) on the NURBS boundary
Output: Integration cells

if \(x_3 \notin \Gamma\) then
   Set \(Q = x_3\)
else
   Set \(Q = 0.5(x_1 + x_3)\)
end if
Set \(\bar{x}_1 = x_1\) and \(\text{visible}=\text{false}\)
for \(k = 1,\ldots,n_s\) do
   if \(Qx_k^l \not\subset \Omega \) and \(\text{visible}=\text{false}\) then
      Set \(\bar{x}_2 = x_{k-1}^l\) and \(\text{visible}=\text{false}\)
      Add cell with vertices \(\bar{x}_1, \bar{x}_2\) and \(Q\) to the list of integration cells
      \(\bar{x}_1 = \bar{x}_2\)
   else if \(Qx_k^l \subset \Omega\) and \(\text{visible}=\text{false}\) then
      Set \(\bar{x}_2 = x_k^l\) and \(\text{visible}=\text{true}\)
      Add cell with vertices \(\bar{x}_1, \bar{x}_2\) and \(Q\) to the list of integration cells
      \(\bar{x}_m = 0.5(\bar{x}_1 + \bar{x}_2)\)
      subMeshOfAnElement \((\bar{x}_1, \bar{x}_2, \bar{x}_m, \{C_i\}, \{x_{l-1}^i\})\)
   end if
end for
edges on the NURBS boundary, the point \( Q \) is selected to be the mid point of the only internal edge.

![Diagram](image)

Figure 11: Illustration of the Algorithm 6 to build the integration cells of an element. (a) Checking the visibility of sampling points, (b) formation of the first two cells, (c) recursive application of the algorithm and (d) final partition of integration cells.

The visibility of the sampling points is checked starting with the first sampling point \( \mathbf{x}_1^1 = \mathbf{x}_1 \) until a sampling point that is not visible is found. In the example of Figure 11(a), the first non-visible sampling point is \( \mathbf{x}_3^1 \). The first integration cell, in blue in Figure 11(b), is formed using the last visible
point, $x_2$. The algorithm flags the last visible point as $\tilde{x}_1$ and continues until a new visible sampling point is found, $\tilde{x}_2$ in Figure 11(b). This allows to form a second integration cell that has no edges on the NURBS boundary, the cell in red in Figure 11(b). The definition of the point $Q$ is now changed to be the mid-point between $\tilde{x}_1$ and $\tilde{x}_2$ and the algorithm is applied to the triangle formed by $\tilde{x}_1$, $\tilde{x}_2$ and $Q$, producing two new cells as illustrated in Figure 11(c). The final partition in integration cells with at most one edge defined by a NURBS curve is shown in Figure 11(d).

4. Extension to higher order elements

This section proposes a procedure to build high-order meshes from the low-order meshes produced using the method described in Section 3. The proposed strategy is based on the linear elastic analogy [23] but it is simpler because an element-by-element solution is proposed in order to maintain internal edges as straight-sided segments, following the mesh requirements introduced in Section 3.1.

4.1. High-order boundary nodal distribution

An edge $\Gamma$ on the boundary with vertices $x_1$ and $x_{p+1}$, defined in (3), is considered where $C_1(\lambda_1^1) = x_1$ and $C_n(\lambda_n^2) = x_{p+1}$. Algorithm 7 details the proposed procedure to build a high-order nodal distribution on $\Gamma$.

First, the desired distance between high-order boundary nodes is computed. This distance can vary from node to node and depends upon the desired nodal distribution that is specified by the user, namely $\{\xi_k\}_{k=1,\ldots,p+1} \in [0,1]$. Equally-spaced nodal distributions in $[0,1]$ can be easily constructed although other alternatives, with better approximation properties, are available [51]. The second step follows the same rationale of the procedure described in Algorithm 1, in Section 3.3, to discretise boundary curves forming loops, but changes a desired element size $h$ by a desired distance between boundary nodes $l_k$.

For interior edges, considered as straight-sided segments, a high-order nodal distribution of the desired degree $p$ is placed by mapping the nodal distribution $\{\xi_k\}_{k=1,\ldots,p+1} \in [0,1]$ to the edge using the linear mapping uniquely defined by the vertices of the edge.
Algorithm 7: High-order nodal distribution over a boundary edge $\Gamma$

**Input:** Ideal high-order nodal distribution $\{\xi_k\}_{k=1,\ldots, p+1} \in [0,1]$

**Input:** NURBS curves, $\{C_i\}_{i=1,\ldots,n_c}$, describing the boundary edge $\Gamma$

**Output:** High-order nodes $\{x_k\}_{k=2,\ldots,p} \in \Gamma$

1. **Initialisation**
   - Compute the total length of the edge $L_{\Gamma}$
   - Compute the desired spacing between high-order nodes in the physical space:
     \[
     l_k = (\xi_{k+1} - \xi_k) L_{\Gamma}
     \]
   - for $k = 1, \ldots, p$ do
     \[
     \lambda_k = 1
     \]
   - end for
   - Set $s = 1$ and $\lambda_1 = \lambda_1$

2. **Computation of high-order boundary nodes**
   - for $k = 2, \ldots, p$ do
     \[
     \lambda_k := \text{boundaryNodeAtGivenDistance}(\{C_i\}_{i=1,\ldots,n_c}, s, \lambda_{k-1}, l_k)
     \]
     - Compute the new boundary node, $x_k = C_s(\lambda_k)$
   - end for

4.2. **High-order elemental nodal distribution**

For an element $\Omega_e$ with closed boundary $\partial \Omega_e$, formed by three edges $\{\Gamma_i\}_{i=1,2,3}$ and with at least one edge on the NURBS boundary, the triangle formed by the vertices of $\Omega_e$ is denoted by $T_e$. A high-order nodal distribution of the desired degree $p$ is placed in $T_e$. This can be easily performed by mapping the nodal distribution from a reference element to $T_e$ using the linear mapping uniquely defined by the vertices of $T_e$.

Then, the following linear elastic problem is defined

\[
\nabla \cdot \sigma = 0 \quad \text{in } T_e
\]  
(7)

The constitutive relation

\[
\sigma = \frac{E \nu}{(1 + \nu)(1 - 2\nu)} tr(\varepsilon) I + \frac{E}{1 + \nu} \varepsilon
\]  
(8)

relates the stress tensor $\sigma$ to the deformation tensor $\varepsilon$ introducing the Young’s modulus $E$ and Poisson’s ratio $\nu$ for the elastic medium [52, 53]. In the above expression, $I$ is the identity tensor, $tr$ denotes the trace operator and the deformation tensor is defined in terms of the displacement field $u$ as

\[
\varepsilon = \frac{1}{2} (\nabla u + (\nabla u)^T)
\]  
(9)
The problem is closed with the following discrete Dirichlet boundary condition

$$u_k = x_k - x_k^0 \quad \forall x_k \in \partial T_e$$

where $x_k^0$ denotes the position of a high-order edge node in $T_e$ and $x_k$ denotes the position of an edge node obtained from Algorithm 7 described in the previous section. It is worth noting that this represents an homogeneous Dirichlet boundary condition if $x_k$ belongs to an interior edge.

The procedure is completed by solving the linear elastic problem (7). The triangle $T_e$ represents the initial, undeformed, configuration. After solving the elastic problem, a high-order nodal distribution is obtained in $\Omega_e$. It is worth remarking that due to the nature of the imposed boundary conditions and the solution of the problem element-by-element, if $T_e$ satisfies the mesh requirements specified in Section 3.1, then the element $\Omega_e$ produced with this strategy also satisfies the same mesh requirements.

Figure 12 illustrates the proposed procedure to build a high-order nodal distribution in a NEFEM element. Figure 12(a) shows a fourth-order nodal distribution with equally-spaced nodes in $T_e$. The boundary nodal distribution obtained from Algorithm 7 is used to define the Dirichlet boundary condition illustrated in Figure 12(b) with arrows. The resulting nodal distribution in $\Omega_e$ is also represented in Figure 12(b).
Remark 4. As mentioned earlier in Section 2.3, NEFEM defines the approximation directly in the physical space, with Cartesian coordinates. Lagrangian polynomials are directly built in $\Omega_e$ based upon the position of the high-order nodes obtained with the elastic analogy [34, 27]. This gives complete freedom on the position of the nodes in $\Omega_e$, contrary to the isoparametric FE formulation, where the position of the nodes in the physical element must be carefully selected in order to guarantee the optimality of the approximation [54, 55]. It is therefore possible to introduce source terms in the elastic problem (7) to obtain the high-order nodes in $\Omega_e$ following a particular spatial distribution. In fact, it is even possible to specify the position of the interior nodes without solving an elastic problem. This could represent an advantage if the solution is known to be complex in certain regions as the nodes can be clustered to better represent the physics of the problem under consideration.

Remark 5. The strategy proposed in this section can be adapted to produce meshes conforming to the CAD boundary representation of the domain and with interior curved edges. This can be achieved by solving the elastic problem in the whole domain $\Omega$ instead of performing an element-by-element solution. In fact, a more efficient layer-by-layer approach can be implemented [23]. The first step involves the solution of the elastic problem in a layer of elements surrounding the boundary represented by NURBS. Dirichlet boundary conditions are used for the edges on the boundary and homogeneous Neumann boundary conditions are used in the outer boundary of the layer. The second step consists on solving another elastic problem in a second layer of elements surrounding the first layer. Dirichlet boundary conditions correspond to the displacement field from the outer boundary of the first layer. The procedure continues until the last layer of elements is deformed or until the displacement field of one layer is small enough.

5. Examples

This section presents three examples that illustrate the applicability of the proposed approach for the generation of meshes suitable for NEFEM. The meshes are analysed in terms of the desired element size and the actual minimum element size of the produced meshes. An application example is also considered where the produced mesh of a complex object is utilised to perform an electromagnetic scattering simulation with NEFEM.

In the implementation of the proposed methodology the input format of the CAD geometry is considered an IGES. Other input formats could
be easily considered because the implementation only uses the control points and associated weights and the knot vector for each NURBS boundary entity.

5.1. Low-order mesh around an aerofoil with a blunt trailing edge

The first example involves the generation of a NEFEM mesh around an aerofoil of unit chord length with a blunt trailing edge of dimension 0.001. Figure 13 shows a NEFEM mesh generated with the technique proposed in this paper. The colour field represents, in logarithmic scale, the spacing distribution function that is induced by two point sources placed at the leading and trailing edges and several line sources to control the element size on the upper and on the lower parts of the aerofoil. The desired element size is 0.005 on the leading edge, 0.01 on the trailing edge and 0.025 on the upper and lower parts of the aerofoil.

The geometry of the aerofoil is described using three NURBS curves and the outer (far-field) boundary is treated in the usual hierarchical manner. The total number of elements with an edge and a node on the NURBS boundary is 89 and 92 respectively. Figure 14 shows two histograms, summarising the size of the edges on the NURBS boundary and the edges of elements with a node on the NURBS boundary. The length is normalised dividing by the desired spacing that, for each edge, is computed as the average of the desired spacing at the nodes.

The minimum and maximum generated edge length on the boundary is approximately 0.00532 and 0.0253 respectively, illustrating the ability of the proposed approach to discretise the boundary with a desired element size, irrespectively of small geometric features. In addition, the minimum and maximum edge length for an element with a node on the boundary is 0.0054
Figure 14: Histogram of the normalised length of (a) the edges on the NURBS boundary and (b) the edges of elements with a node on the NURBS boundary. The histograms correspond to the mesh of Figure 13.

and 0.0261 respectively, showing that the required spacing is maintained for those elements in contact with the boundary elements, generated using the modification of the advancing front method. It is worth noting that all the other elements (i.e., elements not in contact with the NURBS boundary) are generated using the standard advancing front method and, therefore, the edge length for those elements is not reported here.

Figure 15 shows a detailed view near the blunt trailing edge for two meshes, generated with a standard mesh generator and with the proposed technique. Figure 15(a) shows that using a standard mesh generator the minimum element size is given by the dimension of the blunt trailing edge, 0.001 in this example. The mesh produced with the proposed technique, shown in Figure 15(b), maintains the desired element spacing even in regions where small geometric features are present. In this case, the size of the elements near the trailing edge is five times bigger than the dimension of the blunt trailing edge.

The main advantage of the proposed meshing technique in this example is not to reduce the number of elements compared to a traditional FE mesh, but to avoid the small elements introduced by a standard FE mesh generator to capture the blunt trailing edge. The standard FE mesh shown in Figure 15(a)
Figure 15: Detailed view of (a) a standard FEM mesh and (b) a NEFEM mesh near the blunt trailing edge of an aerofoil.

has a minimum element size five times smaller than the minimum element size of the NEFEM mesh. This difference in element size might have an important impact in the total CPU time of a simulation when an explicit time marching algorithm is employed. In addition, the proposed technique avoids abrupt transitions of the element size, inducing better approximation properties, specially if the mesh is used for a low order computation.

5.2. Low and high-order meshes around an aircraft profile

The second example involves the generation of a NEFEM mesh around the aircraft profile represented in Figure 16. A detailed view of the front part is also shown, revealing a number of very small features compared to the size of the aircraft.

Figure 16: (a) Domain Ω representing the exterior of an aircraft profile and (b) detailed view of the domain showing a number of small geometric features.
The aircraft profile is described by 53 NURBS curves and the minimum and maximum length of these NURBS curves is $2 \times 10^{-6}$ and 2.5 respectively, differing in more than six orders of magnitude. A mesh generated with the proposed approach and with a uniform desired element size of 0.04 (i.e., 2000 times larger than the smallest geometric feature) is shown in Figure 17. The mesh has 121 elements with one or more edges on the boundary described by NURBS and 119 elements with one node on the NURBS boundary. Figure 18 shows a detailed view of the mesh near the most critical zone containing the NURBS of length $2 \times 10^{-6}$ and the integration cells in three of the elements with an edge defined by four or even five NURBS curves. It is worth noting that the desired element size is larger than the length of 33 of the boundary curves.

The minimum and maximum generated edge length on the NURBS boundary are 0.0319 and 0.0497 respectively, clearly demonstrating the ability to generate meshes of the desired element size even when geometric features of significant small size are present. For the elements with one node on the NURBS boundary, the minimum and maximum generated edge length are 0.0300 and 0.0689 respectively. Figure 19 shows a histogram summarising the normalised length of the edges of elements with at least one node on the NURBS boundary.

To further illustrate the benefits of the proposed meshing technique, Figure 20 shows a detailed view of a mesh produced using a standard finite element mesh generator with the same desired element size. The resulting mesh has 311 edges on the NURBS boundary but, more importantly, the
Figure 18: Details of the NEFEM mesh of Figure 17 near the regions containing the two shortest NURBS curves. The black dots denote the vertices of the generated triangular elements and the red lines are the subcells used for numerical integration.

Figure 19: Histogram of the normalised length of the element edges on the NURBS boundary corresponding to the mesh of Figure 17.

minimum element size is equal to the minimum size of the geometric features in the domain, $2 \times 10^{-6}$ in this example. The mesh generator used here is the two dimensional version of the FLITE system, a mature in-house mesh generator. It worth mentioning that point sources have been added near the regions with very small geometric features to guarantee that the produced
Figure 20: Detailed views of a standard FE mesh near the regions containing the two shortest NURBS curves.

mesh does not encompass abrupt changes of the element size and, therefore, it is a suitable mesh for finite element analysis.

Next, the mesh shown in Figure 17 is extended to high-order by using the technique proposed in Section 4. For a degree of approximation \( p = 5 \), a reference element with the desired nodal distribution is considered. In this example, a Fekette nodal distribution [56] is employed instead of the more traditional equally-spaced nodal distribution, due to the well known superior approximation properties [51, 57]. The nodal distribution is mapped to each element in the mesh using the affine mapping that is uniquely defined by the vertices of each element. The result is a nodal distribution not conforming to the boundary as illustrated in Figure 21. Using the solid mechanics analogy on each element with at least one edge on the NURBS boundary, the nodal distribution is adapted to the exact geometry and the resulting high-order NEFEM mesh is obtained as represented in Figure 22.

When adapting the nodal distribution to the true geometry using the solid mechanics analogy, it is necessary to specify the material parameters (i.e., Poisson’s ratio and Young modulus) for the elastic medium. When the elastic analogy is used to generate high-order curvilinear meshes for isoparametric finite elements [23] the effect of the materials parameters on the quality of the produced meshes is relevant, but this is not the case for NEFEM. First, as NEFEM defines the approximation on the physical space, the scaled Jacobian, typically used by the FE community to evaluate the quality of a FE
Figure 21: Detail of the NEFEM mesh and the fifth order Fekette nodal distribution mapped from the reference element to the imaginary straight-sided element defined by the vertices of each element represented with discontinuous lines.

Figure 22: Detail of the NEFEM mesh and the fifth order nodal distribution adapted to the exact geometry of each element. The red dots denote the vertices of the triangular elements and the black dots the high-order nodal distribution.
mesh, is irrelevant. Second, this work proposes the solution of an element-by-element elastic problem instead of a global problem, to guarantee the last mesh requirement in Section 3.1.

The effect of the Poisson’s ratio has been found to be minimum on the final nodal distribution obtained. As an example, Figure 23 shows two detailed views of the nodal distributions obtained with $\nu = 0$ and $\nu = 0.49$ for an approximation with degree $p = 3$. Despite the large deformation of the elements shown, the variation of the position of the inner point is very small, usually more than 10 times smaller than the element size. For higher degrees of approximation this difference becomes even less noticeable.

As expected, the value of the Young modulus has no effect on the solution of the elastic problem because imposed displacements are considered on the whole boundary.

5.3. **High-order mesh around a satellite profile**

The last example considers a more complex geometry corresponding to a profile of a satellite as shown in Figure 24. In this example there are multiple regions with very small geometric features and an attempt to remove geometric features not relevant for the finite element analysis stage would clearly result in an extremely tedious and time consuming process. The satellite profile is described by 139 NURBS curves and the minimum and maximum length of the NURBS curves are 0.01 and 2.4 respectively.
A mesh generated with the proposed approach and with a desired element size of 0.06 is shown in Figure 25. It is important to notice that, in this example, almost 70% of the curves have a length smaller than the desired element size. The mesh has 2,519 elements, 1,410 nodes and 143 edges on the boundary described by NURBS. The potential of the proposed approach is clearly illustrated by comparing the number of boundary edges and the total number of NURBS describing the boundary. The similarity between these numbers indicates that the majority of the elements will contain edges defined by more than one NURBS curve.

Figure 26 shows a detailed view of the mesh near two zones containing small geometric features. The nodal distribution corresponds to a polynomial approximation with $p = 4$ generated using the solid mechanics analogy presented in Section 4.

The minimum and maximum generated edge length on the NURBS boundary are 0.0393 and 0.0837 respectively whereas the minimum and maximum generated edge length for elements with one node on the NURBS boundary is 0.0296 and 0.1045 respectively. Figure 27 shows a histogram summarising the size of the edges on the NURBS boundary normalised with the desired spacing.
Figure 25: NEFEM mesh around a complex satellite profile.

Figure 26: Detailed views of the NEFEM mesh of Figure 25 near two regions containing small geometric features.
Figure 27: Histogram of the normalised length of the element edges on the NURBS boundary corresponding to the mesh of Figure 25.

| α  | n_{el} | n_{el}' | n_{no} | h_{min} | h_{max} |
|----|--------|--------|-------|--------|--------|
| 0.5| 2,513  | 130    | 1,407 | 0.0273 | 0.1103 |
| 0.2| 2,509  | 130    | 1,405 | 0.0260 | 0.1039 |
| 0.1| 2,519  | 130    | 1,410 | 0.0296 | 0.1045 |
| 0.02| 2,523  | 130    | 1,412 | 0.0298 | 0.1039 |
| 0.01| 2,515  | 130    | 1,408 | 0.0298 | 0.1058 |

Table 1: Effect of the parameter α used to build the distribution of boundary sampling points.

Next, the effect of the parameter α, introduced in Section 3.2 to produce the distribution of sampling points, is studied. Table 1 reports the total number of elements (n_{el}), the number of elements with at least one edge on the NURBS boundary (n_{el}'), the total number of nodes (n_{no}) and the minimum and maximum edge size (h_{min} and h_{max} respectively), for the meshes generated using different values of the sampling parameter α. The results clearly show that the proposed methodology is not sensitive to this parameter, suggesting that a more sophisticated and computationally expensive approach to produce the distribution of sampling points is not required.

As described in Section 3, the overhead of the proposed approach and a standard mesh generator based on the advancing front method is restricted to the generation of elements with at least one edge defined by NURBS curves. In this example, the time spent on generating the elements with at least one
edge on the boundary is 8% of the total mesh generation time. On average, generating a NEFEM element is 2.5 times more expensive than generating a standard triangular element using the advancing front method. It is worth remarking that this time includes the generation of the integration cells as described in Section 3.5.

Finally, to illustrate the applicability of the proposed technique, an electromagnetic scattering problem [58, 8] is solved using the mesh shown in Figure 25. The problem is governed by the transient Maxwell’s equations and consists of an electromagnetic wave travelling from the left to the right and scattered by the satellite profile. The frequency of the wave is such that the satellite, assumed to be a perfect electric conductor, occupies 14 wavelengths. The problem is solved using a high-order discontinuous Galerkin formulation [6] with NEFEM and a standard fourth order Runge-Kutta explicit time integrator. The three components of the scattered field are shown in Figure 28 for the transverse magnetic mode [58].

![Figure 28: Components of the scattered field by a satellite profile computed with NEFEM and $p = 4$ on the mesh shown in Figure 17.](image)

A reference solution is computed on a standard FE mesh that contains 208,843 elements and 105,716 nodes. The element size has been selected to ensure that at least 20 elements per wavelength are present when a degree of approximation $p = 1$ is employed. For this level of mesh refinement, all the geometric details are captured by a standard mesh generator without any extra refinement. Therefore, this example shows the potential of the proposed technique in a high-order context.

A comparison of the radar cross section (RCS), an engineering quantity of interest in electromagnetic scattering problems [58], is shown in Figure 29. The difference between the reference solution and the RCS computed with
NEFEM and the mesh shown in Figure 17 is less than $2 \times 10^{-2}$ in the $L^2$ norm, demonstrating the possibility of computing accurate solutions with extremely coarse meshes generated with the proposed approach.

In this example, the required cost per cycle is reduced by a factor of 87 by using NEFEM with $p=4$ compared to the reference solution computed with linear elements in a finer mesh. In addition, the linear computation required 79 cycles in order to converge the solution to a harmonic steady state with a tolerance of $10^{-6}$ in the RCS, whereas the NEFEM computation required 49 cycles. This means that the total cost reduction factor with NEFEM and the coarse mesh is approximately 140 ($79 \times 87/49$); an improvement of more than two orders of magnitude.

It is worth remarking that the use of standard high-order isoparametric finite elements is not competitive in this example. Due to the small geometric features, the element size is always restricted by the length of the smallest curve. In fact, a high-order solution will introduce a dramatic restriction in the time step, resulting in a more costly solution compared to linear elements. Therefore, the speed up of the proposed NEFEM approach will be even higher than 140 if the comparison was performed with respect to high-order isoparametric finite elements.
6. Concluding remarks

The first fully automatic mesh generation technique for NEFEM has been presented. It allows to generate meshes that account for the CAD boundary representation of the domain given by NURBS curves and with an element size independent on the geometric complexity of the computational domain.

The proposed technique allows to generalise the original definition of a NEFEM element, ensuring that elements with more than one edge on the NURBS boundary or edges defined by more than one NURBS can be considered. A technique to discretise the boundary with a desired spacing and independent on small geometric features is introduced and a modification of the advancing front method is proposed. A method to extend the produced meshes to higher order approximation has also been presented.

Three examples demonstrate the applicability and potential of the proposed approach by generating meshes of complex objects with small geometric features. An analysis of the produced meshes has been presented in terms of the desired element size. Finally, a numerical example involving the simulation of the scattering of electromagnetic waves demonstrates the applicability of the proposed meshes in a NEFEM context.

It is in the scope of a future publication to present the extension of the methodology presented in this paper to three dimensional domains.

References

[1] P. Solin and K. Segeth, Higher-Order Finite Element Methods. Chapman & Hall, 2003.

[2] G. E. Karniadakis and S. J. Sherwin, Spectral/hp element methods for computational fluid dynamics. Oxford University Press, 2nd ed., 2004.

[3] J. S. Hesthaven and T. Warburton, Nodal Discontinuous Galerkin Methods: Algorithms, Analysis, and Applications, vol. 54. Springer, 2008.

[4] N. Kroll, “The ADIGMA project,” in ADIGMA – A European initiative on the development of adaptive higher-order variational methods for aerospace applications (N. Kroll, H. Bieler, H. Deconinck, V. Couaillier, H. van der Ven, and K. Sørensen, eds.), vol. 113 of Notes on Numerical Fluid Mechanics and Multidisciplinary Design, ch. 1, pp. 1–9, Springer, 2010.
[5] A. Huerta, A. Angeloski, X. Roca, and J. Peraire, “Efficiency of high-order elements for continuous and discontinuous Galerkin methods,” *International Journal for Numerical Methods in Engineering*, vol. 96, no. 9, pp. 529–560, 2013.

[6] J. S. Hesthaven and T. Warburton, “Nodal high-order methods on unstructured grids I. time-domain solution of Maxwell’s equations,” *Journal of Computational Physics*, vol. 181, no. 1, pp. 186–221, 2002.

[7] M. Ainsworth, “Dispersive and dissipative behaviour of high order discontinuous Galerkin finite element methods,” *Journal of Computational Physics*, vol. 198, no. 1, pp. 106–130, 2004.

[8] P. D. Ledger, K. Morgan, and O. Hassan, “Frequency and time domain electromagnetic scattering simulations employing higher order edge elements,” *Computer Methods in Applied Mechanics and Engineering*, vol. 194, no. 2-5, pp. 105–125, 2005.

[9] M. König, K. Busch, and J. Niegemann, “The discontinuous Galerkin time–domain method for Maxwell’s equations with anisotropic materials,” *Photonics Nanostruct.*, vol. 8, pp. 303–309, 2010.

[10] H. Bériot, G. Gabard, and E. Perrey-Debain, “Analysis of high-order finite elements for convected wave propagation,” *International Journal for Numerical Methods in Engineering*, vol. 96, no. 11, pp. 665–688, 2013.

[11] H. Bériot, A. Prinn, and G. Gabard, “Efficient implementation of high-order finite elements for Helmholtz problems,” *International Journal for Numerical Methods in Engineering*, 2015.

[12] H. Luo, J. D. Baum, and R. Löhner, “A fast, $p$-multigrid discontinuous Galerkin method for compressible flows at all speeds,” in *Proceedings of the 44th AIAA Aerospace Sciences Meeting and Exhibit*, (Reno, Nevada), AIAA, January 2006.

[13] R. Sevilla, O. Hassan, and K. Morgan, “An analysis of the performance of a high-order stabilised finite element method for simulating compressible flows,” *Computer Methods in Applied Mechanics and Engineering*, vol. 253, pp. 15–27, 2013.
[14] Z. Wang, K. Fidkowski, R. Abgrall, F. Bassi, D. Caraeni, A. Cary, H. Deconinck, R. Hartmann, K. Hillewaert, H. Huynh, et al., “High-order CFD methods: current status and perspective,” *International Journal for Numerical Methods in Fluids*, vol. 72, no. 8, pp. 811–845, 2013.

[15] H. Huynh, Z. J. Wang, and P. Vincent, “High-order methods for computational fluid dynamics: a brief review of compact differential formulations on unstructured grids,” *Computers & Fluids*, vol. 98, pp. 209–220, 2014.

[16] S. Dey, M. S. Shephard, and J. E. Flaherty, “Geometry representation issues associated with p-version finite element computations,” *Computer Methods in Applied Mechanics and Engineering*, vol. 150, pp. 39–55, Dec. 1997.

[17] F. Bassi and S. Rebay, “High-order accurate discontinuous finite element solution of the 2D Euler equations,” *Journal of Computational Physics*, vol. 138, no. 2, pp. 251–285, 1997.

[18] X.-J. Luo, M. S. Shephard, and J.-F. Remacle, “The influence of geometric approximation on the accuracy of higher order methods,” in *8th International Conference on Numerical Grid Generation in Computational Field Simulations*, 2002.

[19] D. Xue and L. Demkowicz, “Control of geometry induced error in hp finite element (FE) simulations. I. Evaluation of FE error for curvilinear geometries,” *International Journal of Numerical Analysis and Modeling*, vol. 2, no. 3, pp. 283–300, 2005.

[20] L. Krivodonova and M. Berger, “High-order accurate implementation of solid wall boundary conditions in curved geometries,” *Journal of Computational Physics*, vol. 211, pp. 492–512, Jan. 2006.

[21] M. S. Shephard, J. E. Flaherty, K. E. Jansen, X. Li, X. Luo, N. Chevaugeon, J.-F. Remacle, M. W. Beall, and R. M. O’Bara, “Adaptive mesh generation for curved domains,” *Applied Numerical Mathematic*, vol. 52, no. 2-3, pp. 251 – 271, 2005.

[22] P.-O. Persson and J. Peraire, “Curved mesh generation and mesh refinement using Lagrangian solid mechanics,” in *Proceedings of the 47th AIAA Aerospace Sciences Meeting and Exhibit*, AIAA, 2009.
[23] Z. Q. Xie, R. Sevilla, O. Hassan, and K. Morgan, “The generation of arbitrary order curved meshes for 3D finite element analysis,” *Compututional Mechanics*, vol. 51, no. 3, pp. 361–374, 2013.

[24] T. Toulorge, C. Geuzaine, J.-F. Remacle, and J. Lambrechts, “Robust untangling of curvilinear meshes,” *Journal of Computational Physics*, vol. 254, pp. 8–26, 2013.

[25] A. Gargallo-Peiró, X. Roca, J. Peraire, and J. Sarrate, “Distortion and quality measures for validating and generating high-order tetrahedral meshes,” *Engineering with Computers*, vol. 31, no. 3, pp. 1–15, 2014.

[26] D. Moxey, D. Ekelschot, U. Keskin, S. Sherwin, and J. Peiró, “A thermoelastic analogy for high-order curvilinear meshing with control of mesh validity and quality,” *Procedia Engineering*, vol. 82, no. 0, pp. 127 – 135, 2014. 23rd International Meshing Roundtable (IMR23).

[27] R. Sevilla, S. Fernández-Méndez, and A. Huerta, “Comparison of high-order curved finite elements,” *International Journal for Numerical Methods in Engineering*, vol. 87, no. 8, pp. 719–734, 2011.

[28] A. Thakur, A. G. Banerjee, and S. K. Gupta, “A survey of CAD model simplification techniques for physics-based simulation applications,” *Computer-Aided Design*, vol. 41, no. 2, pp. 65–80, 2009.

[29] R. Sun, S. Gao, and W. Zhao, “An approach to b-rep model simplification based on region suppression,” *Computers & Graphics*, vol. 34, no. 5, pp. 556–564, 2010.

[30] S. Gao, W. Zhao, H. Lin, F. Yang, and X. Chen, “Feature suppression based CAD mesh model simplification,” *Computer-Aided Design*, vol. 42, no. 12, pp. 1178–1188, 2010.

[31] M. Li, B. Zhang, and R. R. Martin, “Second-order defeaturing error estimation for multiple boundary features,” *International Journal for Numerical Methods in Engineering*, vol. 100, no. 5, pp. 321–346, 2014.

[32] F. Danglade, J.-P. Pernot, and P. Véron, “On the use of machine learning to defeature CAD models for simulation,” *Computer-Aided Design and Applications*, vol. 11, no. 3, pp. 358–368, 2014.
[33] B. M. Smith, T. J. Tautges, and P. P. Wilson, “Sealing faceted surfaces to achieve watertight cad models,” in Proceedings of the 19th International Meshing Roundtable, pp. 177–194, Springer, 2010.

[34] R. Sevilla, S. Fernández-Méndez, and A. Huerta, “NURBS-enhanced finite element method (NEFEM),” International Journal for Numerical Methods in Engineering, vol. 76, no. 1, pp. 56–83, 2008.

[35] R. Sevilla, S. Fernández-Méndez, and A. Huerta, “NURBS-enhanced finite element method for Euler equations,” International Journal for Numerical Methods in Fluids, vol. 57, no. 9, pp. 1051–1069, 2008.

[36] R. Sevilla, S. Fernández-Méndez, and A. Huerta, “3D-NURBS-enhanced finite element method (NEFEM),” International Journal for Numerical Methods in Engineering, vol. 88, no. 2, pp. 103–125, 2011.

[37] R. Sevilla, S. Fernández-Méndez, and A. Huerta, “NURBS-enhanced finite element method (NEFEM): A seamless bridge between CAD and FEM,” Archives of Computational Methods in Engineering, vol. 18, no. 4, pp. 441–484, 2011.

[38] G. Legrain, “A NURBS enhanced extended finite element approach for unfitted CAD analysis,” Computational Mechanics, vol. 52, no. 4, pp. 913–929, 2013.

[39] R. Sevilla and E. Barbieri, “NURBS distance fields for extremely curved cracks,” Computational Mechanics, vol. 54, no. 6, pp. 1431–1446, 2014.

[40] O. Marco, R. Sevilla, Y. Zhang, J. J. Ródenas, and 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, vol. 103, no. 6, pp. 445–468, 2015.

[41] L. Piegl and W. Tiller, The NURBS Book. London: Springer-Verlag, 1995.

[42] G. Berg, W. Julian, R. Mines, and F. Richman, “The constructive Jordan curve theorem,” Journal of Mathematics, vol. 5, no. 2, pp. 225–236, 1975.
[43] R. Sevilla and S. Fernández-Méndez, “Numerical integration over 2D NURBS shaped domains with applications to NURBS-enhanced FEM,” *Finite Elements in Analysis and Design*, vol. 47, no. 10, pp. 1209–1220, 2011.

[44] D. A. Dunavant, “High degree efficient symmetrical Gaussian quadrature rules for the triangle,” *International Journal for Numerical Methods in Engineering*, vol. 21, no. 6, pp. 1129–1148, 1985.

[45] S. Wandzura and H. Xiao, “Symmetric quadrature rules on a triangle,” *Computers & Mathematics with Applications*, vol. 45, no. 12, pp. 1829–1840, 2003.

[46] C. A. Felippa, “A compendium of FEM integration formulas for symbolic work,” *Engineering Computations*, vol. 21, no. 8, pp. 867–890, 2004.

[47] F. Witherden and P. Vincent, “On the identification of symmetric quadrature rules for finite element methods,” *Computers & Mathematics with Applications*, vol. 69, no. 10, pp. 1232–1241, 2015.

[48] R. Sevilla, O. Hassan, and K. Morgan, “The use of hybrid meshes to improve the efficiency of a discontinuous Galerkin method for the solution of Maxwell’s equations,” *Computers & Structures*, vol. 137, pp. 2–13, 2014.

[49] J. Peiró, “Surface grid generation,” in *Handbook of grid generation* (J. F. Thompson, B. K. Soni, and N. P. Weatherill, eds.), ch. 19, CRC Press, 1999.

[50] J. Peraire, J. Peiró, and K. Morgan, “Advancing front grid generation,” in *Handbook of grid generation* (J. F. Thompson, B. K. Soni, and N. P. Weatherill, eds.), ch. 17, CRC Press, 1999.

[51] Q. Chen and I. Babuška, “Approximate optimal points for polynomial interpolation of real functions in an interval and in a triangle,” *Computer Methods in Applied Mechanics and Engineering*, vol. 128, no. 3–4, pp. 405–417, 1995.

[52] O. C. Zienkiewicz and R. L. Taylor, *The Finite Element Method*, vol. 1. The basis. Butterwort-Heinemann, fifth ed., 2000.
[53] O. C. Zienkiewicz and K. Morgan, *Finite elements and approximation*. John Wiley & Sons, 1983.

[54] M. Lenoir, “Optimal isoparametric finite elements and error estimates for domains involving curved boundaries,” *SIAM Journal on Numerical Analysis*, vol. 23, no. 3, pp. 562–580, 1986.

[55] C. Bernardi, “Optimal finite-element interpolation on curved domains,” *SIAM Journal on Numerical Analysis*, vol. 26, no. 5, pp. 1212–1240, 1989.

[56] M. A. Taylor, B. A. Wingate, and R. E. Vincent, “An algorithm for computing fekete points in the triangle,” *SIAM Journal on Numerical Analysis*, vol. 38, no. 5, pp. 1707–1720, 2000.

[57] M. G. Blyth, H. Luo, and C. Pozrikidis, “A comparison of interpolation grids over the triangle or the tetrahedron,” *Journal of Engineering Mathematics*, vol. 56, no. 3, pp. 263–272, 2006.

[58] A. Taflove, *Computational Electrodynamics: The Finite-Difference Time-Domain Method*. Artech House, Inc., 1995.