Engineering with Computers (2004) 20: 273–285
DOI 10.1007/s00366-004-0295-1

ORIGINAL ARTICLE

Xiao-Juan Luo · Mark S. Shephard · Robert M. O'Bara
Rocco Nastasia · Mark W. Beall

Automatic p-version mesh generation for curved domains

Received: 28 July 2003 / Accepted: 12 April 2004 / Published online: 28 July 2004

Abstract To achieve the exponential rates of convergence possible with the p-version finite element method requires properly constructed meshes. In the case of piecewise smooth domains, these meshes are characterized by having large curved elements over smooth portions of the domain and geometrically graded curved elements to isolate the edge and vertex singularities that are of interest. This paper presents a procedure under development for the automatic generation of such meshes for general three-dimensional domains defined in solid modeling systems. Two key steps in the procedure are the determination of the singular model edges and vertices, and the creation of geometrically graded elements around those entities. The other key step is the use of general curved element mesh modification procedures to correct any invalid elements created by the curving of mesh entities on the model boundary, which is required to ensure a properly geometric approximation of the domain. Example meshes are included to demonstrate the features of the procedure.

Keywords p-version method · Curved meshes · Graded meshes

1 Introduction

The performance of the p-version finite element method is characterized by its ability to attain exponential rates of convergence, thus, allowing the desired level of accuracy to be obtained with many fewer degrees of freedom than other methods, such as the standard h-version finite element method. However, to attain the exponential rates of convergence, the meshes must satisfy specific requirements in terms of mesh entity geometric approximation and mesh gradation that are not satisfied by current mesh generation methods developed for h-version meshes of linear or quadratic elements.

As the order of the finite element basis functions are increased, so must the level of geometric approximation of those finite element faces and edges that lie on curved domain boundaries. Otherwise, the error due to geometric approximation will become the dominate error, thus, yielding any further increase in finite element order meaningless [6, 13]. To attain the optimal convergence, p-version finite elements require strongly graded coarse meshes. In the neighborhood of singularities, the meshes must be geometrically graded in the directions of the singular gradients [1–4, 15]. In the smooth portions of the domain away from the singularities, the meshes must be very coarse with satisfactory geometric approximation.

Approaches to generate high-order meshes for the p-version method usually start from linear mesh generation developed for the h-version finite element method, followed by assigning high-order geometric shapes to the mesh entities on the curved model boundaries [5, 14]. Optimization of the surface mesh generation, hybrid meshing with prismatic elements near the domain boundaries, and curvature-driven surface mesh adaptation are the three strategies employed in [14], and the procedure presented in [5] incrementally corrects the invalid high-order elements by applying available local mesh modification operations.

The ideal approach to accomplish p-version mesh generation is to create curved elements of the size and shape desired one at a time. However, the lack of a known algorithm and the high level of computational effort required by such an approach make it unfeasible. A compromise approach is to combine operations to curve a mesh initially constructed based on linear geometry mesh

X.-J. Luo (✉) · M. S. Shephard
Scientific Computation Research Center,
Rensselaer Polytechnic Institute, Troy, NY 12180, USA
E-mail: xluo@scorec.rpi.edu
E-mail: shephard@scorec.rpi.edu

R. M. O'Bara · R. Nastasia · M. W. Beall
Simmetrix Inc., Clifton Park, NY 12065, USA
E-mail: obara@simmetrix.com
E-mail: nastar@simmetrix.com
E-mail: mbeall@simmetrix.com

entity meshing procedures with consideration of desired aspect ratio and sizes of elements throughout the domain. The steps in the automatic p-version mesh generation procedure for general three-dimensional domains currently under development are:

- Automatically isolate the singular edges and vertices in the model
- Generate a coarse linear mesh accounting for the isolated singular features
- Curve the singular feature isolation mesh to maintain a properly curved mesh isolation
- Curve the remaining mesh entities classified on the curved boundaries
- Apply local mesh modifications as needed to ensure a valid mesh is produced

Unlike the case of low-order Lagrangian finite elements where it is convenient to employ the same functions for the finite element basis and geometric shape, the p-version finite basis functions do not represent a convenient choice for defining the geometry of the element. In this work, Bezier polynomials are selected for the geometric representation of the elements because they provide a convenient framework for the construction of geometric approximations of any order and effectively support the geometry-based operations needed in a finite element calculation.

This paper is organized as follows. Section 2 discusses the use of Bezier representations to provide an effective framework for defining the shape of mesh entities to any order as needed for p-version meshes. Section 3 reviews the requirements to isolate geometric singularities with geometrically graded meshes to support optimal hp-version analysis and indicates the algorithm used to isolate them for proper treatment during mesh generation. Section 4 presents an algorithm to generate coarse geometrically graded meshes of linear geometry around the isolated singular geometric entities. The procedure to curve an initial graded linear geometry mesh to proper order to get a curved mesh with right gradation for hp-version analysis is given in Sect. 5. A set of example meshes are given in Sect. 6.

2 Mesh geometry

In the p-version finite element method, the approximation of the mesh to curved geometric domains must be properly maintained when the finite element basis is improved to ensure that the mapping error does not become the dominant error. A simple analysis based on the relation of approximation theory to the convergence of the error in energy norm indicates that the energy norm will converge so long as the geometric approximation of the mesh is within one order of that used in the finite element basis [6]. As for the other norms of interest, there is limited theoretical information or numerical studies about how well the geometric must be approximated. However, a study of a two-dimensional benchmark elasticity problem clearly demonstrates the loss of convergence in the energy norm and pointwise norms of engineering interest when the geometric approximation is not increased as the order of the finite element basis is increased [13]. The improvement of geometric approximation is met by assigning appropriate high-order geometric shapes to mesh entities classified[1] on curved boundaries.

The functional interfaces to solid modeling systems can support the direct use of the geometry of the portion of the model face or model edge that a mesh face or edge is classified on for purposes of performing the geometry-based calculation needed [6]. However, the high computational cost of this method makes it undesirable. The alternative is to assign the mesh entity an appropriate geometric form, with Lagrange and Bezier polynomial being two of the possibilities. An effective procedure for creating quadratic Lagrangian finite element meshes has been developed [5]. However, the extensions of this approach to high-order Lagrangians indicates that the procedure quickly became computationally demanding and geometrically complex. Bezier representations [8, 9] on the other hand provide an attractive alternative for defining element geometries of any order. Advantageous properties of Bezier polynomials include:

- The convex hull property—a Bezier curve, surface, or volume is contained in the convex hull formed by its control points
- The variation diminishing property—an infinite plane cannot intersect a Bezier curve more times than it intersects the control polygon, which allows more efficient intersection calculations
- All derivatives and products of Bezier functions are easily computed Bezier functions
- Computationally efficient algorithms for degree elevation and subdivision are available, which can be used to refine the shape's convex hull as well as adaptively refine the mesh's shape

One important application of the above Bezier properties is the developed curved element shape validity test. In the case of a Bezier tetrahedron volume, the Jacobian $J(\underset{\sim}{\xi})$ of the geometric mapping $\underset{\sim}{x}(\underset{\sim}{\xi})$ is:

$$J(\underset{\sim}{\xi}) = \left| \frac{\partial \underset{\sim}{x}}{\partial \underset{\sim}{\xi}} \right| \tag{1}$$

where ξ represents the natural coordinates of the element. The determinant of Jacobian det(J) is computed as:

$$\det(J) = \left(\frac{\partial \underset{\sim}{x}}{\partial \underset{\sim}{\xi}_0} \times \frac{\partial \underset{\sim}{x}}{\partial \underset{\sim}{\xi}_1} \right) \times \frac{\partial \underset{\sim}{x}}{\partial \underset{\sim}{\xi}_2} \tag{2}$$

[1] A mesh face or edge that lies on a model face is "classified on that model face" and a mesh edge that lies on a model classified on a model edge is "classified on that model edge."

The partial derivatives in Eq. 2 $\partial \underset{\sim}{x}/\partial \xi_0$, $\partial \underset{\sim}{x}/\partial \xi_1$, and $\partial \underset{\sim}{x}/\partial \xi_2$ are the three partial derivatives of $\underset{\sim}{x}(\underset{\sim}{\xi})$, which are also Bezier functions.

Because the product of Bezier functions are also Bezier functions, det(J) can be represented as a polynomial in Bezier form which is bounded by its convex hull of control points. If all of the control points of det(J) are greater than zero, then the region's det(J) must be greater than zero everywhere. Comparing to validity checks that test the value of det(J) at the integration points used in performing the analysis, the new approach has the following advantages for simplex mesh entities:

- The element geometric shape is determined only by its own control points. Once the element is valid, the determinant of Jacobian det(J) should always be greater than zero inside the element, independent of the basis functions, polynomial order, or integration rules of the finite element approximation.
- The relation of det(J) to the control polygon provides insight on how a region found to be invalid due to some portion having negative det(J) can be made valid most effectively.
- These tests can be extended to test the mesh faces and edges bounding the mesh region. All regions using an invalid mesh edge or face (due to self-intersection) as part of its boundaries is also invalid. Identifying and correcting these lower dimension mesh entities can effectively reduce the number of invalid mesh regions that need to be corrected.

An example of an invalid tetrahedron region determined by the above algorithm is shown in Fig. 1. In this case, the placement of control point P_1 causes $a\cdot(b\times c) < 0$ at the mesh vertex.

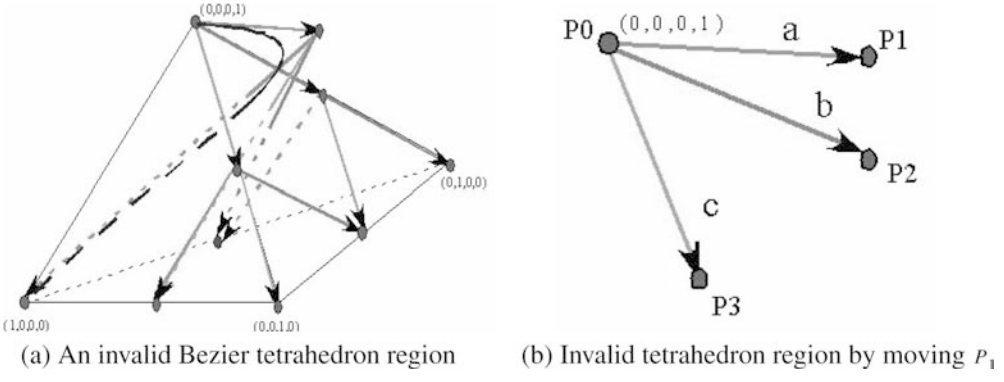
The Bezier shape of a mesh entity is described by a set of control points. In the case of mesh edges classified on the curved model boundaries, their control points are determined based on Bezier curve and surface interpolation methods resulting from evaluating the model geometry at a set of discrete locations. Common methods usually assume the interpolation points are uniformly distributed in the parametric space of the mesh entity. However, this approach can lead to poor geometric approximations. An alternative method that improves the geometric approximation for a given order uses a chord length method [9]. In cases where there are large changes in the curvature in the portion of the model entity being approximated by the mesh entity, the use of a curvature-based procedure for selecting the interpolating points is appropriate [13]. Specific attention needs to be paid to model entities that contain either parametric degeneracies and/or periodicity.

3 Singular features isolation

The character of partial differential equations' exact solution is determined by the shape of the domain, the boundary conditions, the loading functions, and the material parameters. It becomes singular when sharp reentrant corners are presented, or sudden change in material properties, loads, or boundary conditions occur. The influence of the domain's geometric shape on the exact solution has been studied in [1–4, 7, 15], which show the solution is singular in a neighborhood of a reentrant model edge whose interior dihedral angle $w_i > \pi$. The larger the w_i, the stronger the singularity. As long as one of its connecting model edges is singular, the exact solution at the vicinity of a model vertex also has singular behavior [2, 7]. Let $\{G^j\}$ $j=0$, 1 indicate the model vertices and edges of a problem domain. A procedure is needed to define the subset of these entities that are singular, based on the geometric properties of the faces they bound. For example, all of the marked edges and vertices in Fig. 2a are singular.

In the p-version finite element method, meshes should be refined with geometrically graded cylindrical meshes around singular model edges and geometrically graded spherical vertices around singular vertices in order to attain the advantage of exponential convergence rates [1–4, 15]. For example, an appropriate p-version mesh at the neighboring of vertex $G_{16}{}^0$ and $G_1{}^{16}$are shown in Fig. 2b.

Given a problem domain, it is possible to preprocess the geometric domain to mark the model edges and vertices at which the solution will be singular in terms of the interior dihedral angle w_i of the pairs of model faces bounding the model edges. In the case of two-manifold



(a) An invalid Bezier tetrahedron region (b) Invalid tetrahedron region by moving P_1

Fig. 1a, b Determination of an invalid Bezier region. **a** An invalid Bezier tetrahedron region. **b** Invalid tetrahedron region by moving P_1

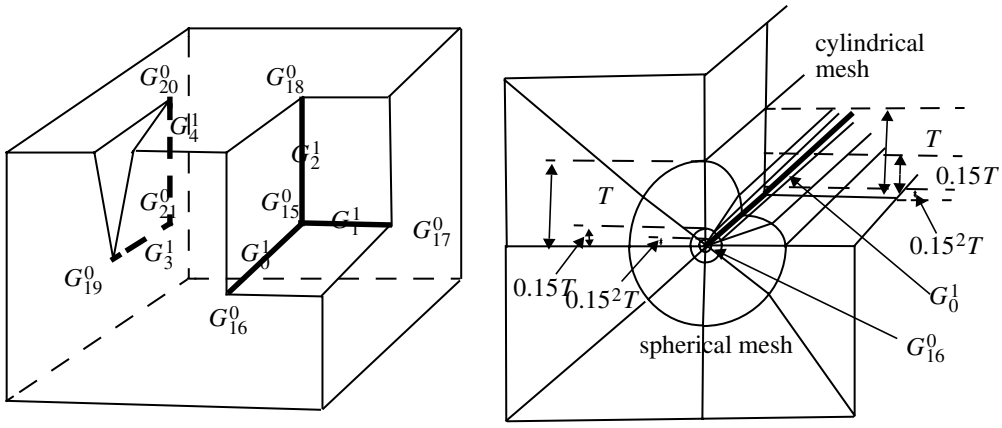


Fig. 2a, b Geometrically singular model features. **a** Model singular features in a 3D domain. **b** Desired mesh for singular model entities

(a) Model singular features in a 3D domain

(b) Desired mesh for singular model entities

domains, the process considers each model edge and the two model faces coming to it. If both of the model faces are planar, the interior dihedral angle is constant along the model edge and can be calculated at any location along the edge. If either one of the model faces is non-planar, then the dihedral angle may no longer be constant. Therefore, the variation of w_i along the model edge needs to be examined. The current procedure uses a simple sampling algorithm along the edge to compute the angle. To be completely effective, this procedure needs to be supplemented with a procedure that will determine all locations where the angle passes through π, which will de-mark singular and non-singular portions. The computation of the angle w_i at the requested locations is performed by queries to the solid modeler that provides face normal and tangent vectors at the specified location. An example of computing $w_i > \pi$ is shown in Fig. 3.

Figure 4 shows an example where the solid model is in the left image and the isolated reentrant model edges are in the right image.

On the assumption that dominate singular behavior is reasonably local to the isolated singular edges, the geometrically graded mesh is created within the neighborhood of the edges. The height of the graded layer mesh T for an isolated singular edge is defined as half of the distance to its closest non-connected model boundary entity. A non-connected entity is one which does not share any common lower-order model entities. A simple algorithm based on the shortest distance between a set of sampling locations to their closest points on non-connected model boundaries is used to get a useful estimation of T. A more advanced algorithm will use bounding boxes to minimize the number of entities checked in detail and will include the use of more intelligent point sampling. The thickness for each layer t_i is calculated as:

$$t_i = \alpha^i T, \quad i = 0, 1, 2 \tag{3}$$

where α is the geometric gradation factor. By default, α is set to 0.15, based on the assumption of a strong singularity [2] and three layers of elements are generated. Figure 5a shows an example where the T of $G_0{}^1$is defined as half of the distance between P_0 and P_1, which are the closest points on non-connected model faces $G_0{}^2$and $G_1{}^2$5b shows the layer definition for $G_0{}^1$with an α of 0.5 simply to make it easier to view the layer construction.

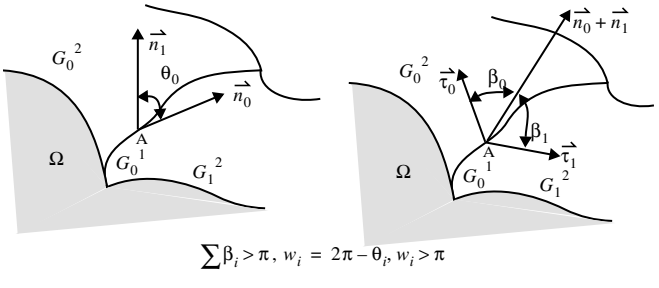


Fig. 3 Computation of dihedral angle w_i in 3D domain

4 Geometrically graded meshes around isolated features

A modified version of the generalized advancing layer method [10] is used to create the structured linear geometry meshes around the singular edges and vertices. The algorithm reliably generates layered meshes with the desired structure around selected model entities for geometric domains of arbitrary complexity. The elements can be all tetrahedra, or a combination of element types, including hexahedra, wedges, and pyramids.

Starting from a linear geometry surface mesh, the algorithm efficiently generates high-quality structured elements by stepping into the domain creating a set of advancing layers following the given geometric progression. Key generalizations to the advancing layers methodology were introduced to deal with the complexities of creating feature isolation meshes in general domains. One new feature is a transitional blend operation to account for large local changes in the model geometry due to high curvature. If an intersection between the exposed faces that may be faces of bound-

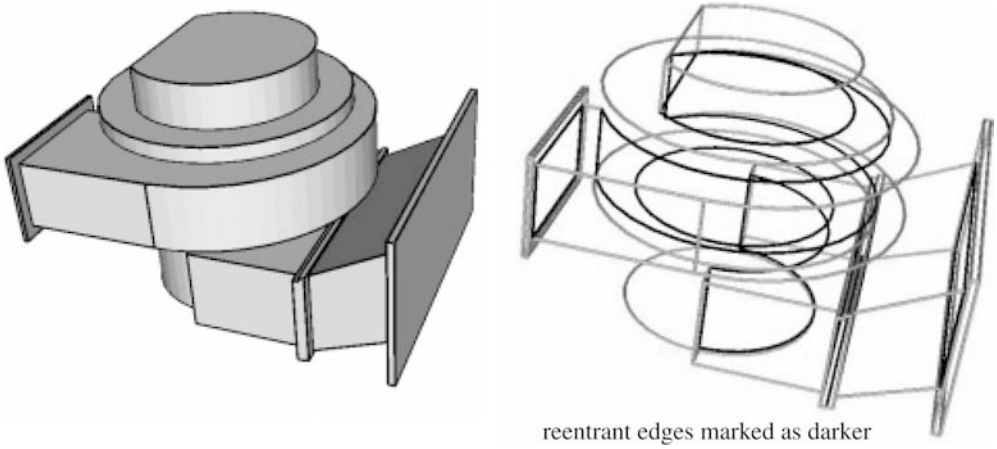


Fig. 4 Automatically marked set of reentrant model edges

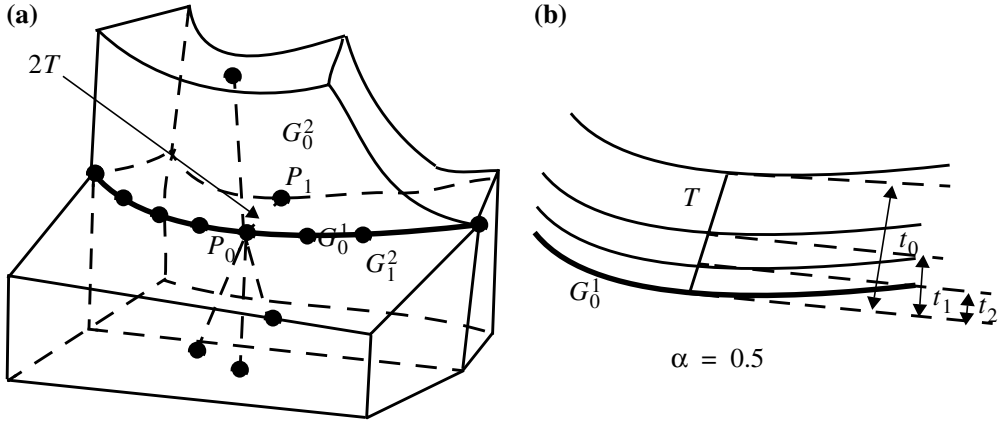


Fig. 5a, b Attributes for an isolated model edge G_0^1

ary layer mesh or faces on model faces is found, it is necessary to reduce the height of a layer, or trim the number layers through the thickness in order to resolve overlapping boundary layers. No layer is shrunk to a height less than the first layer [10].

Each step in the application of these processes performs checks to ensure the validity of the mesh and to optimize the quality of the mesh entities created. These enhancements are quite important to the success of feature isolation when coarse, with respect to local geometric model features, meshes are needed, which is the case with meshes for p-version analysis.

The structured mesh generation is controlled through a set of attributes applied to the model entities being isolated, which, in the case of singularity isolation, are edges and vertices. Attributes can be placed on any topological entity to control mesh entity size and layer gradation. The attributes are the following:

- First layer height
- Total height
- Gradation factor
- Number of layers

If attributes are placed on faces in the model, a typical advancing-front-style mesh of prismatic wedges or hexahedra is created. In the current application, these attributes are assigned to singular edges or vertices of the model. The result in those cases are a cylindrical-like mesh around the edges and a spherical-like mesh around vertices. Figure 6 shows an example of singular edge isolation for a ''penny-shaped crack'' on the interior of a domain. In this example, hexahedral elements were generated around the isolation edge.

5 Process of constructing the curved mesh

Although the ideal approach to accomplish the generation of desired p-version meshes is to create curved mesh

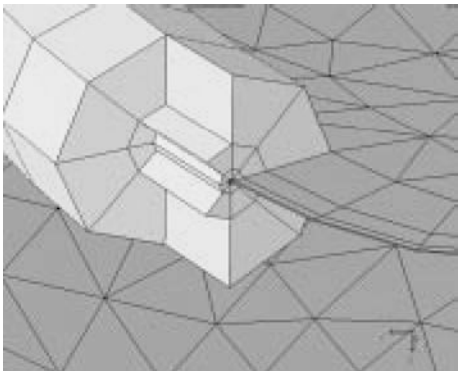


Fig. 6 Example of edge singularity isolation around a crack tip

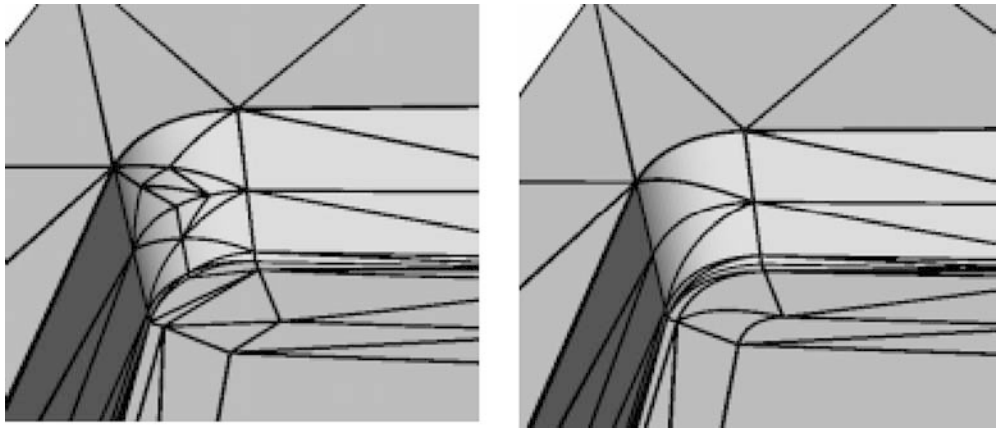


Fig. 7 Meshes without (*left*) and with (*right*) considering the ordering of curving

entities with gradation for isolated singular features and fill the remainder of the domain with optimal elements at a time, the lack of a qualified set of algorithms and the high computational cost that would be required currently makes that approach impractical. A compromised approach is to first generate the graded mesh around the isolated features (Sect. 4) and construct a coarse straight-sided mesh in the rest of the domain and then to incrementally curve the mesh entities in an ordered manner. The local mesh-modification-based procedure to curve a straight-sided mesh for quadratic mesh geometries given in [5] provides key tools for the procedures given here. The current procedure has been extended to:

- Support higher-order mesh entity geometry
- Provide generalized local mesh modification accounting for curved mesh entities
- Maintain the structure of the mesh configuration around isolated singular edges and vertices by carefully ordering and controlling the process of curving mesh entities to ensure the appropriate mesh gradations are maintained

All three of these improvements are important to the generation of curved meshes appropriate for optimal p-version analysis. The proper order and control of the mesh modification operations accounting for the mesh gradations is a most critical aspect of this process. Figure 7 shows the difference in meshes without (left image) and with (right image) consideration of the ordering of mesh curving operations. The mesh curving procedure orders the mesh entity curving into the following two steps:

- Curve the singular feature isolation mesh to ensure a proper structure is maintained in the curved mesh.
- Curve the remaining mesh entities classified on curved boundaries to the required order of approximation. Since curving entities can lead to mesh invalidities, this step includes the application of curved mesh modification operations, including entity shape change, splits, collapses, and swaps, as needed to ensure that a valid mesh of acceptably shaped curved elements is provided.

5.1 Curve the singular feature isolation mesh

To ensure a properly curved graded mesh around the singular features, the process simultaneously curves the mesh entities on the boundary and those within the structured layer above it. Figure 8 demonstrates this process with Fig. 8a showing the linear structured layer mesh associated with a portion of a curved model edge. Figure 8b shows how the local gradation would be disrupted if only the mesh edge, $M_0{}^1\mathrm{G}_0{}^1$were curved. To avoid this problem, the ''parallel'' and ''diagonal'' mesh edges (see Fig. 8a) are also curved such that the entire layer maintains its gradation and element shapes are smooth. This process is accomplished by moving the control points of the parallel and diagonal edges based on how far the control points of the edge on the model boundary are moved with consideration given to the change in length of the edges being curved. The need to account for the length of the edges being curved is demonstrated in Fig. 8c, where all control points were moved the same distance as the associated control points on the mesh edge classified on the curved boundary $M_0{}^1$. It is clear that the short edges are curved more than desired. The method used scales the shape change of the base mesh edge accounting for edge lengths as follows. Let L_{base}, L_{p_i}, and L_{d_i} be the straight-sided edge length of the mesh edge on the curved boundary (the base edge), the ith parallel, and the ith diagonal edge, respectively. The scale factors S_{p_i} and S_{d_i} for the ith parallel and diagonal edges applied to their movement with respect to that of the base edge are:

$$S_{\text{p}_i} = \frac{L_{\text{p}_i}}{L_{\text{base}}}, \quad S_{\text{d}_i} = 0.5\left(S_{\text{p}_{i-1}} + S_{\text{p}_i}\right) \tag{4}$$

Figure 8d shows a smooth curved isolation mesh by using this method.

5.2 Curve mesh entities on the curved boundary

After curving the graded mesh around the isolated mesh edges, the curving procedure curves the remaining mesh edges/faces classified on curved model boundaries. The

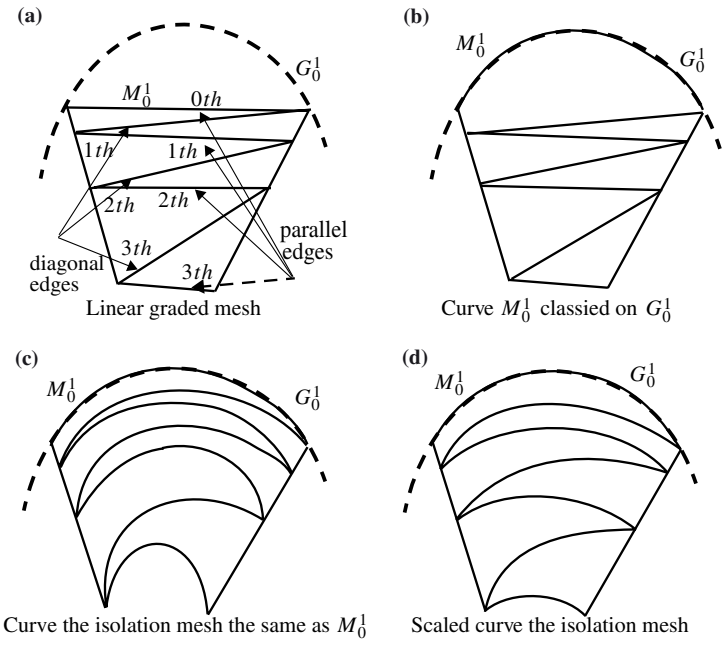


Fig. 8a–d Curve the singular feature isolation mesh. **a** Linear graded mesh. **b** Curve $M_0{}^1$ classified on $G_0{}^1$. **c** Curve the isolation mesh the same as $M_0{}^1$. **d** Scaled curve the isolation mesh

entities are put into a list with the attachment of a proposed Bezier geometric shape to curve them to the boundary.

The process of curving the mesh entities traverses the list. If the movement associated with this curving maintains validity of the connected mesh entities, the entity is curved and removed from the list. If any invalidities arise, additional processing is required. Since changing the shape of a mesh entity changes the shape of the mesh entities it bounds, the process of checking the validity of the mesh must check the shape validity of the connected mesh entities as well. This means that, in addition to verifying no self-intersection of the curved mesh face or edge, the shape validity of the connected high dimensional mesh entities are checked using the shape check algorithm outlined in Sect. 2.

In those cases when a mesh entity does cause an invalidity, local mesh modification operations are applied as indicated in Sect. 5.3. After the needed mesh modifications are performed, the current mesh entity is removed from the list and any new mesh entities created that are classified on curved boundaries are added to the list. Shape quality of curved element is one important open topic in p-version mesh generation because it can not only choose the preferable local mesh modifications when multiple choices are possible, but also guide the mesh to become the most suitable one to produce optimal results in analysis procedure. However, some of the currently existing quality measures, such as interior determinant of Jacobian variation [5], are proposed only based on element geometric shape without any direct relation to the solution accuracy. Under the assumption that a future adaptive procedure will consider altering element interior shape to optimize accuracy, the current procedure presented is focused only on ensuring a positive Jacobian throughout the element domain.

Figure 9 shows a simple example of curving the mesh edges on model edge $G_0{}^1\mathrm{M}_0{}^1$ and $M_1{}^1$ are curved to the boundary without causing any mesh invalidities. However, face $M_2{}^0$ becomes invalid when curving $M_2{}^1$ $\mathrm{M}_0{}^0\,\mathrm{M}_3{}^1$would produce two elements with small angles. Therefore, swapping $M_3{}^1$ provides a better opportunity for maintaining larger angles. However, the swapped edge must be curved as shown in Fig. 9c.

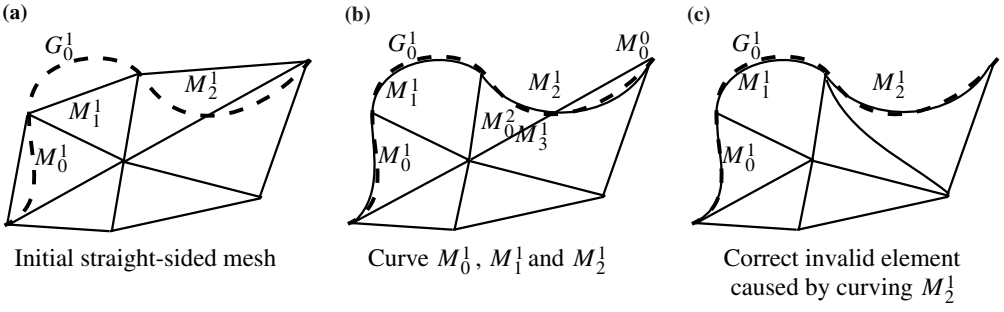


Fig. 9a–c Curving the mesh edges classified on $G_0{}^1$. **a** Initial straight-sided mesh. **b** Curve $M_0{}^1$, $M_1{}^1$, and $M_2{}^1$. **c** Correct invalid element caused by curving $M_2{}^1$

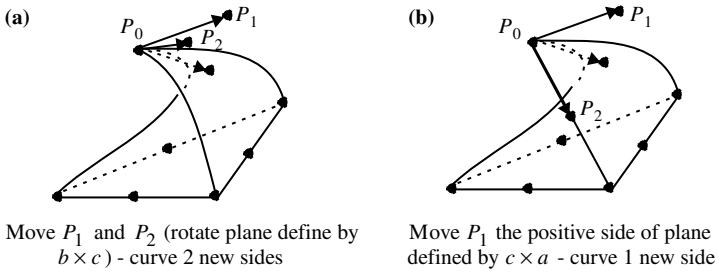


Fig. 10a, b Correct invalid region by shape manipulating. **a** Move P_1 and P_2 (rotate plane defined by $b \times c$)—curve 2 new sides. **b** Move P_1, the positive side of plane defined by $c \times a$—curve 1 new side

5.3 Curved mesh modification operations to correct invalid element

The curved local mesh modifications processes build upon a set of operations that include mesh entity shape modification, split, collapse, and swap.

Mesh entity shape modification is often successful when curving a mesh entity on the boundary has caused an intersection with another mesh entity and there is adequate room on the "other side" of the intersected mesh entity to re-shape it such that the intersection is eliminated. The properties of the Bezier control polygon are used in this process to determine if re-shaping will eliminate the intersection and make the element valid again. For example, one can examine the relationship of the three partial derivative vectors at a mesh vertex where the Jacobian det(J) is negative due to the intersections of bounding mesh entities. Figure 10 presents two possible solutions to correct the invalid region given in Fig. 1. An example where valid mesh is regained by only applying interior mesh entity shape manipulation is shown in Fig. 11.

In those cases where two neighboring mesh entities of a single element are on a curved boundary, the curving process will create angles of 180°. The only option in this case is to execute a split operation so that a new mesh entity not classified on that same boundary entity is introduced between those two entities. Figure 12 gives an example in which when the initial linear mesh in Fig. 12a is curved, there are 180° angles at the four mesh vertices with circles around them in Fig. 12b, as well as along the two edges connecting the two pairs of them. In this case, introducing edge splits of the edges on the top and bottom of the cylinder will eliminate the problem shown in Fig. 12c. The process of splitting takes advantage of the property that the Bezier geometries maintain their shapes under a subdivision process. For example, a curved face and region split are shown in

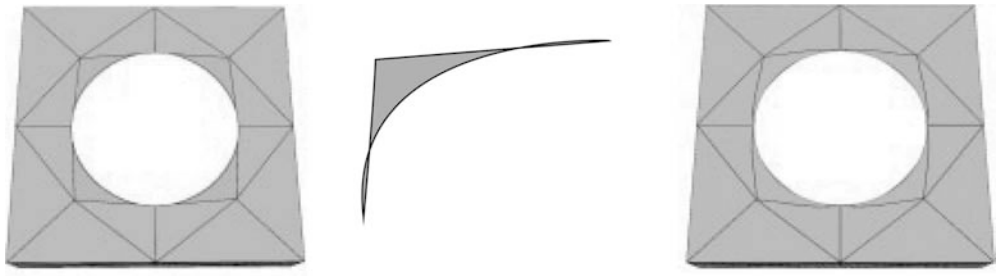


Fig. 11a–c Curve mesh corrected by shape improvement. **a** Invalid mesh. **b** Closeup of the invalidity. **c** Valid mesh

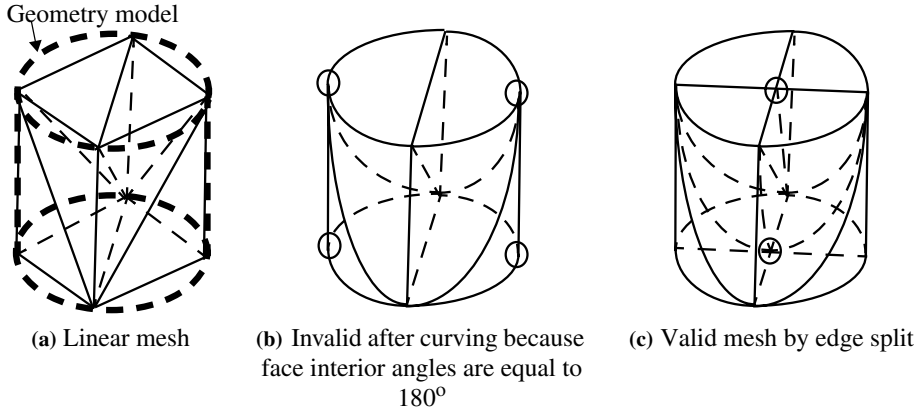


Fig. 12a–c Splitting to correct 180° angles. **a** Linear mesh. **b** Invalid after curving because face interior angles are equal to 180°. **c** Valid mesh by edge split

Fig. 13 where the newly created curved regions exactly decompose the shape of the parent regions.

There are cases where applying only curving and splitting 180° angles will not produce a valid or satisfactory mesh result in terms of element shape. This typically occurs where the current mesh configuration is such that there is insufficient room in the desired direction of motion to re-shape mesh entities. The goal of curved mesh entity collapse and swap operators is to change the mesh topology to produce sufficient space. Since these operators are computationally more demanding than the others, they are only considered when they are needed. Figure 14 shows a case where the invalidity caused by curving M_0^1 to G_0^1 is effectively corrected by collapsing edge M_1^1 (see Fig. 14c). The collapse produces a better mesh configuration maintaining better angles than just curving edge M_2^1

The primary new complexity in collapse and swap operations introduced by curved mesh geometry is to determine appropriate geometric shapes for the new mesh entities created during those operations. Figure 15a shows the curved domain defined by the five regions connected to vertex M_8^0 that will be deleted when collapsing edge M_1^0 by moving M_8^0 to $M_7^0 M_1^1$ and M_2^1 shown in Fig. 15b, as well as their higher order-connected entities, need to have proper curved shapes assigned to them to make the operation valid. Compatibility between a geometric shape and the finite element basis indicates that the geometric order of these mesh entities should be equal to or less than that used for finite element to satisfy the standard finite element completeness requirement. Thus, the geometric order of a new edge or face can only elevate up to the allowable order. Within the limits of the order of geometry allowed, it is important to determine an appropriate shape for the new mesh entities. For example, if the edge shape of M_2^1 was constructed as straight-sided instead of curved, it would intersect the boundary of the affected region, which indicates a mesh invalidity. To avoid this problem, blending operators [11] are applied to compute the shape for new edges inside of curved quadrilateral polygons without interfering with others. For example, the shape of edge M_1^1 is determined by the curved polygon bounded by M_2^0, M_3^0 and M_8^0, M_7^0 and the shape of edge M_2^1 determined by the curved polygon bounded by M_4^0, M_6^0, M_7^0, and M_8^0 (Fig. 15b). The general procedure to compute a curved edge shape inside a quadrilateral polygon is as follows:

Let $(\phi_0, \phi_1, \phi_2, \phi_3)$ be the shapes of the curved quadrilateral polygon and (P_0, P_1, P_2, P_3) are the locations of corresponding vertices M_0^0, M_1^0, M_2^0, and M_3^0. The blend mapping inside the polygon is:

$$x = (1-v)\phi_0 + u\phi_1 + v\phi_2 + (1-u)\phi_3 - (1-u)(1-v)P_0 - (1-v)uP_1 - uvP_2 - (1-u)vP_3 \quad (5)$$

If the new edge from M_0^0 to M_2^0 is quadratic, substitute $u = v = 0.5$ into Eq. 5 to compute the control point location:

$$x = \frac{1}{2}(\phi_0 + \phi_1 + \phi_2 + \phi_3) - \frac{1}{4}(P_0 + P_1 + P_2 + P_3) \quad (6)$$

If the edge is cubic, $u = v = 1/3$ and $u = v = 2/3$ are used for the two control points, respectively (Fig. 16). Since simple blend operations do not ensure the generated entities will be contained in the bounding entities, the

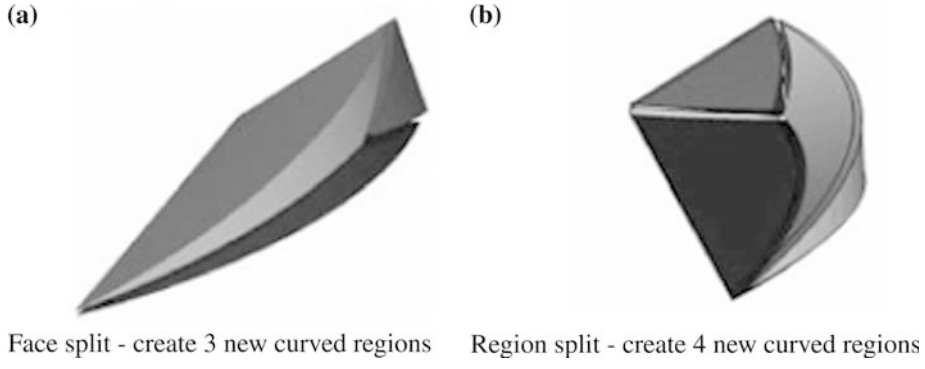


Fig. 13a, b Curved face and region split operations. **a** Face split—create three new curved regions. **b** Region split—create four new curved regions

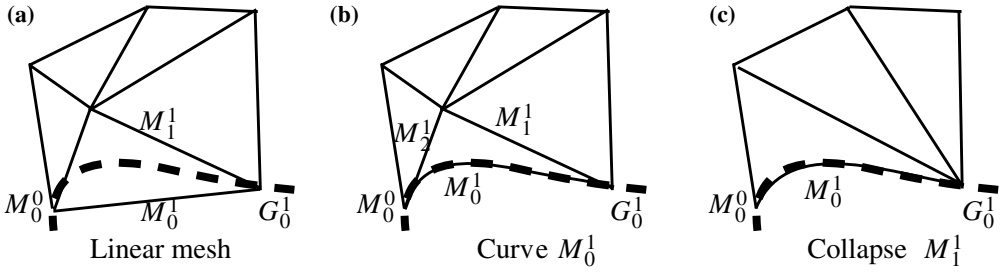


Fig. 14a–c Edge collapse to fix invalid element. **a** Linear mesh. **b** Curve M_0^1. **c** Collapse M_1^1

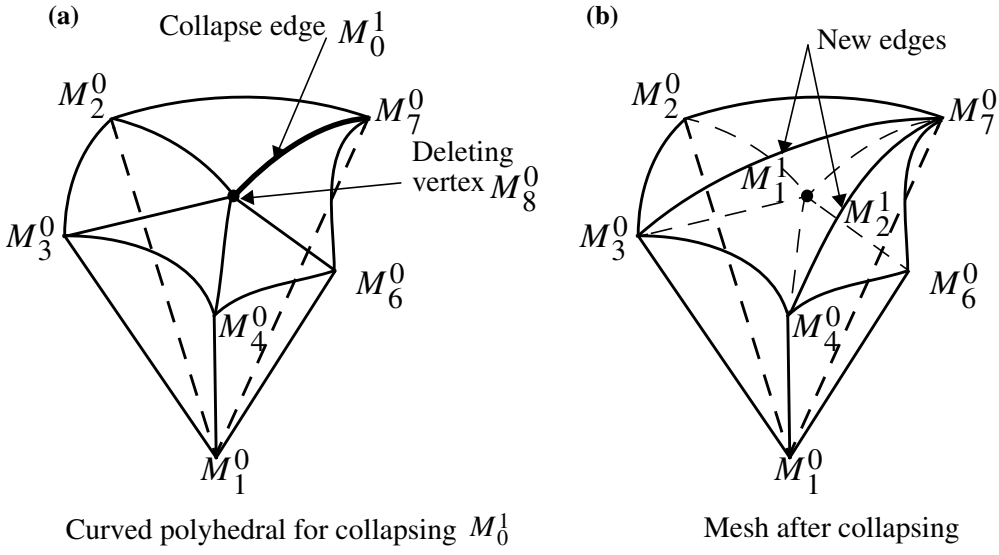


Fig. 15 a, b Curved edge collapsing. **a** Curved polyhedral for collapsing $M_0{}^1$. **b** Mesh after collapsing

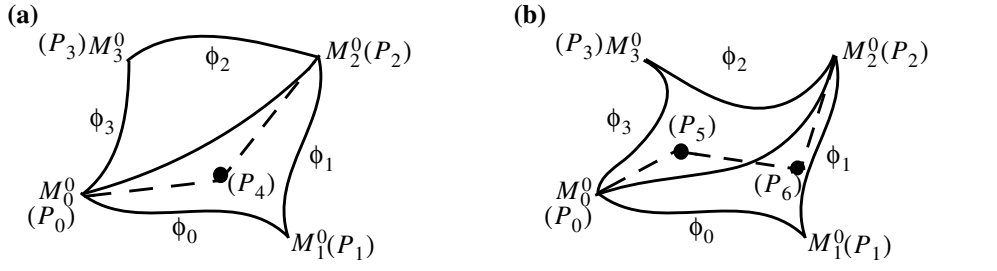


Fig. 16 a, b Blending method to compute geometric shape. **a** $u = v = 0.5$ for P_4 of quadratic shape. **b** $u = v = 1/3$ for P_5 and $u = v = 2/3$ for P_6 of cubic shape

validity checks must still be performed and locations adjusted if needed.

The incremental procedure being developed to select the appropriate local mesh modifications to correct invalid elements builds on the linear geometry algorithm of [12] with appropriate extensions to account for the added complexity of curved mesh entities. Based on the computational cost and complexity of each operation, the process works in the following five steps:

S1. Determine if the invalidity is caused by pairs of neighboring mesh faces or edges classified on the boundary such that angles of 180° or greater are created. In these cases, apply a split operation that will introduce additional mesh entities to subdivide those angles.

S2. Check whether there is adequate room on the "other side" of the intersected mesh entity and that its geometric order is sufficient to re-shape. If these conditions are met, apply mesh entity re-shape to fix the problem.

S3. Examine the possibility of applying a collapse operation to eliminate the invalidity. Apply the collapse if possible.

S4. Examine the application of a swap operation to create the space required to have appropriately shaped mesh entities in the swapped configuration to eliminate the invalidity.

S5. If none of the above processes can be applied, apply local refinement and place all new mesh entities classified on the boundary into the list of entities to be processed. Subdivision creates more options for applying other operations.

Although the last operation (S5) does typically allow completion of the process, it does introduce undesired mesh refinement (see Fig. 17c in the next section). In such cases, an alternative is to inflate the order of the geometry of the mesh entity to provide additional shape freedom without the need to add additional elements. As shown in Fig. 17d, this approach can be successful in obtaining a valid mesh without the need for refining the mesh. Of course, inflating the geometry order can force the need to inflate the order of finite element used at that location past what may have been required for analysis accuracy. However, this is likely to be more cost-effective than the additional cost introduced by the refinement process. Additional investigation of these possibilities is underway.

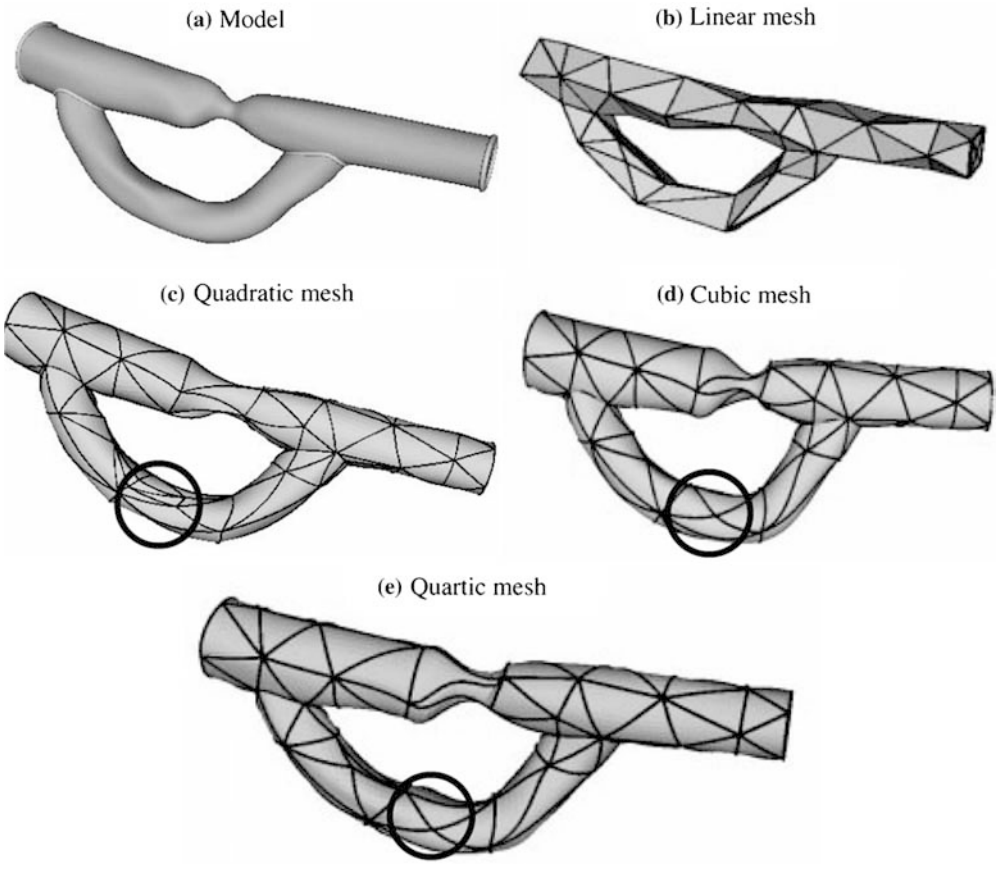


Fig. 17 a–e Blood vessel meshes. **a** Model. **b** Linear mesh. **c** Quadratic mesh. **d** Cubic mesh. **e** Quartic mesh

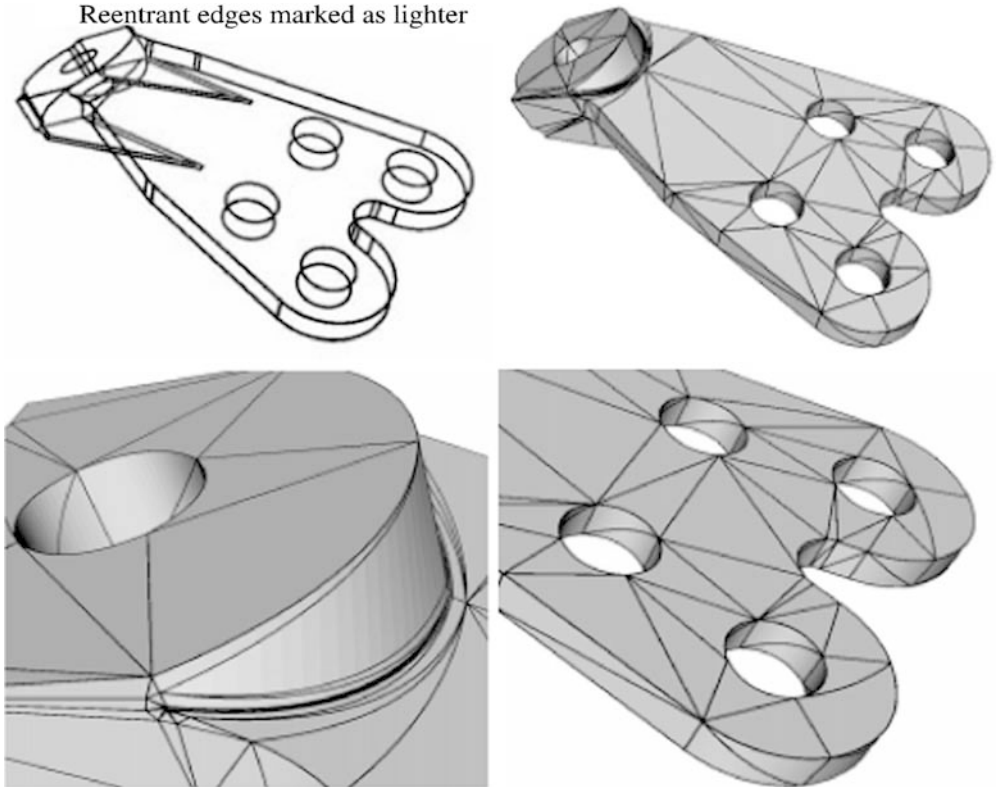


Fig. 18 p-version mesh for model 1

Table 1 Statistics for curving the blood vessel meshes

	Blood vessel			
	Linear	Quadratic	Cubic	Quartic
Δd_{max}	0.1208	0.0702	0.0355	0.0329
Δd_{min}	1.7028	0.9897	0.5006	0.4648
Collapse		2	1	1
Swap		3	1	1
Split		17	5	5
Recurving		29	15	14
Refinement		10	3	2

Table 2 Statistics for curving meshes for models 1 and 2

	Quadratic	
	Model 1	Model 2
Reentrant model edges	9	64
Collapse	0	7
Swap	0	4
Split	1	10
Recurving	12	57
Refinement	0	4

6 Example results

Curvilinear mesh examples are presented in this section. The first example is a biomechanical model used to simulate blood flow. Linear, quadratic, cubic, and quartic meshes are shown in Fig. 17. The quality of the curved mesh with respect to the geometry model is measured by Δd_{max} and Δd_{min}, which are the normalized maximum distance deviation Δd to the longest and shortest linear mesh edge length, respectively. Δd is computed as the maximum distance between sample points of each mesh entity classified on the curved model boundary and their corresponding closest points on the boundary. Statistics for curving the blood vessel meshes are presented in Table 1. The results clearly demonstrate that geometric approximation error in terms of normalized maximum distance deviation has been improved by increasing the geometric

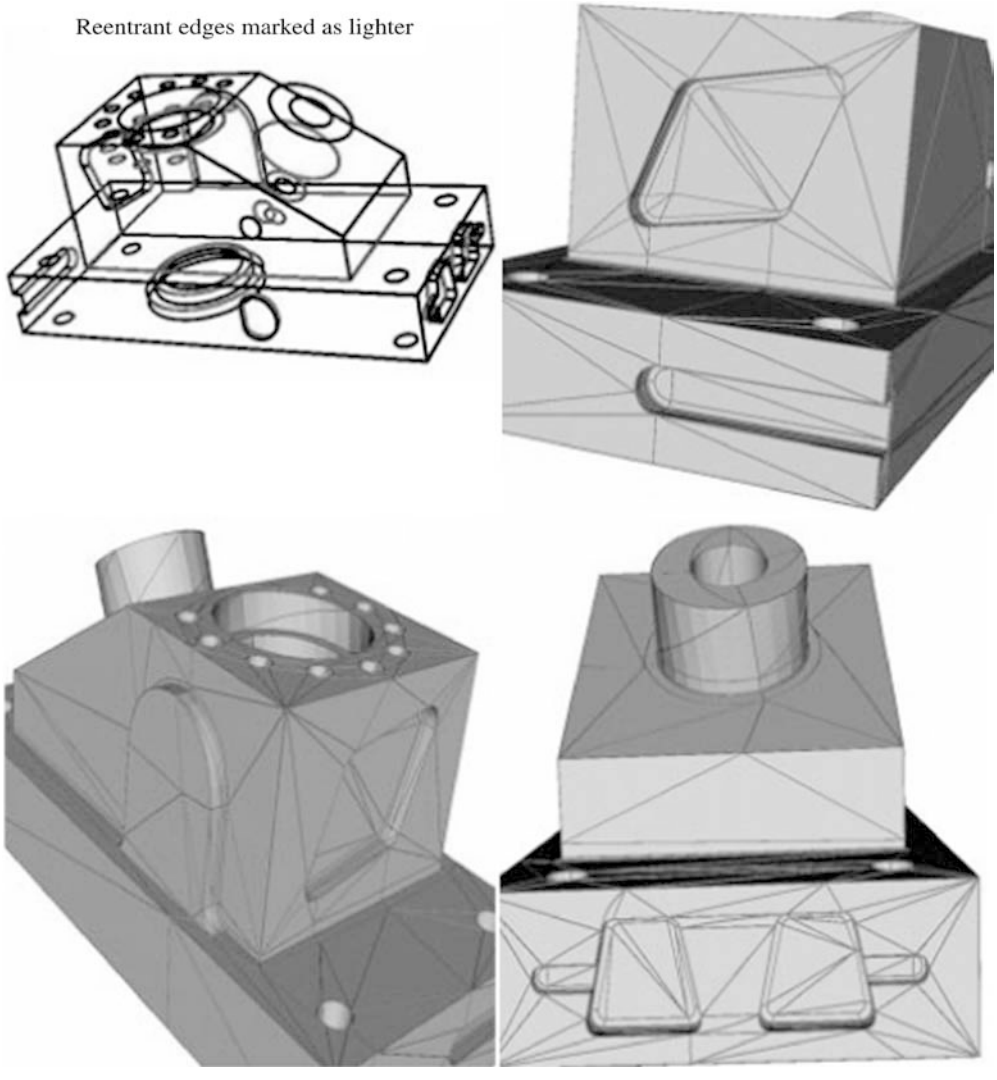


Fig. 19 p-version mesh for model 2

approximation order, especially in going from linear to cubic geometry.

Table 1 also includes the statistics on which local mesh modifications were used to correct the invalid elements. The shape manipulation operations are the most frequently applied in this example. The number of refinements required has been reduced from ten for quadratic shaped elements to three for cubic shaped elements and two for quartic shaped elements, thus, demonstrating the benefit of using high geometric order to alleviate the need for local refinement to fix invalidity. The portion of the domain requiring refinement is circled on the curved meshes shown in Fig. 17c–e.

The next two examples (Figs. 18 and 19) are mechanics components for which curved graded quadratic meshes are generated by using the procedures discussed above beginning from isolating all of the reentrant model edges. Statistics for the examples are presented in Table 2. Recurving remains the dominated operations to fix invalid elements. Figures 18 and 19 include close-ups of curved graded meshes around selected singular edges.

7 Conclusion

This paper discussed an algorithm to automatically generate curved graded meshes for general three-dimensional domains suitable for solving elliptic partial differential equation problems by the p-version finite element method. The procedure starts with the automatic isolation of singular reentrant model entities. Linear geometry elements are generated around those features with gradations as needed for optimal hp-convergence. These elements are then curved while maintaining the appropriate gradation. The procedure next constructs a linear geometry mesh for the rest of the domain. The linear mesh entities classified on curved boundaries are then curved to those boundaries. Whenever those curving operations introduce invalid elements, generalized curved mesh modification operations are applied to eliminate the problems.

As the results demonstrate, the procedure as currently developed has the capability to produce p-version meshes on general three-dimensional domains. Additional improvements to construct more optimal mesh configurations and the development of complete hp-adaptive procedures building on these meshing procedures are underway.

Acknowledgements This work was supported by the National Science Foundation through SBIR grant number DMI-0132742.

References

1. Anderson B, Falk U, Babuska I, Petersdorff TV (1995) Reliable stress and fracture mechanics analysis of complex components using a hp version of FEM. Int J Numer Methods Eng 38:2135–2163
2. Babuska I, Suri M (1994) The p and h-p versions of the finite element method, basic principles and properties. SIAM Rev 36(4):578–632
3. Babuska I, Petersdorff TV, Anderson B (1994) Numerical treatment of vertex singularities and intensity factors for mixed boundary value problems for the Laplace equation. SIAM J Numer Anal 31(5):1265–1288
4. Babuska I, Anderson B, Guo B, Melenk JM, Oh HS (1996) Finite element method for solving problems with singular solutions. J Comp Appl Math 74:51–70
5. Dey S, O'BaraRM, Shephard MS (2001) Towards curvilinear meshing in 3D: the case of quadratic simplices. CAD Comput Aided Des 33(3):199–209
6. Dey S, Shephard MS, Flaherty JE (1997) Geometry representation issues associated with p-version finite element computations. Comput Methods Appl Mech Eng 150(1–4):39–55
7. Dorr MR (1986) The approximation of solutions of elliptic boundary-value problems via the p-version of the finite element method. SIAM J Numer Anal 23(1):58–77
8. Farin GE (1993) Curves and surfaces for computer aided geometric design: a practical guide, 3rd edn. Academic Press, Boston
9. Farouki RT, Rajan VT (1988) Algorithms for polynomials in Bernstein. Comput Aided Geom Des 5:1–26
10. Garimella R, Shephard MS (2000) Boundary layer mesh generation for viscous flow simulations in complex geometric domains. Int J Numer Methods Eng 49(1–2):193–218
11. Harber R, Shephard MS, Abel JF, Gallagher RH, Greenberg DP (1981) A general two-dimensional, graphical finite element preprocessor utilizing discrete transfinite mappings. Int J Numer Methods Eng 17:1015–1044
12. Li X, Shephard MS, Beall MW (2003) 3-D anisotropic mesh adaptation by mesh modifications. Comput Methods Appl Mech Eng (submitted)
13. Luo XJ, Shephard MS, Remacle JF, O'Bara RM, Beall MW, Szabo BA, Actis R (2002) p-version mesh generation issues. In: Proceedings of the 11th international meshing roundtable, Ithaca, New York, September 2002. Sandia National Laboratories, pp 343–354
14. Sherwin SJ, Peiro J (2002) Mesh generation in curvilinear domains using high order elements. Int J Numer Methods Eng 53:207–223
15. Szabo BA (1986) Mesh design for the p-version of the finite element method. Comput Methods Appl Mech Eng 55:181–197
16. Szabo BA, Babuska I (1991) Finite element analysis. Wiley, New York