Available online at www.sciencedirect.com

SciVerse ScienceDirect

Procedia Engineering 00 (2015) 000–000

Procedia Engineering

www.elsevier.com/locate/procedia

24th International Meshing Roundtable (IMR24)

Optimizing Corner Assignment of Submap Surfaces

Shengyong Cai[a], Timothy J. Tautges[b]

[a]*CD-adpaco, 10800 Pecan Blvd, Suite 210, Austin, Tx, 78750, U.S.A.*
[b]*CD-adapco, 1500 Engineering Dr, Madison, WI, 53706, U.S.A.*

Abstract

Submapping is a meshing tool for generating structured meshes on non-4-sided surfaces and non-6-sided volumes. An important part of submapping is the assignment of corners in the map to vertices bounding surfaces or edges bounding volumes. In models with fillet, chamfer and round features, corner assignment is difficult, since these features remove discontinuous changes in edge/surface normal, replacing them with features having smooth changes in normal direction. Also required for submapped surfaces and volumes is that corner or edge types bounding a surface or volume sum to four or six, respectively. This paper combines the use of templates to identify surface submapped corners for a variety of common feature types, and the use of an optimization approach to guarantee satisfaction of the $sum = 4 - 4g$ constraint for submapped surfaces (g is the number of holes). This approach not only works well in the presence of features, but can drastically reduce the user interaction required to set up and mesh models with many features.

Peer-review under responsibility of organizing committee of the 24th International Meshing Roundtable (IMR24).

Keywords: Vertex types, submapping, templates, structured mesh, corner assignment;

1. Introduction

In the industrial and academic applications, the increasing complexity of finite element analysis of structures, fluid flow and interaction problems requires high mesh quality. A large amount of boundary-flow and structural problems require meshes to be boundary structured. Generally, it is almost impossible to generate structured meshes with high quality on real, complex geometry. Hence, manual processing of CAD geometry into well-structured zone is used, which is tedious and time consuming.

Structured meshes play an important role in quad and hex meshing, not only because they align well with model features, but also because by definition the 2d boundary of any swept hexahedral mesh must be a structured quadrilateral mesh. In a typical BREP-based mesh generation process, geometric entities of lower dimension are meshed first, proceeding to higher dimensions with pre-defined boundary mesh. For models meshed using the sweeping algorithm, where mesh faces are extruded into a 3rd dimension, the surfaces on the "side" of the swept region control which

* Corresponding author. Tel.: +1-608-698-5398.
E-mail address: shengyongcai@gmail.com

Peer-review under responsibility of organizing committee of the 24th International Meshing Roundtable (IMR24).

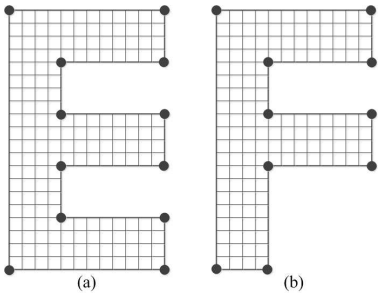

Fig. 1. Structured quadrilateral mesh for letter "E" and "F" surface

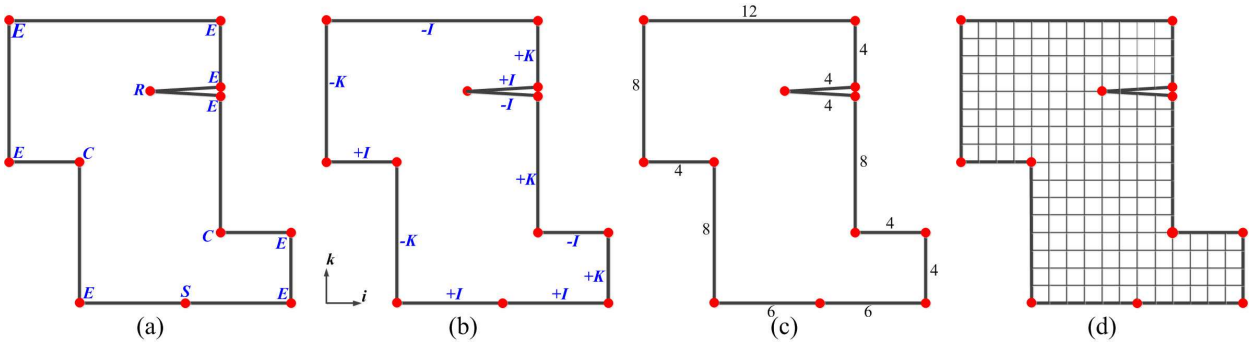

Fig. 2. The work flow of submapping algorithm: (a) corner assignment(or vertex classification) where $E \rightarrow END$, $S \rightarrow SIDE$, $C \rightarrow CORNER$ and $R \rightarrow REVERSAL$; (b) edge parameterization; (c) edge discretization; (d) interior node interpolation for structured quad mesh

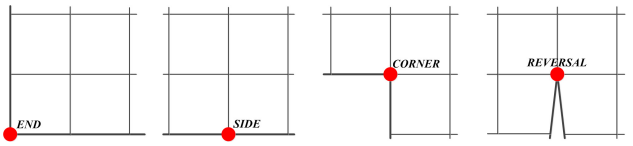

Fig. 3. Four vertex types for submapping method

direction the mesh is swept. Thus, successful application of the sweeping algorithm depends on proper definition of the structured meshes on the sides of the swept region. For complex CAD models of industrial parts, having blends, fillets, and other design features, locating the corners of the map on these surfaces can be challenging. Traditional approaches to this problem depend on angles being close to integer multiples of $\pi/2$, however, design features typically smooth out these abrupt changes on surfaces, usually to avoid stress concentrations or other singularities. In this work, we combine template-based identification with an optimization-based method to improve identification of corners for structured mesh surfaces.

Structured mesh regions can have four "sides", separated from each other by so-called "corners", and parameterized in an integer (i, k) space; such meshes are referred to as "mapped" meshes. However, one can also define structured meshes with more than four sides; these are referred to as "submap" surfaces, afterwards the submapping algorithm is used to generate structured quad mesh, first described in Ref. [3]. Examples of submapped meshes are shown in Fig. 1 where any interior node is shared by exactly four quad elements. Submapping consists of four main steps shown in Fig. 2, that is, vertex classification or corner assignment (Fig. 2(a)), edge parameterization (Fig. 2(b)), edge discretization (Fig. 2(c)) and interior node interpolation (Fig. 2(b)). First, the corner assigned on each vertex should be done so that structured meshes can be embedded into geometries with high mesh quality. There are four vertex types, that is, *END*, *SIDE*, *CORNER* and *REVERSAL*. They are shared by one, two, three and four quad elements, respectively (see Fig. 3). Then edges are parameterized in the local 2D space along i and k. Those geometric edges can be classified as $+i$, $-i$, $+k$ or $-k$. In order to make resulting meshes structured, interval assignment for boundary edges must be resolved [13], namely, the number of divisions on all the $+i$ and $+k$ edges should be equal to that of

all the $-i$ and $-k$ edges, respectively. After boundary mesh discretization, interior nodes are interpolated based on transfinite interpolation with those discretized boundary mesh nodes.

"Map" surface/volume meshes can be parameterized as a 2D/3D space of logical (i, j, k) parameters, with various methods used to derive spatial embedding. Map meshes of d dimensions always have $2d$ logical sides (4/6 for surface/volume meshes, respectively). Traversing around the boundary of a map mesh, the points where a transition is made from one side to another are referred to as "corners" in the map; corners also determine the number of quadrilaterals/hexahedra will share the vertex/edge at the corner. In contrast to map meshes, submap meshes can have more than $2d$ logical sides. Both map and submap meshes are *structured* quad/hex meshes, that is, each interior node is connected to $2d$ other nodes and 2^d d-dimensional elements. For example, consider faces shaped like the block letters "E" and "F" shown in Fig. 1. These faces can be given structured meshes with more than four sides (12 for "E" and 10 for "F"). A corner of a submap mesh is any point/vertex adjacent to two different logical sides (this concept is elaborated more in the following section). Map meshes can be considered as a special case of submap meshes; subsequent discussions using the term "submap" are understood to apply equally (sometimes trivially) to map meshes as well.

Given a geometric face, e.g. from a BREP model, the steps described in Fig. 3 are required to generate submap meshes. The primary focus of this work is the first step, identification of corners for the submap. In practice, this task is not as straightforward as it is for our "E" and "F" examples described above. First, while it is obvious how many quadrilaterals should be connected to a vertex between sides meeting at 90, 180, or 270 degrees, it is less clear what to do for other angles. Furthermore, there can be multiple valid ways to resolve a vertex at a given angle; for example, 2, 3, and even 4 quadrilaterals can be connected at a 180-degree vertex, at least from the perspective of FEA [1]. Previous work has shown that vertex types bounding a geometric face must meet a $sum = 4 - 4 * g$ constraint[3,4], but for non-trivial surfaces, there are many different ways to meet that constraint depending on how corners are assigned. Finally, it has also been shown that corner assignment on faces also influences the meshability of volumes using sweeping[1,2]. Therefore, corner assignment affects not only the quality of the resulting mesh, but also the meshability of both the surface(s) and volume(s).

In previous work, Whiteley et al.[3] presented a heuristic method for submapping. First, corner assignment is done purely based on angles. For vertices with so-called "fuzzy angles", i.e., those angles far away from integer multiple of $\pi/2$, the user is required to assign vertex types. The other steps in the submapping process are described as before. The limitation is that the method described by Whiteley et al.[3] requires user effort to assign vertex types at ambiguous angles. More recently, Ruiz-Girones et al.[4] presented an algorithm to improve the vertex classification during *submapping*. In order to overcome drawbacks of the heuristic *submapping* algorithm, they used optimization to improve the initial classification of surface vertices in such a way that necessary conditions were satisfied. They proposed an objective function that minimized differences between new vertex classification and heuristic angle-based vertex classification. However, they also constrained vertex type changes to no more than one in either direction. In practice, that constraint sometimes prevents convergence of the optimization problem to a feasible solution (this example is described later in this paper (Fig. 9)). The solution described in this paper uses a similar optimization-based approach as was used by Ruiz et. al, but with the addition of templates to handle features commonly found in real-world geometric models.

This paper is structured as follows: in Section 2 provides a statement of the problem of assigning corners for the submapping problem. Our algorithm for optimal corner assignment is described in Section 3, and includes templates in Section 3.1 and vertex type adjustment based on optimization in Section 3.2. Section 4 describes edge discretization and interior node interpolation following corner determination. Finally, examples are shown in Section 5 and conclusions are made in Section 6.

[1] While the accuracy of the resulting analysis may be slightly degraded for a face with a 180/4=45 degree angle compared to a 90 degree angle, it typically won't be be much.

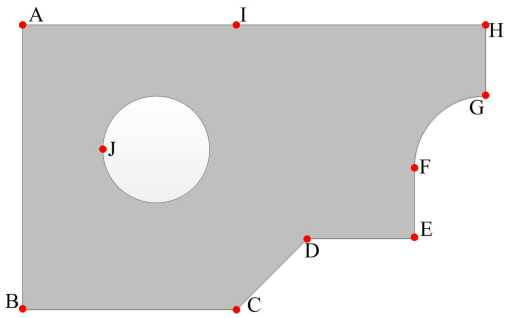

Fig. 4. A problematic example of corner assignment by angle-based approach

2. Problem Statement

The surface vertex type for submapping is defined as the classification of the topology of a vertex bounding a structured all-quadrilateral surface mesh by the number of quadrilaterals on the surface that share that vertex. Let θ_i be an internal angle between two adjacent geometric edges at a vertex i bounding a surface. The traditional vertex type $\overline{\alpha_i}$ is defined as

$$\overline{\alpha_i} = \lfloor 2(1 - \frac{\theta_i}{\pi}) \rceil \tag{1}$$

where $\lfloor * \rceil$ is an integer operator which returns the nearest integer to a real value. Then the surface vertex type can be defined as follows:

$$\overline{\alpha_i} = \begin{cases} 1 & \text{for } \textit{END} \text{ vertex} \\ 0 & \text{for } \textit{SIDE} \text{ vertex} \\ -1 & \text{for } \textit{CORNER} \text{ vertex} \\ -2 & \text{for } \textit{REVERSAL} \text{ vertex} \end{cases} \tag{2}$$

For a simply-connected polygon with discrete internal vertex angles θ_i, the sum of $\overline{\alpha_i}$ must be equal to 4.

$$0 * S + 1 * E + (-1) * C + (-2) * R = 4 \tag{3}$$

where S, E, C and R are the number of *SIDE*, *END*, *CORNER* and *REVERSAL* vertices. For a multiply-connected submapped surface with g holes, the sum of vertex types must be $4 - 4 * g$:

$$\sum_{i=1}^{N} \overline{\alpha_i} = 0 * S + 1 * E + (-1) * C + (-2) * R = 4 - 4 * g \tag{4}$$

Equation (4) can be interpreted as constraints on vertex types for a surface to be submapped.

In practice, corner assignment purely based on angles may not result in a surface being submappable, because it fails to satisfy Eqn. (4). First, it is relatively common for angles between two adjacent edges on a face to not be an integer multiple of $\pi/2$, such as the vertex C and D in Fig. 4. In this case, the heuristic classification of vertices may lead to unacceptable results which do not satisfy Eqn. (4). Furthermore, real-world design models often smooth out abrupt angles at edge or face intersections, to eliminate stress concentrations and other physical singularities there. For example, discrete angles at the vertex F and G in Fig. 4 are smoothed out. In these cases, the geometric (continuous) angle appears to be 270 degrees, while the discrete angle appearing in the mesh at that point might be quite different, depending on the characteristic mesh size at that point. There is another point to be noted that angle-based approach for assigning corners suffers from the same problem of spreading discrete angles into continuous angles when there is a geometry with round features such as the vertex J in Fig. 4.

3. Optimal Corner Assignment

In this section, an optimal corner assignment algorithm for *submapping* is demonstrated. In order to deal with those round, fillet and chamfer features on surfaces, they are relaxed and angle-based constraints are changed by

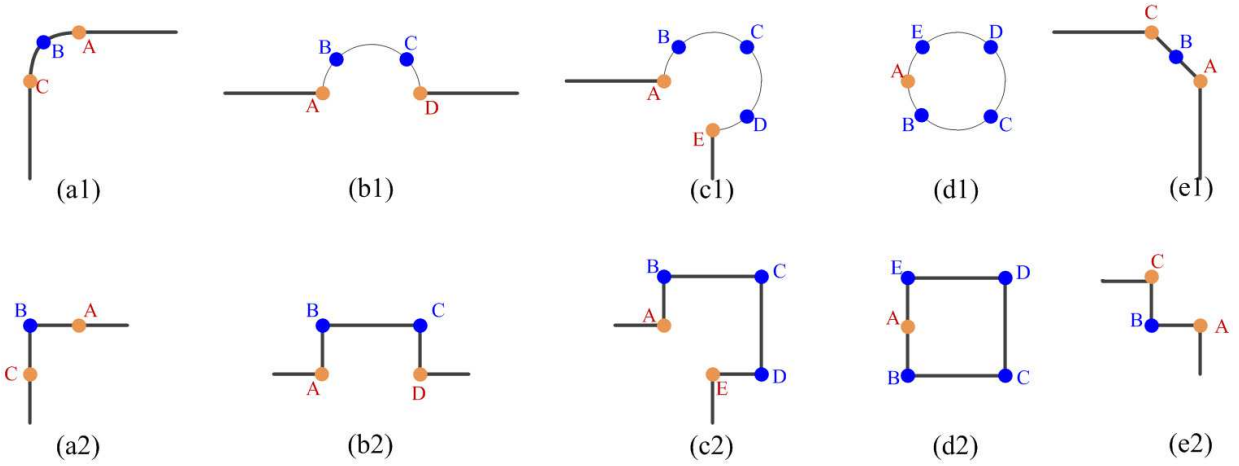

Fig. 5. Templates for classifying vertices during *Submapping*: (a1)geometry feature with a rounding feature(one-quarter circle) between A and C; (b1)geometry feature with a rounding feature(half an circle) between A and D; (c1)geometry feature with a round feature(three quarters of circle) between A and E; (d1)geometry feature with a rounding feature(full circle); (e1)a chamfer feature(a transitional edge) between A and C; (a2)parametric space for (a1); (b2)parametric space for (b1); (c2)parametric space for (c1); (d2)parametric space for (d1); (e2)parametric space for e(1)

using templates described in Sec.3.1. If the angle-based corner assignment is not valid, namely, it fails to satisfy the *submapping* constraint (4), the corner assignment problem is cast as an optimization problem based on the angle-based vertex classification and templates, and solved with a *LP* solver.

3.1. Templates

For many mechanical parts, they have been chamfered and filleted to avoid the stress concentration. However, it is pretty difficult for classifying vertices due to facts that rounds and fillets spread discrete angles into continuous angle changes. For example, in Fig. 14, segments of DE and FG, instead of meeting at a right angle, meet at a fillet EF. Therefore, extra preprocessing is required to deal with these features by relaxing or changing the angle-based constraints used to determine vertex types before generating structured quadrilateral meshes on these surfaces. Meanwhile, virtual vertices are needed to be added on those features where the optimized vertex types can be put. In order to automate these processes and extend capabilities of *Submapping*, some templates are proposed in this paper to get the correct vertex types around these features.

Figure 5 shows the round, fillet and chamfer features on the surface where each top row case can be resolved with corner arrangement in the bottom row. The corresponding structured quadrilateral meshes for those templates shown in Fig. 5 are shown in Fig. 6. If the heuristic method purely based on angles is applied, the invalid corner assignment will be produced since angles are ambiguous and discrete angles are spread into continuous angles. For example, a purely angle-based method would assign a corner type of *SIDE* to all vertices in Fig. 5(d1), and the surface would not be mappable. For the chamfer feature in Fig. 5(e1), the vertex A and C can also be assigned as type-*SIDE* and the middle vertex can be assigned as type-*END*. Detection of those chamfer, fillet, round features can be done through curve type and angles between two tangent vectors which are obtained at the end point of two line segments. Automatically identifying features based on expected configurations will always be limited to “ideal” configurations. In spite of that limitation, the identification of chamfers and especially blends/rounds described in this paper can still save a large amount of user efforts, even if they don’t catch each and every such feature. A possible future enhancement is to separate the identification of these features from their treatment by the algorithm once they are identified, thus allowing users to add manually-identified features to the list. The user effort saved in these manually-identified cases would still be substantial, because of that effort still required to treat those features properly for sweeping.

The templates in Fig. 5 insert virtual vertices on the smooth curve and positions of those virtual vertices have great impacts on the resulting structured meshes. For the fillet and chamfer feature in Fig. 5(a1) and Fig. 5(e1), the virtual vertex B can be simply inserted at the middle position between A and C. For the half-circle feature in Fig. 5(b1), the

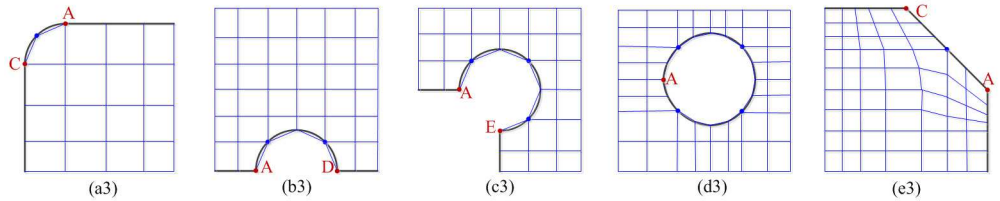

Fig. 6. Structured quadrilateral mesh for templates during *Submapping*: (a3)structured quad mesh for Fig. 5(a1) and Fig. 5(a2); (b3)structured quad mesh for Fig. 5(b1) and Fig. 5(b2); (c3)structured quad mesh for Fig. 5(c1) and Fig. 5(c2); (d3)structured mesh for Fig. 5(d1) and Fig. 5(d2); (e3)structured mesh for Fig. 5(e1) and Fig. 5(e2)

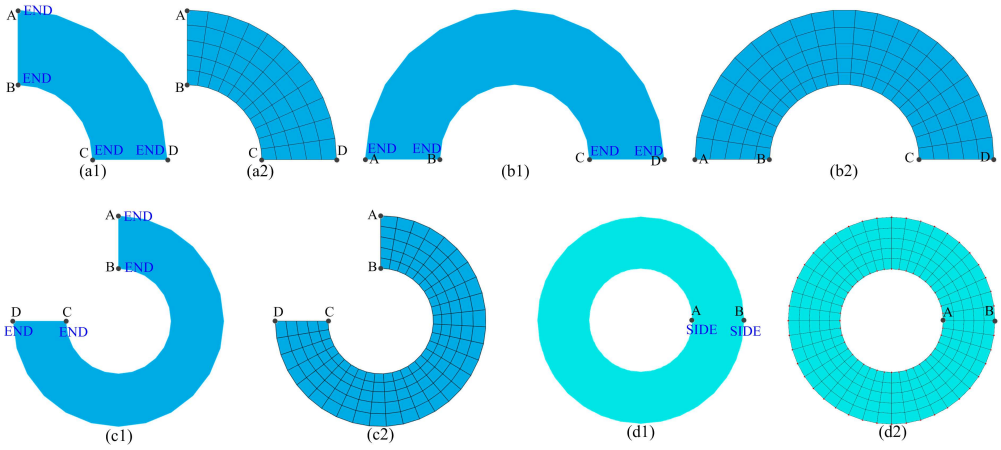

Fig. 7. Templates for concentric rings: (a1)one quarter of concentric ring; (a2)structured mesh for (a1); (b1)one half of concentric ring; (b2)structured mesh for (a2); (c1)three quarters of concentric ring; (c2)structured mesh for (c1); (d1)a full concentric ring; (d2)structured mesh for d(1)

virtual vertices B and C can be inserted at the one-quarter and three-quarter position between A and D, respectively. For the three-quarter circle feature in Fig. 5(c1), the virtual vertices B, C and D are placed at the one-sixth, one-half, five-sixth position between A and E, respectively. The virtual vertices B, C, D and E in Fig. 5(d1) are distributed with equal spacings on a full circle.

For those geometries with concentric ring features such as Fig. 7, the corner assignment is different from templates in Fig. 5: it is not necessary to insert any new vertex and vertex types are assigned directly based on templates. The details are shown in Fig. 7 where Fig. 7(a1), Fig. 7(b1), Fig. 7(c1) and Fig. 7(d1) show the corner assignment for one quarter, one half, three quarters and one full of concentric ring and their corresponding structured quadrilateral meshes are shown in Fig. 7(a2), Fig. 7(b2), Fig. 7(c2) and Fig. 7(d2), respectively.

3.2. Vertex Type Adjustment based on LP

During vertex classification purely based on angles, most vertices can get their correct vertex types. However, there are a few vertices whose vertex types need to be adjusted since internal angles between two consecutive edges are not always integers multiple of 0.5π and Eqn. (4) constrains the corner assignment for *submapping*. Ruiz Girones's method [4] fails to correct the vertex classification in some cases since the variation of vertex types is overly constrained: the maximum variation of vertex types is restricted to one in the Ruiz Girones's method [4]. One fail case by Ruiz Girones's method [4] is shown in Fig. 9: it is impossible to convert vertex type at E from *REVERSAL* to *SIDE*. In order to deal with problems addressed above, a new vertex classification based on *LP* is described in this paper. The key contribution is to generate a valid corner assignment for more geometries based on *LP* by including the *submapping* constraint and constraining the vertex type variation within its geometric angle limits.

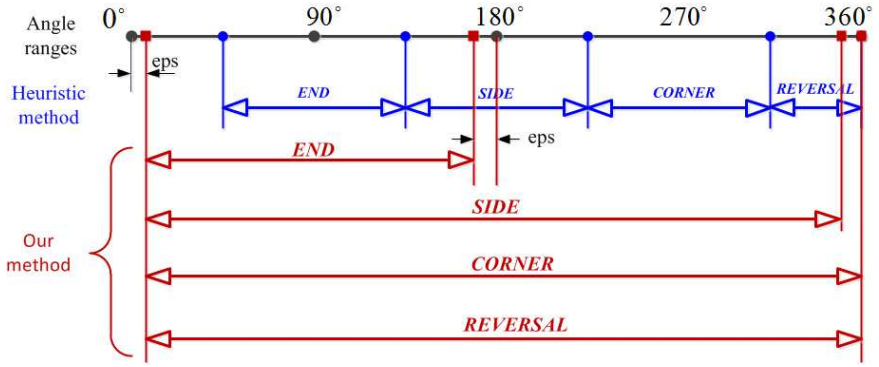

Fig. 8. Ranges of vertex types with respect to the geometric angles

Therefore, the following improved *LP* model is proposed here: the objective is to minimize the maximum deviation between vertex internal angle and the ideal angle associated with assigned optimal vertex types.

$$Objective \qquad min \qquad \max_i\{|\theta_i - (1 - 0.5\alpha_i)\pi|\} \tag{5}$$

where the ideal angle for a vertex with vertex type α_i is $(1 - 0.5\alpha_i)\pi$ and θ_i is a real internal angle at vertex i. Since some *REVERSAL* vertices based on angles may be converted to be *SIDE* due to the *submapping* constraint (4) such as Fig. 9, our approach adjusts vertex types α_i based on the following criteria which are described in Fig. 8:

(1) . If $eps \le \theta_i \le 180^\circ - eps$, $\alpha_i \in \{1, 0, -1, -2\}$.
(2) . If $180^\circ - eps \le \theta_i \le 360^\circ - eps$, $\alpha_i \in \{0, -1, -2\}$.
(3) . If $360^\circ - eps \le \theta_i \le 360^\circ$, $\alpha_i \in \{-1, -2\}$.

In order to add those three criteria into the *LP* model, extra constraints by integrating those criteria are added as follows.

$$\alpha_i < 1 \qquad \text{if } \theta_i \ge 180^\circ - eps \tag{6}$$

$$\alpha_i < 0 \qquad \text{if } \theta_i \ge 360^\circ - eps \tag{7}$$

$$\alpha_i \in \{-2, -1, 0, 1\} \tag{8}$$

In order to make surfaces submappable, sum of vertex types must be equal to $4 - 4 * g$.

$$\sum_{i=1}^{N} \alpha_i = 4 - 4g \qquad \alpha_i \in \{-2, -1, 0, 1\} \qquad \alpha_i \text{ is an integer and } g \text{ is the number of holes} \tag{9}$$

The *LP* model described above contains some nonlinear constraints such as maximal absolute value in the objective function (5) and conditional constraints for Eqn.(6) and Eqn.(7). They can be converted into linear constraints as follows: based on the references[10,11], the absolute value in the objective function can be converted as follows by introducing two positive angle deviation variable: $devAngle^+$ and $devAngle^-$. Meanwhile, a new variable β is introduced to denote maximal differences between internal angle and ideal assigned angle $\max_i\{|\theta_i - (1 - 0.5\alpha_i)\pi|\}$

$$Objective \qquad min\, \beta \tag{10}$$

s.t.

$$\beta >= devAngle_i^+ + devAngle_i^- \tag{11}$$

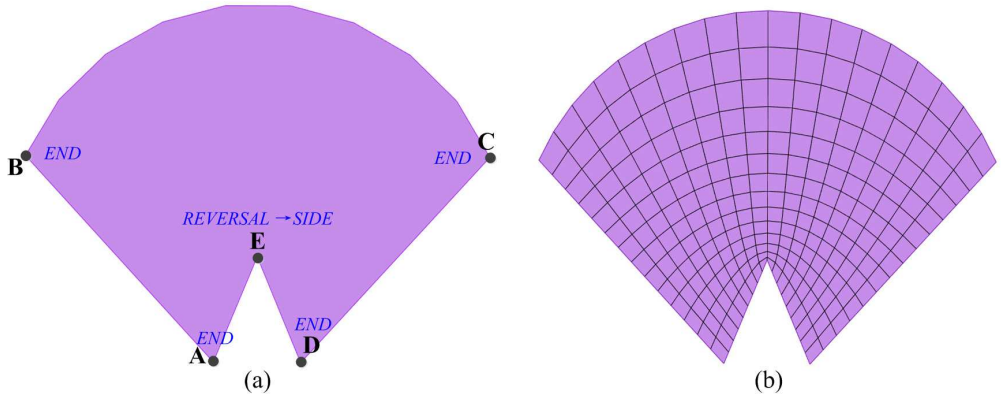

Fig. 9. An example of vertex classification by our method: (a)adjust vertex type at E from REVERSAL to SIDE so that the surface becomes submappable; (b)structured quadrilateral mesh for (a)

$$|\theta_i - (1 - \frac{1}{2}\alpha_i)\pi| = devAngle_i^+ + devAngle_i^- \tag{12}$$

$$devAngle_i^+ - devAngle_i^- = \theta_i - (1 - \frac{1}{2}\alpha_i)\pi \tag{13}$$

$$devAngle_i^+ \geq 0, \qquad devAngle_i^- \geq 0 \tag{14}$$

Based on the reference[12], the conditional constraint (6) and (7) can be converted into the linear constraints by introducing a large enough integer M and binary variables x_i and z_i.

$$-\alpha_i + M * x_i \geq 0 \tag{15}$$
$$180 - eps - \theta_i + M * (1 - x_i) \geq 0 \tag{16}$$
$$360 - eps - \theta_i + M * (1 - z_i) \geq 0 \tag{17}$$
$$1 - \alpha_i + M * z_i \geq 0 \tag{18}$$

Hence, by changing the objective function, keeping the *submapping* constraint and adding extra constraints for vertex type variation, the above *LP* model can be summarized as follows and *LP* model can be solved by a tool: *lpsolve*[9].

$$objective \qquad min\, \beta \tag{19}$$

s.t.

$$\beta >= devAngle_i^+ + devAngle_i^- \tag{20}$$
$$\theta_i - (1 - \frac{1}{2}\alpha_i)\pi = devAngle_i^+ - devAngle_i^- \tag{21}$$
$$devAngle_i^+ \geq 0, \qquad devAngle_i^- \geq 0 \tag{22}$$
$$-\alpha_i + M * x_i \geq 0 \tag{23}$$
$$180^\circ - eps - \theta_i + M * (1 - x_i) \geq 0 \tag{24}$$
$$360^\circ - eps - \theta_i + M * (1 - z_i) \geq 0 \tag{25}$$
$$1 - \alpha_i + M * z_i \geq 0 \tag{26}$$
$$d_i \in \{0, 1\} \quad x_i, z_i \in \{0, 1\} \qquad i = 1, \cdots, N \tag{27}$$
$$M \qquad \text{a large enough integer} \tag{28}$$
$$x_i, z_i \qquad binary\ variables. \tag{29}$$
$$\alpha_i \in \{-2, -1, 0, 1\} \tag{30}$$

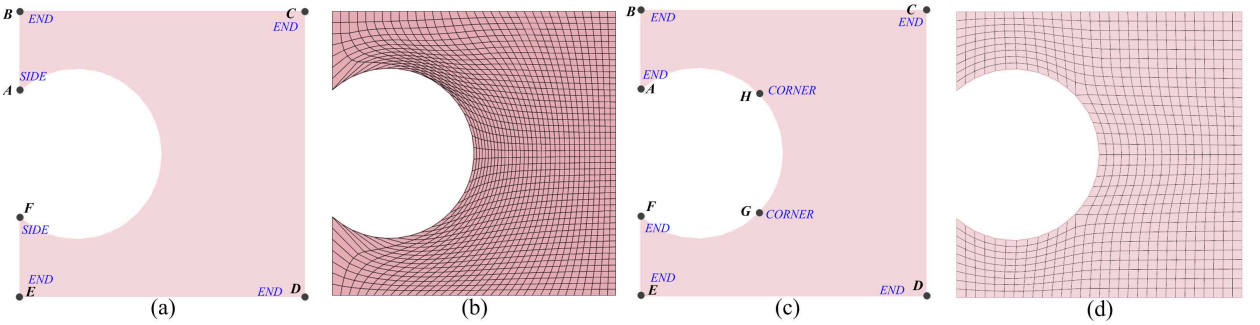

Fig. 10. Structured grid generation of a surface with a three quarters of circle feature by submapping through templates: (a)corner assignment by *Cubit 13.2*; (b)structured grid for (a); (c)corner assignment based on templates in this paper; (d)structured grid for (c)

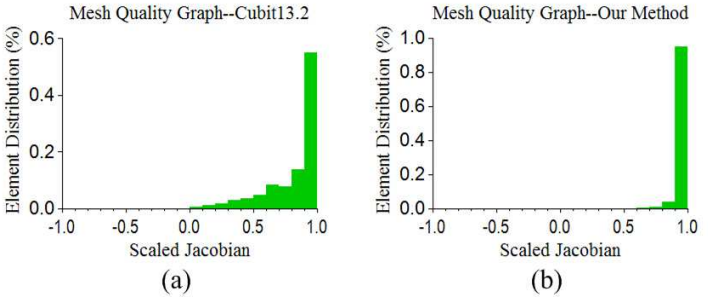

Fig. 11. Mesh quality histogram: (a)mesh quality histogram for Fig. 10(b); (b)mesh quality histogram for Fig. 10(d)

An example is shown in Fig. 9(a) where the vertex *E* is classified as *REVERSAL*. However, the sum of vertex types is equal to 2, which fails to satisfy the submapping constraint. Our corner assignment optimization method can adjust the vertex type at *E* from *REVERSAL* to *SIDE* and resulting structured quadrilateral mesh is shown in Fig. 9(b).

4. Boundary discretization and interior nodes' interpolation

Since surface vertices have already been correctly classified at this point, the edge parameterization, edge discretization [13,14] and interior node's placement can be done as the References[3,4]. Note that generally, interior nodes are embedded in 3D space based on the transfinite interpolation and 3D positions of boundary nodes. For the curved surfaces, the interpolated nodes may not be located on surfaces. Hence, extra geometric processing is needed by projecting the interpolated interior nodes onto surfaces based on the closest points/distances.

5. Examples

In order to assess the mesh quality of structured quadrilateral meshes by submapping with an improved corner assignment algorithm, several examples are provided. Users specify the mesh element size and the algorithm will classify vertices automatically. If there is a multi-connected geometry, it should be virtually decomposed[4] or a path connecting the outmost boundary and interior boundaries should be computed so that the consistent edge parameterization between the outmost boundary and internal holes can be generated. Here, we will skip the details for geometry decomposition in this paper(see reference [4] for details). Note that all the bounding surfaces in all the examples shown below are meshed with structured quadrilateral meshes by submapping.

The first example in Fig. 10 shows a surface with a three-quarters round feature, which makes it difficult to generate a valid corner assignment for submapping. *Cubit 13.2* assigns corners for surface vertices as Fig. 10(a) and the resulting structured quadrilateral mesh is shown in Fig. 10(b). From Fig. 10(b) and Fig. 11(a), poor mesh quality is produced by submapping due to the lack of an optimal corner assignment. By applying templates described in this

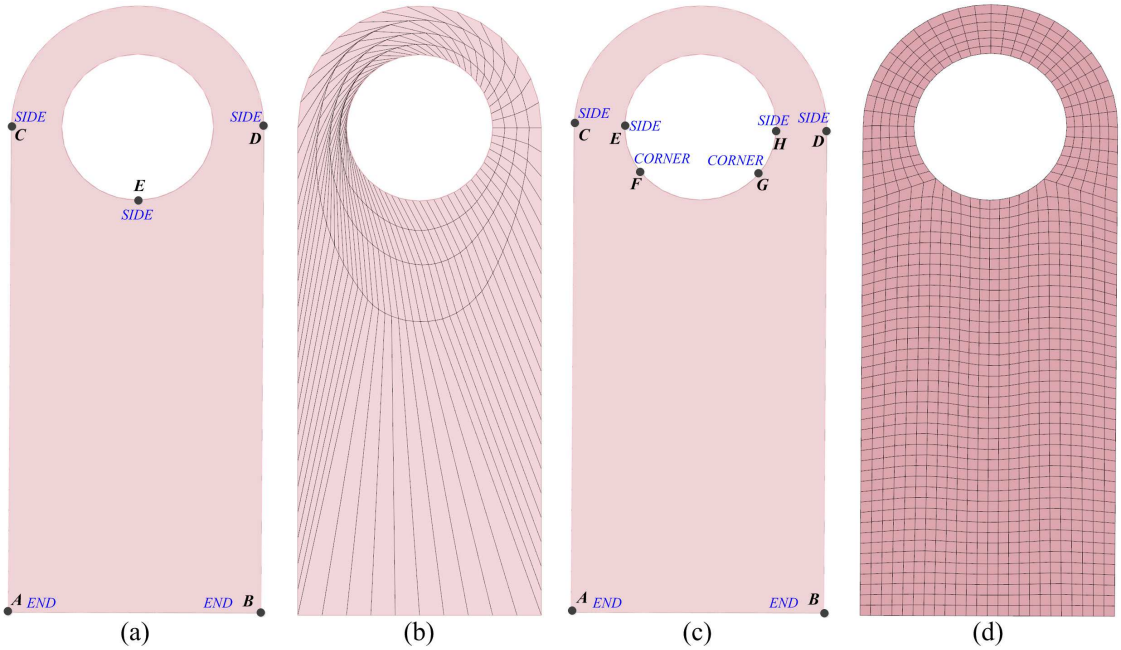

Fig. 12. Structured grid generation of a surface with a half concentric ring and a half circle feature by submapping through templates: (a)corner assignment by *Cubit 13.2*; (b)structured grid for (a); (c)corner assignment based on templates in this paper; (d)structured grid for (c)

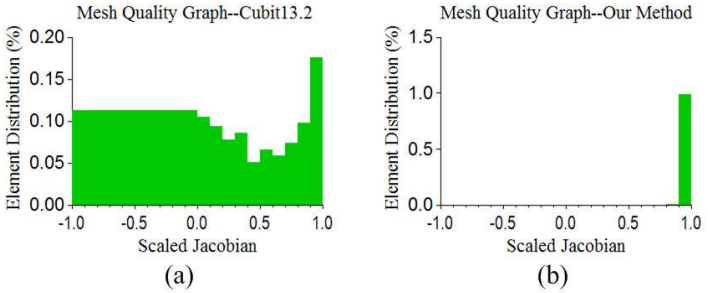

Fig. 13. Mesh quality histogram: (a)mesh quality histogram for Fig. 12(b); (b)mesh quality histogram for Fig. 12(d)

paper, two new vertices are virtually inserted and surface vertices are classified as Fig. 10(c). The corresponding structured quadrilateral mesh is shown in Fig. 10(d) with good mesh quality, which is illustrated in Fig. 11(b).

Figure 12 shows a surface with an half concentric ring feature and an half round feature, which is challenging for current existing automatic submappers. What is more, it is a multiply-connected surface and it is difficult to generate consistent edge parameterization for the outmost and interior boundaries. The most intuitive way to generate a valid corner assignment is to convert the vertex *A* and *B* from *END* in Fig. 12(a) to *SIDE* where the resulting structured quadrilateral mesh is shown in 12(b) by *Cubit 13.2* with the mesh quality histogram plotted in Fig. 13(a). Our method can classify surface vertex types as Fig. 12(c). The corresponding structured quadrilateral mesh is shown in Fig. 12(d) with better mesh quality (Fig. 13(b)) compared to the one in Fig. 12(b). There are two reasons why *Cubit 13.2* fails to generate structured meshes by *Submapping* with good mesh quality: (a) the geometry is a multiply-connected surface and there is no good method for matching boundary nodes between the outmost boundary and interior boundary; (b) poor corner assignment results in poor structured mesh quality.

The third example in Fig. 14(a) shows a surface with a combination of chamfer, fillet and round features, which present difficulties for current existing submapping methods. By applying templates described in this paper, the corner assignment can be generated as Fig. 14(b) with virtually inserted vertices. The resulting structured quadrilateral mesh

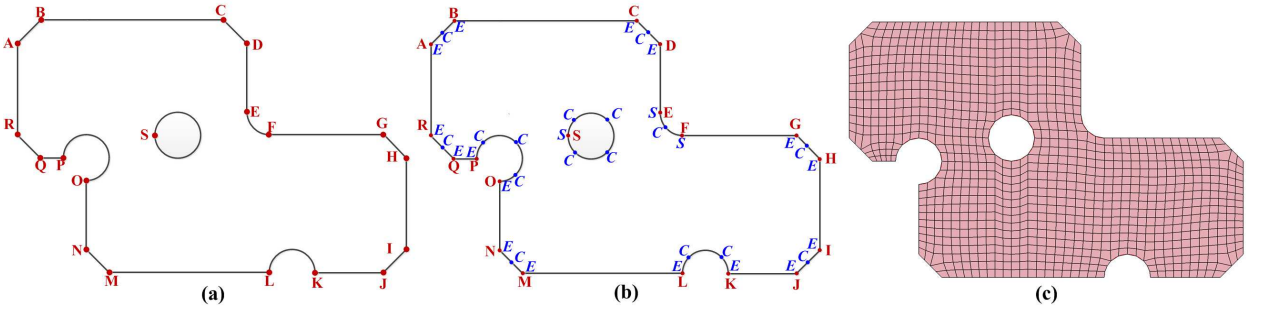

Fig. 14. Structured grid generation of a surface with fillets, rounds and chamfers by submapping using optimization and templates: (a)geometry model; (b)corner assignment; (b)structured quadrilateral meshes

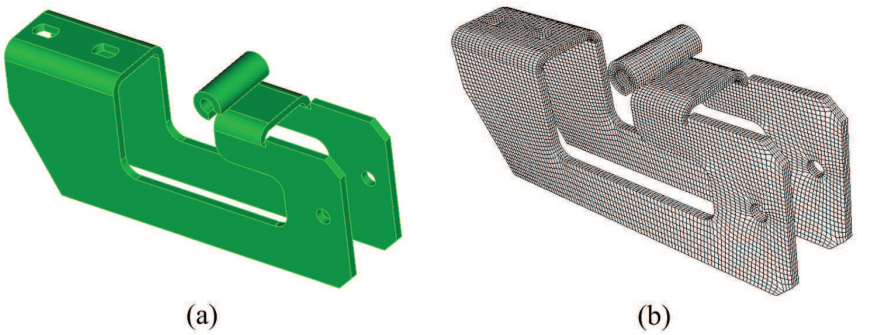

Fig. 15. Structured all-quad mesh of a mechanical part generated by submapping: (a)geometry Model; (b)all-quad mesh for all the surfaces on a mechanical part generated by submapping

is shown in Fig. 14(c). *Cubit 13.2* fails to generate structured quadrilateral mesh for this case due to incapabilities of producing interval matching, which is indirectly caused by invalid corner assignment, namely, meshability caused by bad corner assignment.

The remaining three examples are real-application examples from industry. Figure 15 shows a mechanical part whose bounding surfaces are to be meshed by submapping. The geometry contains several chamfer, fillet, round and concentric ring features, which results in an invalid corner assignment by current existing submapping methods. By applying templates described in this paper and vertex type adjustment based on optimization, an optimal corner assignment can be generated and resulting structured quadrilateral mesh is shown in Fig. 15(b). The fifth example in Fig. 16 shows a mechanical gear with a bond structure. The gear contains several concentric ring features. By applying templates described in Fig. 7, an optimal corner assignment can be generated by our method and subsequent structured quadrilateral mesh is shown in Fig. 16(b). Figure 17 shows another mechanical part with several round and concentric ring features. Our method can generate a valid vertex classification for surface vertices and structured quadrilateral mesh for all the bounding surfaces as Fig. 17(b). The last example in Fig. 18 is a blade-like structure and the resulting structured quadrilateral mesh is shown in Fig. 18(b).

6. Discussion, Conclusions, and Future Work

In this paper, a method for corner assignment during the submapping process is described, which uses optimization and a carefully-chosen set of constraints to arrive at an acceptable submap solution. Our method uses a template-based approach to recognize features like rounds, fillets, chamfers and concentric rings commonly found on real-world models (and on which current angle-based corner assignment tends to fail). Corner assignment based on a combination of angles and templates tends to do better than that based on the angle alone, as showed by a set of real-world example models. For those problems where the template plus angle-based corner assignment does not result in

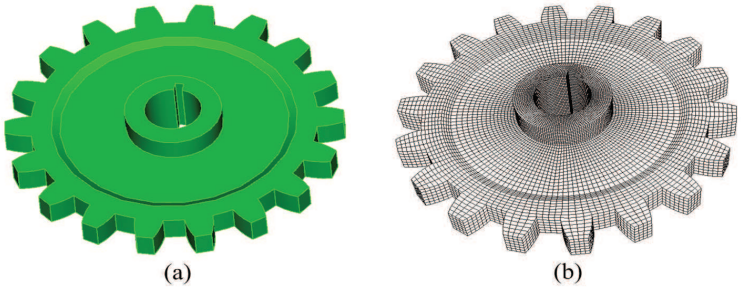

Fig. 16. Structured quadrilateral meshes of a gear generated by submapping: (a)geometry Model; (b) all-quad mesh for gears

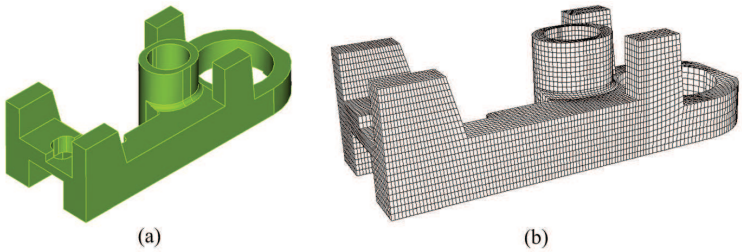

Fig. 17. Structured all-quad mesh of a mechanical part generated by submapping: (a)geometry Model; (b) structured quadrilateral meshes for all the surfaces on a mechanical part by submapping

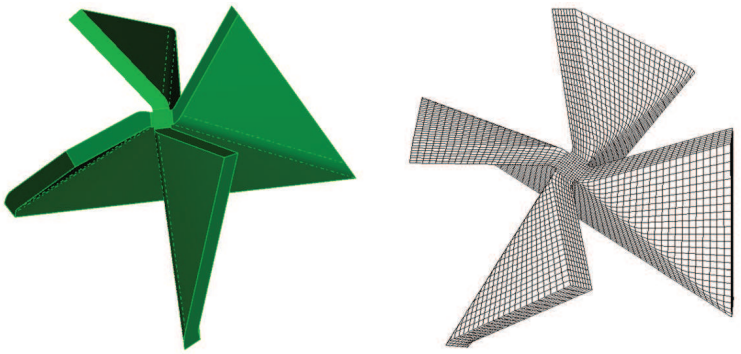

Fig. 18. Structured all-quad mesh of a mechanical part generated by submapping using optimization: blade part (a)geometry model; (b)structured quadrilateral meshes for curved surfaces

meeting the $sum = 4 - 4 * g$ submapping constraint, optimization is used to adjust vertex classification to meet the constraint that also seeks to minimize the maximum change in any one vertex type. The examples described above show a variety of cases where current tools fail and our approach succeeds.

For current meshing problems, the approach described in this paper can improve both the time to mesh and the quality of the resulting mesh. However, this capability will be of increasing importance as more capable sweeping algorithms[5–8] become more widespread. In particular, the introduction of multiple source to multiple target sweeping (multisweeping) capability[7,8] can dramatically increase the need for better submapping corner assignment. To cite just three examples: previous work has been done to mesh pipe networks [15] and networks of blood vessels [16]; cylindrical through-holes are also quite commonly found in models that would otherwise be sweepable in a direction orthogonal to the through-hole. Automatic handling of the setup for submapping problems of this type, along with multisweeping, could drastically reduce user time required to mesh these models.

One part of this combination of submap setup and multi-sweeping that has not been addressed in the context of our work is automatic decomposition of connected surfaces after corner assignment. To illustrate this problem, consider the model shown in Figure 10, but swept into the third dimension perpendicular to the page (sweeping from the top to

the bottom). Template-based corner assignment results in vertices G and H shown in Figure 10(c). In order to sweep the 3D model, one would also have to decompose the surface adjacent to edge $AHGF$ and perpendicular to the page, resulting in a new target surface (adjacent to AH), a new side or *linking* surface (adjacent to HG) and a new *source* surface (adjacent to GF). Given that one of those surfaces would need to be submappable for sweeping to be possible, it would likely not be difficult to compute the splitting lines and to perform that split following the corner detection.

Another extension of this work is to generate structured hexahedral meshes by submapping where all the bounding surfaces are meshed by structured quadrilateral meshes and every interior node is shared by exactly eight hexahedra. When meshing the bounding surfaces with structured quadrilateral meshes, our method described in this paper can be applied directly. For the volume submapper, one more thing to be resolved is to classify edges shared by two adjacent surfaces so that structured hexahedra can be embedded into the geometry. During this process, templates and corner assignment optimization method described in this paper should be able to be directly applied as well.

Acknowledgement

This work was funded under the auspices of the Nuclear Energy Advanced Modeling and Simulation (NEAMS) program of the Office of Nuclear Energy, and the Scientific Discovery through Advanced Computing (SciDAC) program funded by the Office of Science, Advanced Scientific Computing Research, both for the U.S. Department of Energy, under Contract DE-AC02-06CH11357. We also thank all the Fathom members (from both Univ. of Wisconsin-Madison and Argonne National Lab) for their efforts on the CGM[17], MOAB[18], Lasso[19] and MeshKit[20] libraries, which were used heavily to support this work. Finally, I'd like to thank my current employer CD-adapco for supporting my attendance for 24th IMR.

References

[1] White, David R., and Timothy J. Tautges. "Automatic scheme selection for toolkit hex meshing." International Journal for Numerical Methods in Engineering 49.1 (2000): 127-144.
[2] Shepherd, Jason F., Mitchell, Scott A., Knupp, Patrick, White, David R. Methods for Multisweep Automation. Proceedings of the 9th International Meshing Roundtable, Sandia National Laboratories Report SAND2000-2207, September 2000.
[3] Whiteley, M., White, D., Benzley, S., Blacker, T. Two and three-quarter dimensional meshing facilitators. Engineering with Computers, 1996(12):144-154.
[4] Ruiz-Girones, E., Sarrate, J. Generation of structured meshes in multiply-connected surfaces using submapping. Advances in Engineering Software, 41.2(2010):379-387.
[5] Cai, S. and Tautges T.J. Robust One-to-One Sweeping with Harmonic ST Mappings and Cages. Proceedings of the 22nd International Meshing Roundtable, Springer International Publishing, 2014: 1-19.
[6] Cai, S. and Tautges T.J. One-to-one sweeping based on harmonic ST mappings of facet meshes and their cages. Engineering with Computers, 2014: 1-14.
[7] Cai, S. and Tautges T.J. Surface Mesh Generation based on Imprinting of S-T Edge Patches. Procedia Engineering, 2014(82): 325-337.
[8] Cai, S. Algorithmic Improvements to Sweeping and Multi-Sweeping Volume Mesh Generation. University of Wisconsin-Madison, PhD dissertation, 2015(4).
[9] Eikland, K. *lpsolve*. http://sourceforge.net/projects/lpsolve, 2010
[10] Eiselt, H., Sandblom, C. Linear Programming and its Applications. Springer Science & Business Media, 2007
[11] Gass, S.I., Linear Programming: Methods and Applications. Courier Corporation, 2003
[12] Nicholson, T.A.J. Optimization in Industry: Optimization Techniques. Transaction Publishers, 2007.
[13] Mitchell, S.A. High Fidelity Interval Assignment. Proceedings of 6th International Meshing Roundtable, 1997(33):44.
[14] Mitchell, S. A. Simple and fast interval assignment using nonlinear and piecewise linear objectives. In Proceedings of the 22nd International Meshing Roundtable, 2014: 203-221.
[15] Miyoshi, K, and Blacker, T. Hexahedral Mesh Generation Using Multi-Axis Cooper Algorithm. Proceedings of the 9th International Meshing Roundtable, Sandia National Laboratories Report SAND2000-2207, September 2000.
[16] Verma, Chaman Singh, Fischer, Paul F., Lee, SE, and Loth, F. An all-hex meshing strategy for bifurcation geometries in vascular flow simulation. Proceedings of the 14th international meshing roundtable. Springer Berlin Heidelberg, 2005.
[17] T. J. Tautges: CGM: A geometry interface for mesh generation, analysis and other applications. Engineering with Computers, 17(3):299-314 (2001)
[18] T. J. Tautges: MOAB: a mesh-oriented database. Sandia National Laboratories, (2004)
[19] Lasso: https://trac.mcs.anl.gov/projects/ITAPS/wiki/Lasso
[20] T. J. Tautges and R. Jain: Creating geometry and mesh models for nuclear reactor core geometries using a lattice hierarchy-based approach. Engineering with Computers, 28(4):319-329 (2012)