A Hex-Mesh Generation Method Using Mean Value Coordinates

CHEN RUIZHI, XI PING
School of Mechanical Engineering and Automation
Beihang University
No. 37, Xueyuan Road, Haidian District, Beijing
CHINA
richardchen@me.buaa.edu.cn

Abstract: - In order to find an applicable way of generating high-quality all-structured hexahedral meshes, a method of generating structured hexahedral mesh using mean value coordinates is proposed. Firstly, a new edge classification method and its execution condition are proposed which greatly cuts down limits for sub-mappable volumes by means of spreading local coordinate system. Secondly, virtual edges are created according to values of mean value coordinates in the computational domain and added into volume's corresponding directed graph to eliminate inner loop vertices in the graph; therefore virtual volume decomposition process is avoided. Eventually, examples are given to illustrate the applicability of the proposed method. Meshing results manifest that the proposed method can stably generate high-quality hex-meshes and indicate potential application for hexahedral meshing of sub-mappable volumes.

Key-Words: - hexahedral mesh generation; submapping; mean value coordinates.

1 Introduction

Parameter modification, mesh generation, simulation and result evaluation are four key steps in the iterative process of three-dimensional mechanical model optimal design. Mesh generation is a critical input for finite element method and finite difference method, thus the precondition of simulation.

Accuracy and duration of simulation are highly dependent on the average and worst quality of the mesh. Because of providing better element stiffness matrix, high quality meshes always quickly lead to a more accurate solution. The quality of hexahedral mesh is much better than tetrahedral. In order to achieve the same level of accuracy, tetrahedral mesh typically requires four to ten times more elements and much longer time than a hexahedral mesh [1]. Due to higher quality than ordinary hexahedral mesh, Structured hexahedral mesh can significantly reduce processing time and simplify parallelization, and consequently becomes a preferred choice of mesh generation [2].

As far as concerned, no general valid solution has been given which can realize hexahedral mesh generation stably and automatically. In actual projects, mesh generation had been done manually so as to pursue mesh quality. Therefore plenty of time had been wasted.

Many research has been done to realize auto hex-mesh generation [3], many algorithms are given including grid-based methods [4][5], medial surface methods [6], advancing front techniques [7][8], etc. Although mature and stable, none of the methods mentioned above can generate highly structured mesh.

Submapping [9] method can automatically generate hexahedral mesh for blocky volumes. After the edge classification and volume decomposition process, product model can be decomposed into many blocks topologically similar to hexahedron. These simpler blocks can easily be meshed by using well-known methods. Mesh of the recognized model consists of meshes of all the blocks. Nevertheless, existing edge classification methods have difficulty in processing general blocky volumes, and the situation encountered during volume decomposition maybe too complex to find a valid solution, which becomes a major obstacle to apply submapping method.

Submapping using Recognized Model [10] (Referred to as SRM) is a feasible approach of generating structured hexahedral mesh for general blocky volumes. Recognized model is a polycube in the computational domain which has the same topology structure as the original model. One can get structured mesh through meshing the polycube and mapping the mesh back onto the original model.

Recognized model has a regular shape and can be meshed directly by submapping. The process of mapping mesh of polycube onto the original model can be regarded as the inverse process of constructing recognized model, which can be easily realized using well-known methods. How to get

recognized model from the original model is the key step of SRM, thus a focal point for the community.

Y. Su proposed an algorithm of constructing recognized model in Ref. [11] by tessellating the original model into small triangles and then employing a fuzzy logic method to determine the normal directions of the triangles. Y. Su's method consists of so many complex details that its reliability is difficult to guarantee.

For three-dimensional product model topologically similar to polycube, E. Ruiz-Gironés' method [12][13] constructed the linear programming:

$$\begin{aligned} &\min \sum_{e} \omega_e \left| n_e - N_e \right| \\ &s.t. \ \sum_{e\in I_k^+} n_e = \sum_{e\in I_k^-} n_e, \ \sum_{e\in J_k^+} n_e = \sum_{e\in J_k^-} n_e, \ \sum_{e\in K_k^+} n_e = \sum_{e\in K_k^-} n_e \end{aligned} \quad (1)$$

where N_e is the target element number on edge e, ω_e is the weight related to edge e, both of which are designated before solving equation(1), I_k^+ is the set of edges on edge loop L_k which has the same direction with I axis of the computational domain. $\{L_1, L_2, \cdots\cdots, L_k\}$ is a basis of all edge loops of the model. According to edge classification results and solution of Equation 1, recognized model can be obtained quickly. However, for volumes containing inner loop vertices, virtual volume decomposition is still needed which implies that problem of volume decomposition still exists.

SRM significantly broadens submapping method's scope of application, but its disadvantages still exist. Edge classification specifies the axis direction corresponding to each edge in the computational domain, which is a precondition of applying SRM, however, edge classification method is too simple to classify edges in complex situations. The shortage of general edge classification method in submapping also limits the SRM's scope of application. Furthermore, problem in volume decomposition also bothers SRM: volume decomposition process eliminates inner loop vertices' free state from the original structure and combines them tightly with sub-structures, and thus the instability of volume decomposition algorithm caused by the free state of inner loop vertices is avoided. In order to get rid of the vertices in free state, SRM has no other choice but virtual volume decomposition.

All of the aforementioned methods have difficulties in generating all-structured hexahedral meshes especially when there are multi-loop faces in the model. In order to find a robust way of generating all-structured hexahedral meshes, our work introduces a new Hexahedral mesh generation method using Mean Value coordinates (HMV) which remedies the deficiencies of sub-mapping and broadens its scope of application. Firstly, a new edge classification method and its execution condition are proposed. Secondly, virtual edges are created according to values of mean value coordinates in the computational domain. Thirdly, virtual edges are added into volume's corresponding directed graph to eliminate inner loop vertices in the graph. Therefore virtual volume decomposition process is avoided. Eventually, examples are given to illustrate the applicability of the proposed method.

2 Methodology

In order to get a robust approach of structured hexahedral mesh generation, a new Hexahedral mesh generation method using Mean Value coordinates (referred to as HMV) is proposed. It consists of the following steps:

1. Edge classification
2. Interval assignment initialization using linear programming
3. Locate inner loop vertices using mean value coordinates
4. Add virtual edges into initial directed graph
5. Mesh generation

2.1 Edge classification

We suppose the existence of a polycube in the computational domain has a similar topology structure with three-dimensional product model M. Therefore, any edge in M corresponds to an edge parallel to axis in the computational domain.

Let I, J, K represents for the axes of the computational domain. If the corresponding edge of edge e is parallel to axis I, then edge e can be called I-direction edge. The J-direction edge and K-direction edge are similarly defined.

Given model M and its edge set $E = \{e_i\}$, let E_I be an edge set consisting of all I-direction edges in the model. E_J and E_K are similarly defined. $E = E_I \bigcup E_J \bigcup E_K$ and any edge e_i can only belongs to one axis-direction set.

A demonstration of edge classification is shown in Figure 1. Local coordinates and its spread direction of initial vertex v_0 are shown on the left, where $e_i \in E_I, e_j \in E_J, e_k \in E_K$. The polycube

corresponding to the model in the computational domain is shown on the right.

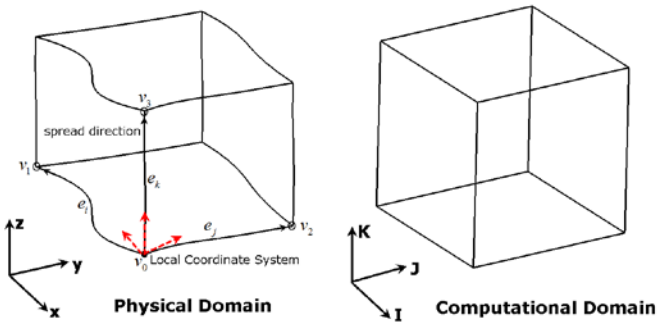

Fig.1 Edge classification and spread of coordinate system. e_1, e_2, e_3 are three adjacent edges of vertex v_0. The three arrows starting at v_0 represent the local coordinate system corresponding to v_0. According to the local coordinate system of v_0, the local coordinate systems corresponding to vertices v_1, v_2, v_3 can also be obtained by use of the Algorithm of Spreading local Coordinate System (ASCS).

Edge classification is realized by means of spreading local coordinate system. The following describes detailed steps of spreading local coordinate system:

1. Choose $\forall v_i \in V$, assign its local coordinate system
2. Determine which axis-direction edge set should each edge e_j which adjacent to v_i belongs to
3. Determine local coordinate system of another vertex v_k on edge e_j
4. Iterate step (2) and (3) until local coordinate system which has been spread to all vertices of the model

The Algorithm of Spreading local Coordinate System (ASCS) is crucial to edge classification. definition and proposition are given to help describe and discuss ASCS in details.

Let e_l, e_m, e_n denote for edges adjacent to vertex v_i. The unit tangent vector of e_l, e_m, e_n at v_i are represented by $\vec{n_l}, \vec{n_m}, \vec{n_n}$. $\vec{X}_i, \vec{Y}_i, \vec{Z}_i$ represent for axis unit vectors of Local coordinate system S_i corresponding to v_i. $\vec{X}_i, \vec{Y}_i, \vec{Z}_i$ correspond to I, J, K direction in the computational domain respectively.

DEFINITION 1 If $\left\|\vec{n_l} \cdot \vec{X}_i\right\| \geq \max\left(\left\|\vec{n_l} \cdot \vec{Y}_i\right\|, \left\|\vec{n_l} \cdot \vec{Z}_i\right\|\right)$, then $e_l \in E_I$. One can also define $e_l \in E_J$ or $e_l \in E_K$ similarly. The principle used to determine edge direction is called Maximum Inner Product Principles.

Given local coordinate system S_i, all three edges adjacent to vertex v_i can be classified based on Maximum Inner Product Principles.

DEFINITION 2 For $\forall e \in \{e_l, e_m, e_n\}$, $\forall E \in \{E_I, E_J, E_K\}$, if $\left(\{e_l, e_m, e_n\} \setminus \{e\}\right) \bigcap E = \Phi$ when $e \in E$, then local coordinate system S_i can be compatible with vertex v_i, denoted by $S_i \in C(v_i)$, where $C(v_i)$ represents for the set consisting of all the local coordinate systems compatible with v_i.

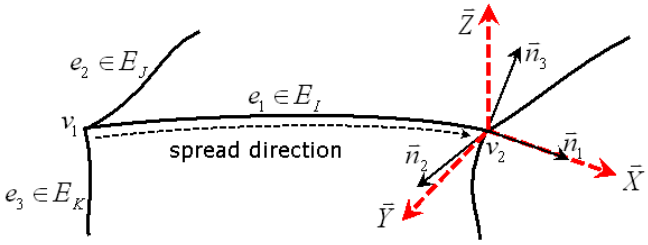

Fig.2 Spread of local coordinate system. $e_1 \in E_I, e_2 \in E_J, e_3 \in E_K$ represents three edges adjacent to vertex v_1, the unit tangent vectors at v_2 are donated by $\vec{n}_1, \vec{n}_2, \vec{n}_3$. Vectors $\vec{X}, \vec{Y}, \vec{Z}$ represent the axes of local coordinate system corresponding to v_2. $\vec{X}, \vec{Y}, \vec{Z}$ can be calculated according to the local coordinate system of v_1.

The unit axis vectors of local coordinate system S_2 are calculated as follows:

$$\begin{cases} \vec{X} = \vec{n}_1 \\ \vec{Z} = \dfrac{\vec{n}_2 \times \vec{n}_1}{\left\|\vec{n}_2 \times \vec{n}_1\right\|} \\ \vec{Y} = \vec{X} \times \vec{Z} \end{cases} \quad (2)$$

PROPOSITION 1 Any S_i calculated by ASCS is compatible with its corresponding vertex v_i if for $\forall v_i \in V$, $\left\|\vec{n_1} \times \vec{n_2}\right\| \geq \max\left(\sqrt{2}\left\|\vec{n_1}\right\|\left\|\vec{n_2}\right\|/2, \sqrt{2}\vec{n_2} \times \vec{n_1} \cdot \vec{n_3}\right)$.

PROOF: As shown in figure 2, vectors $\vec{n}_1, \vec{n}_2$ share the same plane with $\vec{X}, \vec{Y}$, thus sine value of angle between $\vec{n}_1, \vec{n}_2$ should be larger than $\sin 45^\circ$, which can be simplified into $\left\|\vec{n_1} \times \vec{n_2}\right\| \geq \sqrt{2}\left\|\vec{n_1}\right\|\left\|\vec{n_2}\right\|/2$. In order to distinguish $\vec{n}_3$ from $\vec{n}_1, \vec{n}_2$, cosine value of the angle between $\vec{n}_3$ and $\vec{Z}$ should be smaller than $\cos 45^\circ$, which can be simplified into $\left\|\vec{n_1} \times \vec{n_2}\right\| \geq \sqrt{2}\vec{n_2} \times \vec{n_1} \cdot \vec{n_3}$. □

There may be more than one connected graph in initial directed graph G_0. Given proper initial local coordinate system, ASCS can finish the spread of local coordinate system for any connected graph in G_0. Although ASCS has some requirement to the model, ASCS relieves model from the traditional

limit that angles between edges should be an integer multiple of $\pi/2$, and submapping method's scope of application is significantly broadened.

2.2 Interval assignment initialization

The recognized model in the computational domain must be polycube to guarantee mean value interpolation process, thus it is inevitable to use linear programming method to accomplish interval assignment which is to designate initial element number on edges.

Given vertex set $V=\{v_i\}$ and edge set $E=\{e_i\}$ of model M, any given edge direction, initial directed graph G_0 can be constructed. The method E. Ruiz-Gironés proposed in Ref.[12] is employed to construct linear programming as shown in Equation 1, by solving which initial element number N_i of any edge e_i can be got. Initial recognized model can be easily constructed using edge classification results and initial element numbers.

2.3 Location of inner loop vertices using mean value coordinates

For the purpose of locating inner loop vertex with outer loop vertex on the same surface, associated point pairs are created. Let $F^C=\{f_i^c\}$ represents for set of all surfaces containing inner loop, $V(f_i^c)$ and $E(f_i^c)$ represents for sets of all vertices and edges of f_i^c respectively, V_{in} and E_{in} represents for set of all inner loop vertices and edges in model M respectively.

DEFINITION 3 Define v_i、v_k as an associated point pair, marked as $a_{ik}=\{v_i,v_k\}$, if $\forall v_i\in V_{in}$, $\exists v_k\in V(f_j^c)\backslash V_{in}$, *s.t.* $\|p(v_i)-p(v_k)\|\le\|p(v_i)-p(v_l)\|$ where $\forall l\ne k$, $v_l\in V(f_j^c)\backslash V_{in}$, $p(v_i)$ represents for parameter coordinates of v_i on f_j^c.

Let $A=\{a_{ik}\}$ be a representative of set of all associated point pairs.

Given $a_{ik}=\{v_i,v_k\}$, then $\exists f_j^c\in F^C$, *s.t.* $v_i,v_k\in V(f_j^c)$. Through edge classification and the initialization of element number on edges, for $\forall v_i\in V(f_j^c)\backslash V_{in}$, its coordinates (I_i,J_i,K_i) in the computational domain can be got. By means of mean value coordinates, computational domain coordinates of all vetices in $V(f_j^c)\bigcap V_{in}$ can be calculated. Moreover, coordinate difference Δa_{ik} corresponding to a_{ik} can also be calculated, which helps to realize the locating of inner loop vertices with respect to outer loop vertices.

By fixing the attribute value of the outer loop vertices of region Ω, the problem of interpolating attribute values of inner vertices can be considered as the problem of finding barycentric coordinate of inner vertices with respect to outer loop vertices.

If $\partial\Omega$ is piecewise linear shape, then any of wachspress coordinate [14], discrete harmonic coordinate, harmonic coordinate [15] and mean value coordinate [16][17] can be chosen to build barycentric coordinate of inner vertices. However, mean value coordinate has advantages. It can be easily computed and still works well even if Ω is concave [18], which make it more suitable for our interpolation requirements. Considering complex curves may be included within $\partial\Omega$, the integral form of mean value interpolation proposed by C. Dyken [19] is employed:

$$g(x)=\int_0^{2\pi}\sum_{j=1}^{n(x,\theta)}\frac{(-1)^{j-1}}{\rho}f(p_j(x,\theta))d\theta\Big/\Phi,\Phi=\int_0^{2\pi}\sum_{j=1}^{n(x,\theta)}\frac{(-1)^{j-1}}{\rho}d\theta \quad (3)$$

where $n(x,\theta)$ is the number of intersection points $p_j(x,\theta)$ at which $\partial\Omega$ intersects the radial with starting point x and starting angle θ. p_j is arranged according to its distance away from x so that $\|p_1-x\|\le\|p_2-x\|\le\cdots\le\|p_n-x\|$.

Suppose surface $\tilde{f}_j^c$ in the computational domain is corresponding to $\tilde{f}_j^c$ perpendicular to K axis, then for $\forall v_i\in V(f_j^c)$, $K_i=const$, which means only coordinate value of I,J need to be interpolated. Take interpolating coordinate value of I for example, we describe the locating method using mean value coordinates.

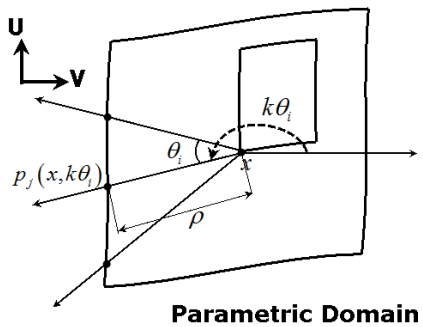

Fig.3 Radials of mean value interpolation. $p_j(x,k\theta_i)$ represents the j-th intersection point between outer edge loops and the radial which starts at vertex x and has an angle $k\theta_i$ with the U-axis of the parametric domain. ρ represents the distance

between x and $p_j(x,k\theta_i)$. In practice, the radials starting at x should averagely split the parametric domain. In order to be laconic, only a few radials are drawn in this figure.

As shown in figure 3, the mean value interpolation process is carried out on the parametric plane corresponding to f_j^c. An iteration process is adopted to calculate the approximation of the integrals in Equation 3 as follows:

1. Given iterative precision $\varepsilon > 0$, constant $m \in N$.
2. When doing the i-th iteration, radials are created to intersect with all curves corresponding to outer loop edges of f_j^c in the parametric domain. All radials start at x and averagely split the whole parametric domain with $\theta_i = \pi / im$ as interval angel.
3. Calculate the approximation of integrals in Equation 3 in the i-th iteration:

$$g_i = \sum_{k=1}^{2im}\left(\sum_{j=1}^{n(x,\theta)}\frac{(-1)^{j-1}}{\rho} f(p_j)\right)\theta_i \Big/ \Phi, \Phi = \sum_{k=1}^{2im}\left(\sum_{j=1}^{n(x,\theta)}\frac{(-1)^{j-1}}{\rho}\right)\theta_i \quad (4)$$

where $f(p_j)$ represents for I coordinate value of $p_j(x,k\theta_i)$ in the computational domain.

4. Calculate convergence precision $\delta = |1 - g_{i+1}/g_i|$. Iterate step (2) and (3) until $\delta < \varepsilon$, then g_{i+1} will be the output approximation of $g(x)$ in Equation 3.

Convergence of the iteration process is guaranteed by Cauchy's Convergence Test. As far as concerned, no valid solution has been found to averagely split three-dimensional domain using radials, which is referred to as the "Thomson Problem" in the community, therefore how to generalize the aforementioned inner loop locating method in three-dimensional domain to eliminate free state of vertices of void structure remains an open problem.

2.4 Virtual directed graph construction

2.4.1 Virtual edges creation

DEFINITION 4 Edges added into initial directed graph for connecting associated data pairs are called virtual edges. Virtual edges do not exist in the three-dimensional model, and parallel to one axis of computational domain.

The computational domain coordinate differences of points in a_{ik} are denoted by $\Delta a_{ik} = (\Delta I_{ik}, \Delta J_{ik}, \Delta K_{ik})$, where $\Delta I_{ik} = I_i - I_k$ and I_i represents for I coordinate value of v_i in the computational domain.

Using the location relationship calculated in Section 2.3, Δa_{ik} of $a_{ik} \in A$ is determined.

PROPOSITION 2 For $\forall a_{ik}$, $\Delta I_{ik} \times \Delta J_{ik} \times \Delta K_{ik} = 0$.

PROOF: Given $a_{ik} = \{v_i, v_k\}$, $\exists f_j^c \in F^C$, *s.t.* $v_i, v_k \in V(f_j^c)$. Because of the existence of polycube in the computational domain which has exactly the same topology structure with model M, surface $\tilde{f}_j^c$ in the computational domain corresponding to f_j^c must be perpendicular to some axis.

If $\tilde{f}_j^c$ is perpendicular to I axis, then for $\forall a_{ik} \in V(f_j^c)$, $\Delta I_{ik} = 0$. Similarly we get $\Delta J_{ik} = 0$, $\Delta K_{ik} = 0$。 Therefore, for $\forall a_{ik}$, $\Delta I_{ik} \times \Delta J_{ik} \times \Delta K_{ik} = 0$. □

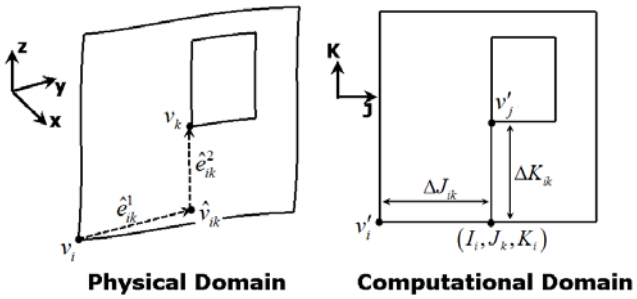

Fig.4 Creation of virtual edges. v_i', v_j' respectively represent for the corresponding point of v_i, v_j in the computational domain. $\hat{e}_{ik}^1, \hat{e}_{ik}^2$ are virtual edges newly-created. Edge $\hat{e}_{ik}^1$ starts at v_i and ends at $\hat{v}_{ik}$. Edge $\hat{e}_{ik}^2$ starts at $\hat{v}_{ik}$ and ends at v_k. (I_i, J_k, K_i) represents the coordinate of vertex $\hat{v}_{ik}$ in the computational domain. $\Delta J_{ik}, \Delta K_{ik}$ respectively represent the corresponding length of edge $\hat{e}_{ik}^1$ and $\hat{e}_{ik}^2$ in the computational domain.

For every $a_{ik} \in A$, virtual edges are created according to coordinate differences $\Delta a_{ik} = (\Delta I_{ik}, \Delta J_{ik}, \Delta K_{ik})$. Supposing $\Delta I_{ik} = 0$, then two virtual edges $\hat{e}_{ik}^1, \hat{e}_{ik}^2$ can be created in the physical domain as shown in Figure 4.

The vertices shared by different virtual edges are called virtual vertices. Vertex $\hat{v}_{ik}$ is shared by $\hat{e}_{ik}^1, \hat{e}_{ik}^2$ which makes it a virtual vertex. Virtual vertices do not exist in the physical domain; however they have coordinate values in the computational domain, take (I_i, J_k, K_i) for $\hat{v}_{ik}$ as an example. Virtual edges can be created similarly when $\Delta J_{ik} = 0$ or $\Delta K_{ik} = 0$.

For $\forall a_{ik} \in A$, the maximum number of virtual edges created according to Δa_{ik} is two. If the number of a_{ik} is m, then the maximum number of virtual edges need to be added to the initial directed graph

G_0 is $2m$. After added by virtual edges, Initial directed graph becomes virtual directed graph.

2.4.2 Virtual edge length assignment

Length assignment of all the virtual edges is an inevitable precondition for virtual directed graph to be used to mesh generation.

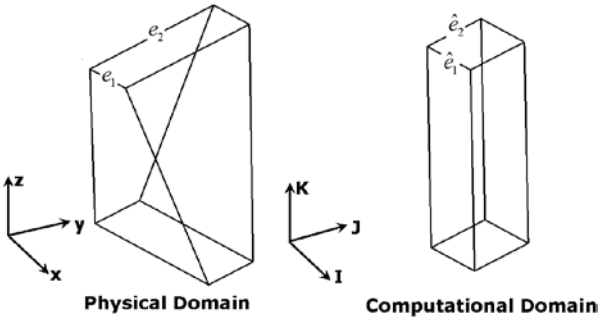

Fig.5 An example of face scaling. $\hat{e_1}, e_2$ respectively represent the corresponding edges of e_1, e_2 in the computational domain.

Compared to faces of the original model, the faces in the initial recognized model are compressed or tensed in axis directions of the computational domain. Fig.5 demonstrates an example of face scaling effect between original (left) and recognized model (right). Through linear programming, edge e_1 is tensed and edge e_2 is compressed in the computational domain.

The computational domain coordinate differences Δa_{ik} are calculated according to initial element numbers, thus the lengths of virtual edges in the computational and physical domain have the similar scaling relationship with the faces they belong to. Take the case shown in figure 4 for example, the method for assigning virtual edge lengths is proposed as follows.

In order to reflect the shape scaling of faces along axis direction between computational and physical domains, let

$$S_J = \frac{1}{n}\sum_j \frac{l(e_j)}{|J_{sj} - J_{ej}|} \qquad (5)$$

represents for the scaling factor in J-axis direction of the computational domain, where $e_j \in E(f_i^c) \cap E_J \setminus E_{in}$, let $l(e_j)$ represents for the length of e_j, J_{sj} and J_{ej} represents the J-axis coordinate value of start and end vertices of edge e_j respectively, n represents total element number in the set where e_j belongs to. S_J is a reflect of average scaling states of all edges in set $E(f_i^c) \cap E_J \setminus E_{in}$.

Virtual edges $\hat{e}_{ik}^1$ and $\hat{e}_{ik}^2$ have the same scaling factor as faces where they belong to, thus the length of $\hat{e}_{ik}^1$ and $\hat{e}_{ik}^2$ in the physical domain can be calculated as: $l(\hat{e}_{ik}^1) = |\Delta J_{ik}| \cdot S_J$, $l(\hat{e}_{ik}^2) = |\Delta K_{ik}| \cdot S_K$.

2.5 Mesh Generation

After adding virtual edges corresponding to all $a_{ik} \in A$ to the initial directed graph G_0, final directed graph G without inner loop vertices can be got. Using the same method as employed in Section 2.2 with G as an input, final element number n_i on any edge e_i can be obtained and used to generate structured mesh through well-known method.

Laplacian-Isoparametric method [20] is employed to fulfill the mesh generation process. Firstly, quadrilateral meshes of all surfaces of the model are generated according to final element number of edges. Secondly, structured hexahedral mesh is generated through the use of surface meshes.

Laplacian-Isoparametric method always leads to problem of solving large-scale linear equations, which is a specialized research field. In this paper, Matlab software is integrated to the HMV system to help solving large-scale linear equations.

3 Meshing experiments

Two different parts are tested to verify the feasibility of HMV method. Both parts have inner loop vertices. Model 1 is a simple mechanical part, while model 2 is the leading edge structure of some turbine blade.

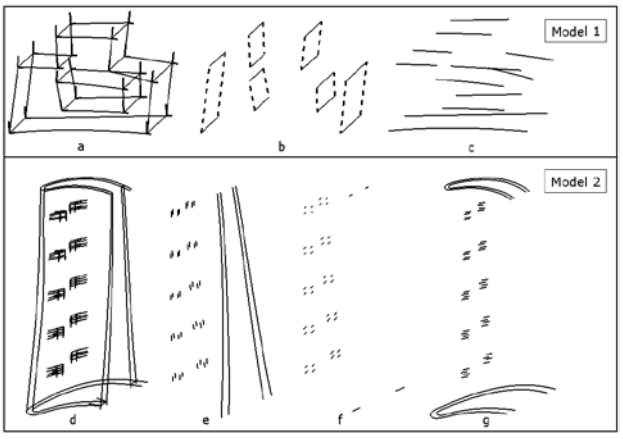

Fig.6 Results of edge classification. a. local coordinate axes of Model 1; b. edge classification result, dotted lines represent

edges belong to E_J, solid lines for edges belong to E_K; c. edge classification result, edges belong to E_I; d. local coordinate axes of Model 2; e. edges belong to E_I; f. edges belong to E_J; g. edges belong to E_K.

The results of spread of local coordinate system and edge classification are shown in figure 6. In order to show the spread results clearly, only two axes of the local coordinate system are displayed: short edge for I axis and long edge for J axis in model 1; short edge for I axis and long edge for K axis in model 2.

The results shown in figure 6 demonstrate the effect of edge classification method ASCS. ASCS can spread local coordinate system and classify edges effectively.

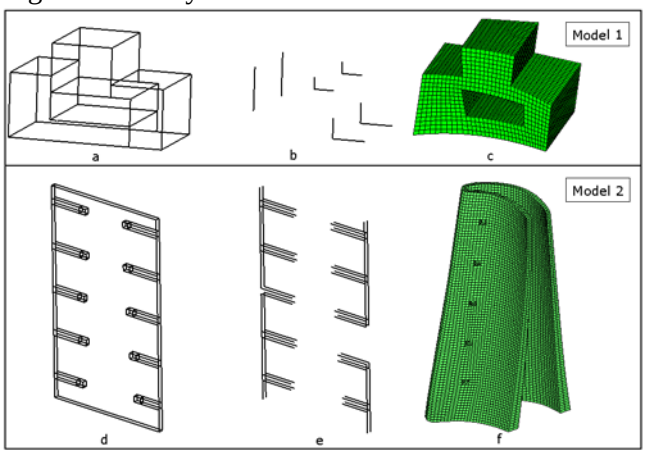

Fig.7 Virtual edges and meshing results. a. recognized model of Model 1; b. virtual edges generated for Model 1; c. mesh result of Model 1; d. recognized model of Model 2; e. virtual edges generated for Model 2; f. mesh result of Model 2.

Recognized models with virtual edges in the computational domain and the final meshes using HMV method are shown in figure 7. It can be clearly identified from 7.a and 7.d that virtual edges connect all inner loop vertices with the main structure and no vertex is at free state. Meshes generated are totally structured hexahedral meshes.

Table 1 gives out the iteration result of mean value inter-polation as $\varepsilon = 0.01$, $m = 2$. The results show that the locating method using mean value coordinates can give out result quickly even under high iterative precision.

Admittedly, many commercial meshing software, such as HyperMesh and ICEM, include modules corresponding to sub-mapping. These modules can assist mesh generation but restricted to simple quasi-polycube entity. Only after being artificially decomposed into simple sub-entities, complex quasi-polycube entities can be meshed by aforementioned software. Because of complex structures, original topology of mechanical models can easily be destroyed while decom-position. Thus, the meshing process usually takes several hours with the help of experts. Using method proposed in this paper, automation of hex-mesh generation is realized which guarantees high mesh quality and shortens the meshing period into several seconds.

With a similar element size 2 mm, other meshing methods are also used to mesh Model 1 shown in Fig.7 to compare their meshing performance with method proposed. Table 2 shows qualities of the mesh revealed in Fig.8. Meshing result of HMV has the maximum numbers of Max. Asp. Ratio as well as Min. Jac., which indicates the good quality of the mesh generated. It is obvious that HMV can generate all-structured hexahedral meshes with better qualities.

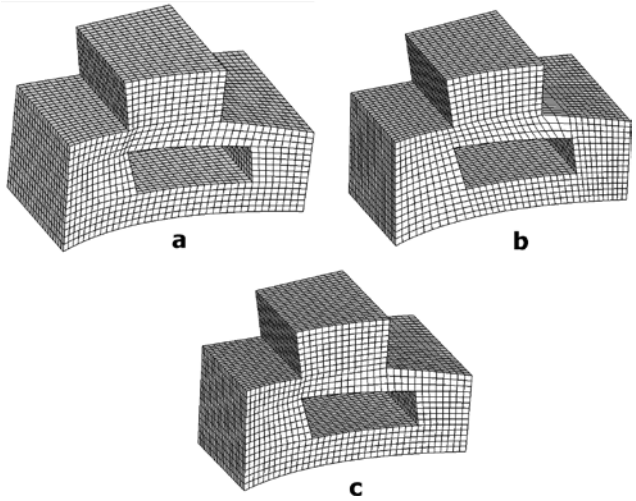

Fig.8 Meshing results of different methods. a. meshing result of method proposed in Ref.[13] by Ruiz-Gironés E; b. meshing result of method proposed in Ref.[11] by Y. Su; c. meshing result by HMV.

Table 1 Iteration results of locating inner loop vertices

Model	Max. Iteration time	Min. Interval angel (deg)	Max. Duration(s)
Model1	3	30	0.085
Model2	6	15	0.173
Model	inner loop point num	Total Duration(s)	Max. δ
Model1	8	0.617	0.004043
Model2	80	9.244	0.00788

Table 2 Mesh qualities in Fig.8

Model	Max. Asp. Ratio	Min. Jac.
Fig.8.a	1.82	0.89
Fig.8.b	1.61	0.7
Fig.8.c	2.24	0.94

4 Conclusion

Based on the traditional linear programming method, a hexahedral mesh generation method using mean value coordinates for models containing inner loop vertices is elaborated with case studies. A new edge

classification method is proposed which broadens submapping method's scope of application. The method of creating virtual edges according to mean value coordinates is introduced, which releases submapping method from volume decomposition process. Mesh generation results show the efficiency and stability of HMV when generating all-structured hexahedral mesh for solid models with inner loop vertices.

However, HMV method has its disadvantages. Firstly, there still remain a huge amount of models which cannot get their corresponding recognized models through the method proposed in this paper. In the future work, finding new methods for obtaining recognized models will be inevitable. Although the authors of this paper have made some effort on this issue [21][22], problems still exist. Secondly, mean value interpolation method can precisely give out the relationship between inner-loop vertices and the outer loop, but it will become time-consuming when dealing with faces which have a large number of inner-loop vertices. Therefore, fast precise interpolation methods are also urgently required.

References:

[1] Shivanna, K.H., Tadepalli, S.C., and Grosland, N. M, "Feature-based multiblock finite element mesh generation, Computer-Aided Design", Vol. 42, No. 12 pp.1108-1116, 2010,.

[2] Gregson, J., Sheffer, A., and Zhang, E, "All‐Hex Mesh Generation via Volumetric PolyCube Deformation", Computer graphics forum, Vol. 30, No. 5, pp. 1407-1416, 2011.

[3] Frey, P. and George, P. L, "Mesh generation", John Wiley & Sons, 2010.

[4] Su, Y., Lee, K. H., and Kumar, A.S, "Automatic hexahedral mesh generation for multi-domain composite models using a hybrid projective grid-based method", Computer-Aided Design, Vol. 36, No. 3, pp. 203-215, 2004.

[5] Zhang, Y., Liang, X., and Xu, G, "A robust 2-refinement algorithm in octree or rhombic dodecahedral tree based all-hexahedral mesh generation", Computer Methods in Applied Mechanics and Engineering, Vol. 256, pp. 88-100, 2013.

[6] Price, M.A., Armstrong, C. G., and Sabin, M.A, "Hexahedral mesh generation by medial surface subdivision: Part I. solids with convex edges", International Journal for Numerical Methods in Engineering, Vol. 38, No. 19, pp. 3335-3359, 1995.

[7] Ledoux, F. and Weill, J. C, "An extension of the reliable whisker weaving algorithm", In Proceedings of the 16th International Meshing Roundtable, Heidelberg, Berlin, January. Springer, pp. 215-232, 2008.

[8] Staten, M.L., Kerr, R.A., Owen, S.J., et al, "Unconstrained plastering—Hexahedral mesh generation via advancing‐front geometry decomposition", International journal for numerical methods in engineering, Vol. 81, No. 2, pp. 135-171, 2010.

[9] Roca X. and Sarrate J, "An automatic and general least-squares projection procedure for sweep meshing", Engineering with computers, Vol. 26, No. 4, pp. 391-406, 2010.

[10] Nagakura S., Noguchi S., Yamashita H. and Cingoski V, "Automatic Hexahedral Mesh Generation for FEM Using Shape Recognition Technique and Tree Method", Magnetics, IEEE Transactions on, Vol. 38, No. 2, pp. 417-420, 2002.

[11] Su Y., Lee K.H. and Kumar A.S, "Automatic hexahedral mesh generation using a new grid-based method with geometry and mesh transformation", Computer methods in applied mechanics and engineering, Vol. 194, No. 39, pp. 4071-4096, 2005.

[12] Ruiz-Gironés E. and Sarrate J, "Generation of structured meshes in multiply connected surfaces using submapping", Advances in Engineering Software, Vol. 41, No. 2, pp. 379-387, 2010.

[13] Ruiz-Gironés E. and Sarrate J, "Generation of structured hexahedral meshes in volumes with holes", Finite Elements in Analysis and Design, Vol. 46, No. 10, pp. 792-804, 2010.

[14] Floater M., Gillette A. and Sukumar N, "Gradient bounds for Wachspress coordinates on polytopes", arXiv preprint arXiv:1306.4385, 2013.

[15] Joshi P., Meyer M., DeRose T., Green B., Sanocki T, "Harmonic coordinates for character articulation", ACM Transactions on Graphics, Vol. 26, No. 3, pp. 71(1-9) , 2007.

[16] Floater M.S, "Mean value coordinates", Computer Aided Geometric Design, Vol. 20, No. 1, pp. 19-27, 2003.

[17] Hormann K. and Floater M.S, "Mean value coordinates for arbitrary planar polygons", ACM Transactions on Graphics, Vol. 25, No. 4, pp. 1424-1441, 2006.

[18] Ju T., Liepa P., Warrenc J, "A general geometric construction of coordinates in a convex simplicial polytope", Computer Aided

Geometric Design, Vol. 24, No. 3, pp. 161-178, 2007.
[19] Dyken C. and Floater M.S, "Transfinite mean value interpolation", Computer Aided Geometric Design, Vol. 26, No. 1, pp. 117-134, 2009.
[20] Herrmann, L. R, "Laplacian-isoparametric grid generation scheme", Journal of the Engineering Mechanics Division, Vol. 102, No.5, pp. 749-907, 1976.
[21] Chen R, Xi P, "A quadrilateral mesh generating method using mean value interpolation", Journal of Mechanical Science and Technology, Vol. 28, Issue 2, pp. 637-643, 2014.
[22] Chen R, Xi P, "A digraph-based hexahedral meshing method for coupled quasi-polycubes", Computer Methods in Applied Mechanics and Engineering, Vol. 268, pp. 18-39, 2014.