Efficient conformal tetrahedral finite element mesh generation for moving objects with contact by mesh adaptation

Hiroki MAEHAMA*, Hiroaki DATE* and Satoshi KANAI*
*Graduate School of Information Science and Technology, Hokkaido University
Nishi-9, Kita-14, Kita-ku, Sapporo 060-0814, Japan
E-mail: h_maehama@sdm.ssi.ist.hokudai.ac.jp

1. Introduction

Recently, for developing high quality products, CAE based on FEA has become absolutely imperative in product design. In some analyses of assembly models with movable parts (e.g. motor) such as structural analysis, CFD, heat analysis, or electro-magnetic field analysis, several types of meshes need to be generated. First, volume meshes of not only objects but also spaces are required for CFD and electro-magnetic field analysis. Second, conformal meshes where each intersection of two connecting elements can be formed by a single face, edge, or vertex, are preferred over non-conformal meshes because of the accuracy of the analysis. Third, hexahedral meshes are preferred over tetrahedral meshes from the viewpoint of accuracy of FEA. However, tetrahedral meshes are used in several analyses because automatic generation, editing and quality improvement of tetrahedral meshes are easier than for hexahedral meshes. Therefore, in this research, we deal with conformal tetrahedral meshes consisting of meshes both of objects and space (called “object mesh” and “space mesh” in this paper).
The general product design and analysis process for models with moving parts is shown in Fig. 1(a). In this process, poses of moving parts of CAD models are first modified. Then meshes of modified CAD models are generated. Finally, FEA is performed at each pose of the object in motion. However, meshing for objects and spaces at the same time is unstable and time-consuming. Therefore, many mesh adaptation methods (Alauzet, 2014, Barral et al., 2014, Chen et al., 2015, Compère et al., 2010, Dobrzynski and Frey, 2008) have been proposed to reduce the frequency of meshing for moving objects and space. As shown in Fig. 1(b), the methods can provide the mesh of each object pose by modifying the mesh connectivity and vertex positions depending on the object motion. However, the existing methods are inefficient because the mesh topology and geometry are globally adapted even if the differences in poses of the objects in motion are very small. In addition, the existing methods do not deal with contacts of the object meshes.

In this paper, we propose a new efficient mesh adaptation method of assembly conformal tetrahedral meshes for moving objects with contact. Our method consists of mesh segmentation, region extraction, mesh adaptation, and quality improvement. In our method, for efficient mesh adaptation, the mesh adaptation process is applied to only a set of space mesh elements around the moving object extracted by a distance from surfaces of the moving object mesh. Moreover, contacts are accurately detected using surface parameters, and to keep mesh conformity on the contact regions between object meshes, the topology and geometry of surface triangular meshes of contacting object meshes are adapted by vertex repositioning and local topological operations. In addition, in order to obtain a high quality mesh in each motion step, shape qualities of tetrahedral elements are improved and edge length is controlled by using a method based on Optimal Delaunay Triangulation (ODT) (Chen and Holst, 2011) with local topological operations.

The rest of this paper is organized as follows. First, related works are shown in Section 2. In Section 3, the outline of our method is first described. Then, the details of four processes (mesh segmentation, region extraction, mesh adaptation, and quality improvement) are mentioned. In Section 4, we demonstrate the effectiveness of our method through application of the method to some tetrahedral meshes. Finally, we conclude and discuss our future work in Section 5.

2. Related works

There are many tetrahedral mesh adaptation methods (Alauzet, 2014, Barral et al., 2014, Chen et al., 2015, Compère et al., 2010, Dobrzynski and Frey, 2008) for conformal mesh generation of moving objects and space. It is known that linear elasticity-based mesh adaptation methods are more robust and can produce higher quality meshes than Laplacian-based methods and spring analogy methods. In linear elasticity mesh adaptation methods, the movement of space mesh vertices is obtained by solving an elasticity-like equation. Dobrzynski and Frey (2008) proposed a linear elasticity-based mesh adaptation method with a quality improvement. In the quality improvement, edge length is controlled by Delaunay vertex insertion and edge collapse while keeping element shape qualities by edge flip and vertex repositioning. Compère et al. (2010) also proposed a linear elasticity-based mesh adaptation method. In this method, edge length is controlled by edge split and edge collapse, and element shape qualities are improved by edge flip. In addition, degenerated elements are removed by combinations of edge split and edge collapse. On the other hand,
Alauzet (2014) proposed a mesh adaptation method based on linear elasticity while keeping the number of vertices of a given mesh. In this method, trajectories of space mesh vertices are calculated at each motion step. If inverted elements are predicted by analyzing the trajectories, the time step (displacement of the moving object) is reduced to half. In addition, element shape qualities are improved by edge flip and vertex repositioning. Barral et al. (2014) extended the method proposed by Alauzet. In their method, explicit Inverse Distance Weighted (IDW) interpolation is used instead of the linear elasticity mesh adaptation. In comparison with the method proposed by Alauzet, the number of required motion steps is reduced. For more efficiency and a higher quality of output meshes, Chen et al. (2015) proposed a parallel re-meshing method to remove elements with bad shape generated by mesh adaptation. By these methods, the conformal mesh of each object pose can be generated more efficiently than re-meshing. However, the mesh topology and geometry are globally adapted even if the differences in poses of the objects in motion are very small. In addition, these methods do not deal with contacts of the object meshes. In this paper, we propose a new efficient tetrahedral mesh adaptation method for moving objects with contact. In our method, for efficient mesh adaptation, the mesh adaptation and quality improvement process is applied to only a set of space mesh elements around the moving object. In addition, mesh adaptation methods based on linear elasticity or IWD need to adjust some parameters appropriately. On the other hand, Mean Value Coordinates (MVC) (Ju et al., 2005) which is used for vertex repositioning of some mesh deformation methods (e.g. Maehama et al., 2014a, Sun et al., 2016) does not need to adjust any parameters. Therefore, in our method, for vertex repositioning of mesh adaptation, MVC (Ju et al., 2005) is used instead of linear elasticity and IWD.

In order to create a conformal mesh of two or more contacting object meshes, surface meshes between object meshes have to be identical to each other. Juntunen and Korotov (2013) proposed a post-refinement method for conformal tetrahedral mesh generation from two meshes which are generated individually and contact with each other by planes. In their method, vertices on a mesh are added to the other mesh and intersection points are inserted to both meshes as new vertices in order to make vertices identical in both meshes. A similar procedure is performed in a conformal tetrahedral mesh generation method proposed by Song et al. (2013). In their method, a mesh near the surface is generated by advancing front method and a mesh of the remaining inner part is created by Delaunay triangulation. Next the two meshes are merged. In their method, the geometry of the contacting boundary between two tetrahedral meshes is identical, but redundant vertices are added to the surface of the tetrahedral mesh generated by Delaunay triangulation. Therefore, in the mesh merging process, the redundant vertices are added to the tetrahedral mesh generated by the advancing front method. Wang et al. (2013) proposed a method for linking two dissimilar structured hexahedral meshes. In their methods, contact regions between two hexahedral meshes are triangulated by insertion of vertices to positions of vertices of a mesh, intersecting points of edges of two meshes, and centers of quadrangles. These three methods have the potential for a drastic increase in the number of vertices on the contact regions. Staten et al. (2008) proposed a mesh merging method for two hexahedral meshes in order to create a conformal mesh of assembly models. In their method, surface meshes of contacting objects are modified by some hexahedral dual operations in order to be the same. The quality of the resultant conformal hexahedral meshes is high and the increase in number of elements is moderate. However, the method is not completely automatic. In this paper, we propose an automatic adaptation method for keeping mesh conformity on the contact regions between object meshes while avoiding rapid increase of vertices. In our method, topology and geometry of surface triangular meshes of contacted object meshes are adapted by vertex repositioning and local topological operations.

Many methods of quality improvement for tetrahedral meshes have been proposed (Chen and Holst, 2011, Du and Wang, 2003, Maehama et al. 2014b). In particular, the methods based on Optimal Delaunay Triangulation (ODT) and Centroidal Voronoi Tessellation (CVT) (Chen and Holst, 2011, Du and Wang, 2003, Maehama et al. 2014b) can provide high-quality tetrahedral meshes. The methods based on ODT (ODT smoothing) (Chen and Holst, 2011) improve quality of tetrahedral meshes based on the particular mesh energy. The improvement is done by iterative repositioning of vertices and flip operations. Repositioning is done by moving each vertex to the average of circumcenters of tetrahedral elements including it. Therefore, the computational cost is low. The methods based on CVT (Du and Wang, 2003) improve the quality of tetrahedral meshes by iterative generation of Voronoi Diagram and Delaunay Triangulation using the centroid of the Voronoi regions. In general, ODT smoothing converges faster than the CVT-based method. ODT/CVT-based methods fail to remove slivers (degenerated elements), and methods removing them such as (Klingner and Shewchuk, 2007, Compère et al., 2010) are needed. In our method, an ODT-based quality improvement which consists of removing degenerated and inverted elements, controlling edge length, and improving element shape qualities is used.
3. Mesh adaptation method

3.1 Overview

Inputs of our method are a conformal tetrahedral mesh composed of space mesh $M^S$ and object meshes $M^O_i = \{M^O_i\}$ whose element shape qualities are high and a sequence of 4x4 homogeneous rigid transformation matrices for the object motion $(T_1, T_2, \ldots, T_n)$, which guarantees that objects do not penetrate others. In our method, we assume that the contact region consists of planar, cylindrical, conical, spherical, or torus surfaces which are used often in the design of many mechanical parts. In addition, we assume that contact regions of two object meshes are the same surface type and surface parameters. In this paper, the moving object mesh is described by $M^O_m$ and an object mesh which contacts with $M^O_m$ is described by $M^O_i$.

The overview of our method is shown in the Fig. 2. Our method consists of the following four steps: (A1) mesh segmentation, (A2) region extraction, (A3) mesh adaptation, and (A4) quality improvement.

3.2 Mesh segmentation

For accurate contact detection, surface types and surface parameters of the object meshes are used. In order to calculate surface parameters, a mesh segmentation method proposed by Maehama et al. (2014a) is applied to each object mesh. By this method, the surface of the meshes can be accurately divided into each surface region (a set of triangles representing a surface) which is connected with others by $C^0$ or $C^1$ boundaries. In this method, at first, the surfaces of the object meshes are divided into some planar, cylindrical, conical, spherical, and torus surface regions by region growing based on normal vector, curvatures, and principal direction. Then their surface parameters are calculated by surface fitting based on Levenberg-Marquardt method (Shakarji, 2001). In addition, boundary line segments between surface regions are also extracted and classified into straight lines or conic sections. Finally, parameters of each boundary line segment such as endpoints, radii of circles, and direction vectors of straight lines are calculated as boundary information.

3.3 Region extraction

In our method, two regions, called the “contact region” and “deformed region,” are extracted in each motion step. A contact region is a set of vertices of object meshes, which are on the overlapping area of surface of two object meshes. On the other hand, a deformed region is a set of tetrahedral elements of $M^S$ around the moving object mesh. In our method, for efficient extraction of these two regions, two regular grids (distance field grid $G_d$ and vertex searching grid $G_v$) are used. In this section, we describe how to generate these two grids and extract these two regions.

**Fig. 2** The overview of our mesh adaptation method. Our method consists of the following four steps: (A1) mesh segmentation, (A2) region extraction, (A3) mesh adaptation, and (A4) quality improvement.
generated (see Fig. 3(a)). The $G_d$ represents a distance field from the surface of $M_w^O$, and moves together with $M_w^O$. On the other hand, $G_i$ stores vertices of the given mesh in each cell and is fixed. In the generation of $G_d$, at first, a regular grid which covers the whole of the given mesh is generated. Then, the distance between each cell and the surface of $M_w^O$ is calculated by following Dijkstra’s algorithm as shown in Fig. 3(b).

Step 1: Cells including vertices of $M_w^O$ are identified as “trial cells” and pushed to a trial cell list and their distance values are set to 0. On the other hand, other cells are labeled as “far cells” and their distance values are set to infinity.

Step 2: A cell $c_{\text{min}}$ whose distance value $d_{\text{min}}$ is the minimum in the trial cell list is popped and it is labeled as a “known cell.” Its neighboring far cells are labeled as “trial cells” and pushed to the trial cell list, and their distance values are set to $d_{\text{min}} + 1$. This step is repeated until the distance value of a popped cell is larger than a threshold $t_d$.

Finally, as shown in Fig. 3(c), for efficient calculation, $G_d$ is resized so that $G_d$ consists of cells included in an Axis Aligned Bounding Box (AABB) which includes cells whose distance values are smaller than $t_d$.

On the other hand, in the generation of $G_i$, at first, a regular grid which covers the whole of the given mesh is generated. Then, each vertex of the given mesh is stored in each cell which includes it.

**Contact Region Extraction (Fig. 4)**

In each motion step, if $M_w^O$ contacts with other object meshes, the contact regions are extracted. In this process, $M_w^O$ is first detected using $G_d$. In addition, surface regions overlapping with other surface regions, boundary line segments and sets of vertices on the overlapping area are extracted using surface information. This process consists of the following three steps.

Step 1: The rigid transformation $T_r$ is first applied to $M_w^O$ and $G_d$ (see Fig. 4(a)). All object meshes which have vertices included in cells of $G_d$ whose distance value is 0 are detected as $M_w^O$ which contact with $M_w^O$.

Step 2: Surface regions of $M_w^O$ and $M_v^O$ with the same surface type and surface parameters are extracted as contact surface regions $S_m$ and $S_v$, respectively (see Fig. 4(b)). Boundary line segments $B_0$ of the overlapping area are extracted using boundary information of $S_m$ and $S_v$.

Step 3: Two sets of vertices $V_m \subseteq M_w^O$ and $V_v \subseteq M_v^O$ in the overlapping area are extracted as contact regions. As shown in Fig. 4(c), for accurate extraction of the vertices, the inclusion check of each vertex using $B_0$ is performed on a 2D space defined by surface parameters of $S_m$ and $S_v$, e.g. the development planes for the cylindrical surface regions defined by height along their axis $z$ and width obtained by length of arc $r\theta$.

**Deformed Region Extraction (Fig. 5)**

In each motion step, for efficient mesh adaptation, a set of tetrahedral elements of $M^S$ near $M_w^O$ is extracted as the deformed region using $G_d$ and $G_i$. In this process, at first, the positions of $M_w^O$ and $G_d$ are returned to those before the contact region extraction by performing the inverse transformation of $T_r$. Then, a subset of cells $C_{i1}$ of $G_i$ whose center points are included in the AABB of $G_d$ is extracted (see Fig. 5(a)). Then, distance values are assigned to each cell $c_{i(j,k)} \subseteq C_{i1}$ from each cell of $G_d$ which includes the barycenter of $c_{i(j,k)}$. Next, a set of cells $C_{i2}$ of $G_i$ whose distance

---

Fig. 3 The grid generation: (a) two regular grids (distance field grid $G_d$ and vertex searching grid $G_i$), (b) calculation of distance value of $G_d$ by Dijkstra’s algorithm, and (c) resizing $G_d$. 

is extracted as the deformed region. In the existing methods of mesh adaptation, linear elasticity and IWD are used for adaptation of the are smaller than a threshold. The deformed region extraction: At first, (a) a set of cells \( C_i \) of the vertex searching grid \( G_i \) included in the AABB of the distance field grid \( G_d \) is extracted. Then, (b) a set of cells \( C_{i'} \) of \( G_i \) near the moving object mesh \( M_m^O \) is extracted using distance values given from \( G_i \). Finally, (c) a set of tetrahedra \( T_D \) whose vertices included in cells of \( C_{i'} \) is extracted as the deformed region.

values are lower than a threshold is extracted (see Fig. 5(b)). Finally, a set of tetrahedral elements of \( M^S \) with vertices included in cells \( c_{(i,j,k)} \in C_{i'} \) is extracted as the deformed region \( T_D \) (see Fig. 5(c)).

3.4 Mesh adaptation

The overview of mesh adaptation (Fig. 2, A3) is shown in Fig. 6. This process consists of the following five steps: (i) space mesh elements removal, (ii) rigid transformation, (iii) deformed region adaptation, (iv) contact region adaptation, and (v) re-triangulation. If \( M_m^O \) contacts with other object meshes, all steps are performed. Otherwise, only (ii) rigid transformation and (iii) deformed region adaptation are performed.

(i) Space Mesh Elements Removal

In our method, if \( M_m^O \) contacts with other object meshes, in order to avoid degenerated elements, tetrahedral elements of \( M^S \) near the contact region are removed (see Fig. 7). In this process, at first, tetrahedral elements of \( M^S \) with vertices included in the contact region (i.e. \( v_i \in V_m \cup V_o \)) or their neighboring vertices \( \{v_n\} \) in \( M_m^O \) and \( M_o^O \) are removed, and a hole \( H \) is created. Here, vertices of \( M^S \) on the surface of \( H \) except for vertices of \( M_m^O \) and \( M_o^O \) are denoted by \( v_H \). Then, because \( v_H \) have the possibility of penetrating \( M_m^O \) after motion, tetrahedral elements of \( M^S \) with \( v_H \) whose distances from the surface of \( M_m^O \) are smaller than a threshold \( r_{th} \) are removed. The second type of removal is repeated until the distances between all \( v_H \) and the surface of \( M_m^O \) are larger than a threshold.

(ii) Rigid Transformation and (iii) Deformed Region Adaptation

In our method, \( M_m^O \) is moved by the rigid transformation and inner vertices \( V_{ID} \) of \( T_D \) are moved according to the motion of \( M_m^O \). In the existing methods of mesh adaptation, linear elasticity and IWD are used for adaptation of the position of vertices in \( M^S \) according to the motion of the objects. However, some parameters need to be adjusted appropriately. In our methods, for simple calculation, we use a space embedding method based on Mean Value Coordinates (MVC) (Ju et al., 2005) similar to Maehama et al. (2014a) which does not need to adjust any parameters.

At first, the position \( p_i \) of each vertex of \( V_{ID} \) is represented by a linear combination of its neighboring vertices \( H_i \) as shown in Eq. (1) (See Fig. 8);
The weights are calculated as MVC for a polyhedron consisting of $H_i$. After the movement of $O_m$, the positions of $V_i$ are derived as a solution of the following system of linear equations;

$$p_i = \sum_{j \in H_i} \omega_{ij} p_j,$$  \hspace{1cm} (1)

$$p_i = \sum_{j \in H_i} \omega_{ij} p_j = \sum_{k \in H_i^k} \omega_{ik} p_k \quad (i \in V_{id})$$  \hspace{1cm} (2)

where $H_i^F$ is a set of vertices of fixed boundary of $T_D$ and $M_o^F$ in $H_i$, whose positions are known after movement of $M_o^F$. $H_i^k$ is $H_i - H_i^F$ (See Fig. 8). The derived positions $p_i$ preserve relative location to neighbors before modification as much as possible.
(iv) Contact Region Adaptation

In our method, to maintain mesh conformity on the contact regions between object meshes without rapid increase of vertices, the topology and geometry of surface triangular meshes in the contact regions of $M^o_m$ and $M^o_c$ are adapted by vertex repositioning and local topological operations. In this process, at first, the boundary shape (lines) of the overlapping area is represented by edges of $M^o_m$ and $M^o_c$. Secondly, positions and numbers of $V_m$ and $V_c$ are equalized. Then, connectivity of surface triangular meshes on the contact regions of $M^o_m$ and $M^o_c$ is made identical. Finally, the element shape qualities of $M^o_m$ and $M^o_c$ are improved.

In order to represent $B_O$ accurately, the intersection points $P_i$ of boundary line segments of the contact regions are needed as vertices of $M^o_m$ and $M^o_c$. However, there may be no vertices on the location of the intersection points. Therefore, as shown in Fig. 9, vertices $v_i$ located on the intersecting points are first inserted to $M^o_m$ and $M^o_c$ by edge split.

After that, the positions and number of vertices in $V_m$ and $V_c$ are equalized by vertex repositioning, edge split, and face split. As shown in Fig. 10(a), at first, corresponding vertex pairs are extracted from $V_m$ and $V_c$. In this process, each vertex $v \in V_m \cup V_c$ is first labeled as a “non-corresponding vertex”, and if a non-corresponding vertex $v'_i \in V_c$ is the closest vertex of a non-corresponding vertex $v''_i \in V_m$ and $v''_i$ is the closest vertex of $v'_i$, they become “corresponding vertices” of each other. Then, vertices are classified into “movable” or “un-movable” vertices. The movable vertices guarantee that their movement does not generate inverted elements, and detail of this classification process is described later. Next, as shown in Fig. 10(b), all vertices in $V_m$ and $V_c$ except for un-movable vertices move to positions of their corresponding vertices. The vertex classification and the vertex repositioning are repeated until no vertices move. After that, as shown in Fig. 10(c), vertices are inserted to the position of non-corresponding vertices and un-movable vertices by edge split and face split. In the vertex classification process, signed volumes of tetrahedral elements are used for finding “movable” vertices. If signed volumes of neighboring tetrahedral elements of vertex $i$ are always positive under the possible movements of the vertices of the tetrahedron, the vertex $i$ can move without causing any inverted elements. As shown in Fig. 11, for each tetrahedron with vertices in $V_m$ and $V_c$, its signed volume for at most 15 patterns of the vertex movement are calculated, and if the signed volume becomes a negative value, vertices repositioned in the pattern are labeled as “un-movable.” In the process, at first, all vertices in $V_m$ and $V_c$ are labeled as “movable,” and the labeling of un-movable vertices is applied to all tetrahedral elements including vertices in the $V_m$ and $V_c$.

After the equalization of positions and number of vertices, the connectivity of surface triangular meshes $F_m$ and $F_c$ on the contact regions of $M^o_m$ and $M^o_c$ are made identical to each other by edge swap and edge split. In this paper, an edge whose two endpoint positions are same as those of an edge $e$ is called a “corresponding edge” of $e$, and if the corresponding edge of $e$ does not exist, $e$ is called a “non-corresponding edge.” In addition, an edge obtained by swap of $e$ is denoted by $e'$, and edges which can be swapped are called “swappable” edges. In this process, at first, as shown in Fig. 12(a), for non-corresponding swappable edge $e$, if $e'$ is a corresponding edge of a non-corresponding edge, $e$ is swapped. If there are no such edges but the connectivity of $F_m$ and $F_c$ are not the same, as shown in Fig. 12(b), other non-corresponding swappable edges are swapped. These two swapping processes are repeated until any non-corresponding edge cannot be swapped or the iteration times reached a threshold. Finally, as shown in Fig. 12(c), if any non-corresponding edge cannot be swapped on $F_m$ and $F_c$, vertices located on the intersection points between two non-corresponding edges are inserted to $M^o_m$ and $M^o_c$ by edge split.

After that, in order to improve element shape qualities, phased ODT smoothing (Maehama et al., 2014b) is applied to the set of tetrahedral elements for which one or more vertices are included in $V_m$ or $V_c$.
Fig. 10 The equalization of positions and numbers of vertices (two meshes contact with each other on the bold line): (a) extraction of corresponding vertex pairs, (b) vertex repositioning, and (c) vertex insertion.

$V_{c}, V_m$ : Sets of Vertices in the Contact Region

Fig. 11 The movable check of vertices. For each tetrahedron with vertices in the contact region, its signed volumes $V(x_i, x_j, x_k, x_l)$ are determined. When the signed volume becomes a negative value, vertices repositioned in the pattern are labeled as “unmovable.”

$V^{*}$ is position of a vertex for at most 15 patterns of the vertex movement are calculated. If the signed volume becomes a negative value, vertices repositioned in the pattern are labeled as “unmovable.”

Fig. 12 The equalization of connectivity of surface triangular meshes. (a) An edge $e$ of the moving object mesh $M_a^o$ (or the contact object mesh $M_a^c$) whose swapped edge $e'$ has same two endpoints of an edge of $M_a^o$ (or $M_a^c$) is swapped. (b) Edges which do not have the correspondence are selected randomly and swapped. (c) Vertex located on an insertion point of two edges is inserted by edge split.

(v) Re-triangulation

After mesh adaptation of the contact region, if $M_a^o$ contacts with other object meshes, the hole $H$ generated by process (i) is triangulated by the constrained Delaunay Triangulation. A set of tetrahedral elements generated by this triangulation is denoted by $T_H$ in this paper.

3.5 Quality improvement

After mesh adaptation, many tetrahedral elements with bad shapes are included in the resulting conformal mesh. Therefore, element shape qualities are improved by a quality improvement based on Optimal Delaunay Triangulation (ODT). In this paper, stretch (Baker, 1989) is used as an element shape quality measure. The stretch $s(t)$ of a tetrahedron $t$ is defined as Eq. (3):

$$s(t) = \frac{6\sqrt{G(t)}}{\max_{e \in E_t} l(e)}$$  \hspace{1cm} (3)

where $V(t)$ is the signed volume of $t$, $S(t)$ the surface area of $t$, $E_t$ a set of edges of $t$, and $l(e)$ is the length of edge $e$. The stretch is 1 for a regular tetrahedron, and the stretch becomes smaller for a distorted tetrahedron. It becomes 0 for a degenerated tetrahedron and becomes a negative value for an inverted tetrahedron.

In addition, in order to obtain output meshes having desirable mesh density, the edge length of $M^o$ is controlled by using $G_e$. In our method, the range of target edge lengths is defined by minimum and maximum edge length $r_{\text{min}}$ and $r_{\text{max}}$ at each cell in the $G_e$. If keeping the edge length of the original input mesh is required, $r_{\text{min}}$ and $r_{\text{max}}$ are determined.
according to the original edge length. In our implementation, for each cell, \( \tau_{\text{min}} = 0.75 \times \tau_{\text{ave}} \) and \( \tau_{\text{max}} = 1.5 \times \tau_{\text{ave}} \) where \( \tau_{\text{ave}} \) is the average length of original input mesh edges whose midpoints are included in the cell are used. If meshes which become coarser at the portion far from the object are required, \( \tau_{\text{min}} \) and \( \tau_{\text{max}} \) are assigned according to the distance from the surface of object meshes.

In our method, at first, a set of tetrahedral elements \( T_Q \) of \( M^S \) which satisfy any of the following conditions is extracted from \( T_D \) and \( T_H \). 

Condition 1: the stretch is smaller than a threshold \( \tau_{s1} \).  
Condition 2: the minimum or maximum edge length is out of the range of target edge lengths of cells including their midpoints.

Then, a quality improvement consisting of the following four steps is applied to \( T_Q \). 

Step 1: Tetrahedral elements whose stretches are smaller than a threshold \( \tau_{s2} < \tau_{s1} \) (i.e. inverted and degenerated elements) are removed by edge collapse, double edge split collapse and face collapse (Compère et al., 2010, Maehama et al., 2014b).

Step 2: Each edge shorter than the threshold \( \tau_{\text{min}} \) of a cell including its midpoint is collapsed by edge collapse.

Step 3: Each edge longer than the threshold \( \tau_{\text{max}} \) of a cell including its midpoint is divided into two edges by edge split.

Step 4: ODT smoothing (Chen and Holst, 2011) is applied to \( T_Q \). For keeping mesh conformity, surface vertices and triangles of each object mesh are handled as boundary in the smoothing.

4. Results

In this section, we first show an application of our method to the motion of a single object mesh in a space mesh. In this experiment, a mesh of a rectangular parallelepiped is rotated and translated. Then, in order to show that our method can deal with object motion with contact, an experiment was carried out using object meshes of a moving cylinder and a half tube whose radii are the same. In this experiment, the cylinder was translated toward a half tube and contacted with the half tube. Finally, it is shown that the contact region adaptation of our method enables us to generate conformal tetrahedral meshes of moving assembly models from a set of meshes generated individually without space mesh. Table 1 shows the data of each experiment. In the all of our experiments, we used 0.2 and 0.05 as \( \tau_{s1} \) and \( \tau_{s2} \), respectively. All experiments were performed by a PC (CPU: Core i7 3.00GHz, RAM: 64GB).

Translation of a rotating rectangular parallelepiped (Fig. 13)

The original mesh model (\#vertices: 37629 and \#tetrahedra: 210465) is shown in Fig. 13(a). The red rectangular parallelepiped moved 0.1 toward the right side along \( z \) axis and rotated by 1 degree around an axis which passes a barycenter of the rectangular parallelepiped and is parallel to \( x \) axis in each motion step. The total number of motion steps is 360. The minimum average stretch of all output meshes was almost the same as the average stretch of the input mesh. The minimum stretch of tetrahedral mesh in each motion step and the cross-section views of output meshes are shown in Fig. 13(b) and (c), respectively. The minimum stretch of all output meshes is 0.13, and it is larger than the recommended lower limit (0.05) of stretch. The generation of all meshes took 53 s. Therefore, in each motion step, the conformal tetrahedral mesh without any inverted elements could be generated within only 0.15 s. As shown in Fig. 13(c), space mesh could be adapted depending on the motion of the rectangular parallelepiped, and the adaptation was done only around the rectangular parallelepiped.

Translation of a cylinder toward half tube (Fig. 14)

The other example (\#vertices: 31530 and \#tetrahedra: 160791 in the original mesh) is shown in Fig. 14(a), where an object contacts with another. In this experiments, the red cylinder moved 1 toward the green half tube along its axis in each motion step. The total number of motion steps is 45. The minimum average stretch of all output meshes was almost the same as the average stretch of the input mesh. The minimum stretch of tetrahedral mesh in each motion step and the every 5 step output meshes after 30 steps are shown in Fig. 14(b) and (c), respectively. The minimum stretch of all output meshes is 0.019. Tetrahedral elements with stretches lower than 0.05 were observed in only two motion steps. The output mesh of the 35th step has 6 such elements on the boundary of \( T_H \) with two object meshes. Therefore, the quality improvement described in section 3.5 should be made more efficient for removing such elements in future works. As a result, the total calculation time was 220 s. In each motion step, the conformal tetrahedral mesh without any inverted elements could be generated in about 5 s. The generation time of the input mesh by TetGen (Si, 2006) was
about 10 s, and a quality improvement based on ODT was performed after that. As shown in Fig. 14(c), all edges on the boundary between two object meshes were shared by two object meshes, and all triangles between two object meshes were also shared by two tetrahedral elements of two object meshes. The final output mesh has 34353 vertices, which means that vertices increased by about 9 percent. These results show that the tetrahedral mesh was adapted to the object motion with contact, and conformal tetrahedral meshes can be generated more efficiently by our method.

Table 1  Information of motion and meshes, element shape qualities of meshes, and generation time of all output meshes

<table>
<thead>
<tr>
<th>Figure</th>
<th>#Motion steps</th>
<th>#Vertices</th>
<th>#Tetrahedra</th>
<th>Minimum stretch</th>
<th>Average stretch</th>
<th>Generation time of all meshes [s]</th>
</tr>
</thead>
<tbody>
<tr>
<td>Fig. 13</td>
<td>360</td>
<td>37629</td>
<td>210465</td>
<td>0.28</td>
<td>0.13</td>
<td>53</td>
</tr>
<tr>
<td>Fig. 14</td>
<td>45</td>
<td>31530</td>
<td>160791</td>
<td>0.15</td>
<td>0.019</td>
<td>220</td>
</tr>
<tr>
<td>Fig. 15</td>
<td>36</td>
<td>8521</td>
<td>34445</td>
<td>0.28</td>
<td>0.004</td>
<td>97</td>
</tr>
</tbody>
</table>

Fig. 13 Translation of a rotating rectangular parallelepiped: (a) input mesh, (b) minimum stretches of output meshes, and (c) cross-section of output meshes.

Fig. 14 Translation of a cylinder toward a half tube: (a) input mesh, (b) minimum stretches of output meshes, and (c) cross-section of output meshes.
Finally, we show that conformal tetrahedral meshes of moving assembly models can be generated by our method from object meshes generated individually without space mesh. The input simple assembly mesh (#vertices: 8521 and #tetrahedra: 34445) whose interfaces are non-conforming is shown in Fig. 15. The green part (crank shaft) is rotated 10 degrees in each motion step around an axis (shown by a black arrow in Fig. 15(a)) and the blue part (piston head) can only translate along the orange arrow. The total number of motion steps is 36. The average stretch of the input mesh was 0.638 and the minimum average stretch of all output meshes was 0.622. The minimum stretch of each motion step is shown in Fig. 15(b). The minimum stretch of all output meshes is 0.004. Tetrahedral elements with stretches lower than the recommended lower limit were observed in only three motion steps and the number of such elements is at most 4. Such elements appeared in the contact region. Therefore, we will consider a method for the contact region adaptation without such elements in our future works. The output meshes of the first step, 12th step, 24th step, and 36th step are shown in Fig. 15(c). As shown in Fig. 15(c), triangles between two object meshes were shared by at most two tetrahedral elements of two object meshes. The total generation time of the input meshes by Octree method was about 3.3 s, and after that ODT smoothing was performed. On the other hand, the average generation time of the conformal mesh at each motion step is 2.7 s, and it means that our method could reduce about 20 % of the mesh generation time. Although the element shape qualities of output meshes in some motion steps are smaller than the recommended lower limit of stretch, the result shows our method can be used for efficient generation of conformal meshes without inverted elements from a set of meshes which are generated individually.

6. Conclusion

A mesh adaptation method consisting of the mesh segmentation, the region extraction, the mesh adaptation, and the quality improvement was proposed. In our method, for efficient mesh adaptation, the mesh adaptation process was applied to only a set of space mesh elements around the moving object. Moreover, contacts between object meshes were accurately detected using surface parameters, and for keeping mesh conformity on the contact regions between object meshes, the topology and geometry of surface triangular meshes of contacting object meshes were adapted by vertex repositioning and local topological operations. In addition, in order to obtain a high quality mesh in each motion step, shape qualities of tetrahedral elements were improved by using a method based on Optimal Delaunay Triangulation (ODT).

The effectiveness of our method was demonstrated through three simple experiments. At first, our method applied to a mesh with the motion of a single object mesh in a space mesh, and it was shown that space mesh could be modified depending on the motion of the object mesh, and the adaptation was done only around the moving object mesh. Then, we applied our method to a mesh which consisted of a moving cylinder, a fixed half tube, and a space mesh around them in order to show that our method can deal with contact between object meshes. As a result, in each motion step, a conformal tetrahedral mesh without any inverted elements could be generated. In addition, we showed that our method can be used for generation of conformal tetrahedral meshes of moving assembly models from a set of meshes which are
generated individually. In our experiments, the processing time of each motion step is at most 5 s for meshes including about 160k tetrahedral elements. In future works, we will develop a quality improvement method in order to obtain higher quality meshes.

Acknowledgement

This work was supported by JSPS KAKENHI Grant Number 15J01584.

References