Efficient Hexahedral Mesh Generation for Complex Geometries Using an Improved Set of Refinement Templates

Yasushi Ito, Alan M. Shih, and Bharat K. Soni

Enabling Technology Laboratory
Department of Mechanical Engineering
University of Alabama at Birmingham, Birmingham, AL, U.S.A.
{yito,ashih,bsoni}@uab.edu

Abstract. We proposed a new, easy-to-implement, easy-to-understand set of refinement templates in our previous paper to create geometry-adapted all-hexahedral meshes easily [Ito *et al.*, "Octree-Based Reasonable-Quality Hexahedral Mesh Generation Using a New Set of Refinement Templates," *International Journal for Numerical Methods in Engineering*, Vol. 77, Issue 13, March 2009, pp. 1809-1833, DOI: 10.1002/nme.2470]. This paper offers an extension of the template set to refine concave domains more efficiently. In addition, two new options, temporal rotation and temporal local inflation, are introduced to create hexahedral meshes with fewer elements for long objects or objects with thin regions. Several examples are shown to discuss the improvements of the mesh generation method.

1 Introduction

All-hexahedral mesh generation using an octree data structure is a promising approach to automatically create meshes for complex geometries. However, there are still several issues that need to be addressed for practical applications.

The refinement of a hexahedral mesh with a concave refinement domain is a challenging problem because a resulting mesh must have only hexahedral elements without hanging nodes and the number of elements to be added must be minimized outside of the refinement domain. Several solutions have been presented before, using a combination of templates, and/or pillowing (also known as sheet insertion) techniques [1-7]. Schneiders *et al*. proposed the first template-based method in 1996, which has been used widely because of its simplicity to implement [1]. However, it can be applied to convex refinement domains only. To refine concave refinement domains, many extra elements are needed outside of the domains. Although other template-based methods can be applied to concave refinement domains more efficiently, their implementation is neither easy nor clear, especially for multiple levels of refinement [2, 3]. The pillowing methods are not easy to implement because generally sheets cannot be added locally.

To overcome this problem, we have recently proposed a new, easy-to-implement, easy-to-understand set of refinement templates as part of octree-based,

geometry-adapted hexahedral mesh generation [8]. Two new templates were developed for three-node (three nodes of one of the six quadrilaterals) and two-face (two quadrilaterals sharing an edge) refinement. The new template set allows refining hexahedra individually and concave refinement domains can be refined efficiently. In this paper, we propose adding one more new template to the refinement template set to improve the efficiency.

Another challenging problem is how to keep the number of hexahedra minimum for complex geometries while their geometrical fidelity is kept. The alignment of hexahedra in an octree-based mesh mostly depends on the position of the initial octree. The easiest way to select it is by the use of an axial aligned bounding box around the input geometry. However, it tends to create more elements for long objects that do not align with the *xyz* axes. Objects with thin regions, such as skulls and hearts, need many elements to be represented by octree-based hexahedra without holes. We propose two mesh generation options to relieve this problem.

The reminder of the paper is organized as follows. In Section 2, the new template set is introduced. In Section 3, new options added to our hexahedral mesh generator are introduced to create meshes with fewer elements for complex geometries. In Section 4, examples are shown to demonstrate our mesh generation method. Finally, conclusions are given in Section 5.

2 Refinement Templates

Figure 1 shows the new set of templates for edge (Figure 1a), face (Figure 1b), volume (Figure 1c), three-node (Figure 1d), two-face (Figure 1e) and opposite-edge (Figure 1f) refinement. The templates for edge, face, volume, three-node, two-face and opposite-edge refinement are represented as $T_2(\underline{AB}CDEFGH)$, $T_4(\underline{ABCD}EFGH)$, $T_8(\underline{ABCDEFGH})$, $T_3(\underline{AB}C\underline{D}EFGH)$, $T_6(\underline{AB}C\underline{DEF}G\underline{H})$, and $T_{4o}(\underline{AB}CDEF\underline{GH})$, respectively. The underlines indicate nodes to be refined, which correspond to the black points with white letters in Figure 1 (hereafter referred as marked nodes). Only the main structures of T_3, T_6 and T_{4o} are shown in Figure 1 to simplify the drawing. T_2, T_4 and T_8 are from pillowing [4, 5]. T_3 and T_6 are proposed in [8]. T_{4o} is a new template proposed in this paper.

To obtain T_3, the hexahedron is first divided into six base hexahedra as shown in Figure 1d. T_2 is then applied to two hexahedra: $T_2(\underline{AB}JIEFNM)$ and $T_2(\underline{DA}ILHEMP)$. The other four hexahedra, *CDLKGHPO*, *BCKJFGON*, *IJKLMNOP* and *MNOPEFGH*, are used as is.

To obtain T_6, the hexahedron is first divided into five base hexahedra as shown in Figure 1e. T_2 is applied to two hexahedra: $T_2(\underline{BF}GCJNOK)$ and $T_2(\underline{HD}CGPLKO)$. T_4 is applied to two other hexahedra: $T_4(\underline{AEFB}IMNJ)$ and $T_4(\underline{DHEA}LPMI)$. The hexahedron in the center, *IJKLMNOP*, is used as is.

To obtain T_{4o}, the hexahedron is first divided into seven base hexahedra as shown in Figure 1f. T_2 is applied to four hexahedra: $T_2(\underline{AB}JIEFNM)$, $T_2(\underline{AB}CDIJKL)$, $T_2(\underline{GH}POFEMN)$ and $T_2(\underline{GH}DCOPLK)$. The hexahedron in the center, *IJKLMNOP*, is used as is.

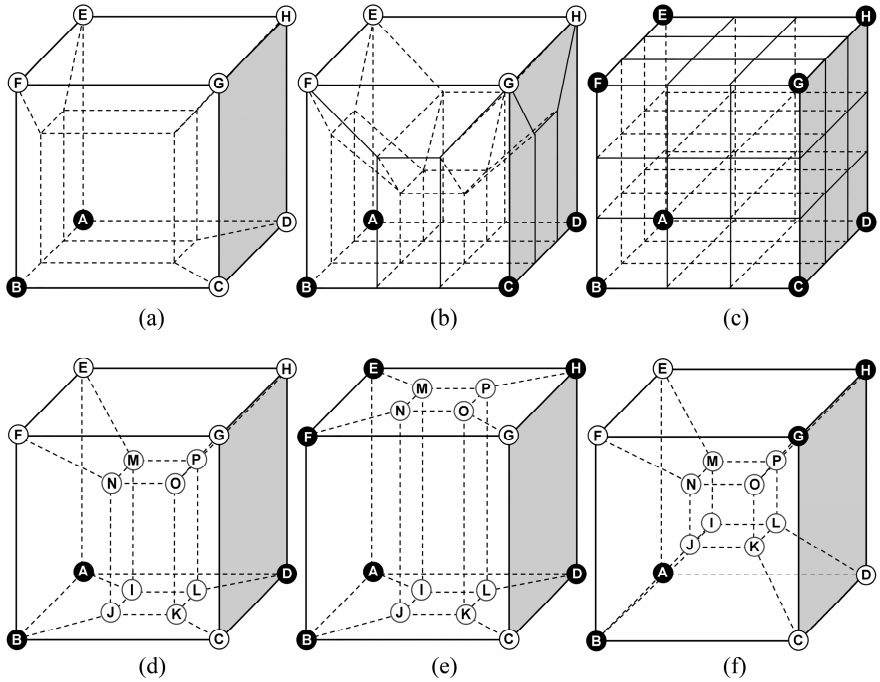


Fig. 1. New set of refinement templates for (a) edge (T_2), (b) face (T_4), (c) volume (T_8), (d) three-node (T_3), (e) two-face (T_6) and (f) opposite-edge refinement (T_{4o}). Note that d, e and f only show the main structures of our new templates.

Table 1. Number of hexahedra in each of the six refinement templates

Templates	T_2	T_4	T_8	T_3	T_6	T_{4o}
# of hexahedra	5	13	27	14	37	23

We can consider that those three templates, T_3, T_6 and T_{4o}, are created by multiple application of sheet insertion. Table 1 shows the number of hexahedra in each of the six templates. Table 2 shows a conversion table for possible refinement patterns and corresponding refinement templates. This conversion table is based on Candidate 2 in [8] to use the new opposite-edge refinement template. Let us think about default templates for quadrilaterals to understand the conversion table better. If a quadrilateral has only one marked node (Figure 2a) or two marked nodes that are in opposite corners (Figure 2b), it is not refined. If a quadrilateral has two neighboring marked nodes (Figure 2c), three marked nodes (Figure 2d) or four marked nodes (Figure 2e), it is refined using $T_{Q2}(\underline{AB}CD)$, $T_{Q3}(\underline{AB}C\underline{D})$ or $T_{Q4}(\underline{ABCD})$, respectively. Patterns 1, 2B, 2C, 3C and 4F shown in Table 2 are not refined, and one of the marked nodes in Pattern 3B or 4D is ignored, accordingly. Although the marked nodes are not always refined, the conversion table works

Table 2. Refinement patterns and corresponding refinement templates

Patterns	Template used
1, 2B, 2C, 3C, 4F	N/A
2A, 3B	T_2
3A, 4D	T_3
4A	T_4
4E	T_{4o}
4B, 4C, 5A, 5B, 6A	T_6
5C, 6B, 6C, 7, 8	T_8

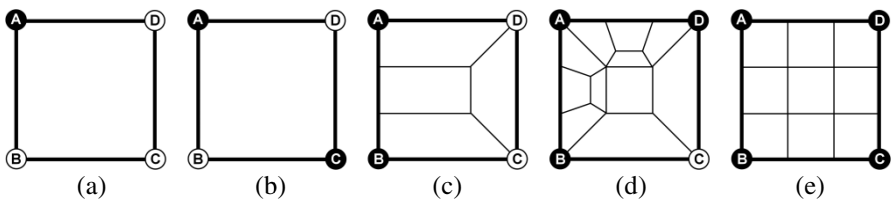


Fig. 2. Quadrilateral refinement templates for (a) node, (b) opposite-node, (c) edge (T_{Q2}), (d) three-node (T_{Q3}) and (e) full refinement (T_{Q4})

well most of the time because those refinement patterns appear only on the verge of the domain to be refined.

All the templates need new nodes. Their positions must be determined consistently and easily such that subdivided hexahedra are in good quality. Note that initial hexahedra are not necessarily cubes but can have low minimum scaled Jacobian values, *e.g.*, the base hexahedra in T_3, T_6 and T_{4o} to which T_2 or T_4 is applied. Let us think about how to calculate the positions of the new nodes in T_8 because those in the other templates are calculated in the same way. Let $\mathbf{x}_i$ be the coordinates of node *i*. Two nodes added between any edge are calculated as its trisection points. For example, the two nodes between edge *AB* are obtained as $\frac{2\mathbf{x}_A+\mathbf{x}_B}{3}$ and $\frac{\mathbf{x}_A+2\mathbf{x}_B}{3}$. Each of four nodes added on any quadrilateral is calculated as the center of its any three nodes. For example, the four nodes on quadrilateral *ABCD* are obtained as $\frac{\mathbf{x}_A+\mathbf{x}_B+\mathbf{x}_C}{3}$, $\frac{\mathbf{x}_B+\mathbf{x}_C+\mathbf{x}_D}{3}$, $\frac{\mathbf{x}_C+\mathbf{x}_D+\mathbf{x}_A}{3}$ and $\frac{\mathbf{x}_D+\mathbf{x}_A+\mathbf{x}_B}{3}$. Eight nodes added inside T_8 are calculated as the trisection points of the four diagonal lines: *AG*, *BH*, *CE* and *DF*.

3 Options for Hexahedral Mesh Generation

To create hexahedral meshes efficiently for complex geometries, the following two options have been added to our mesh generation method: temporal rotation and temporal local inflation.

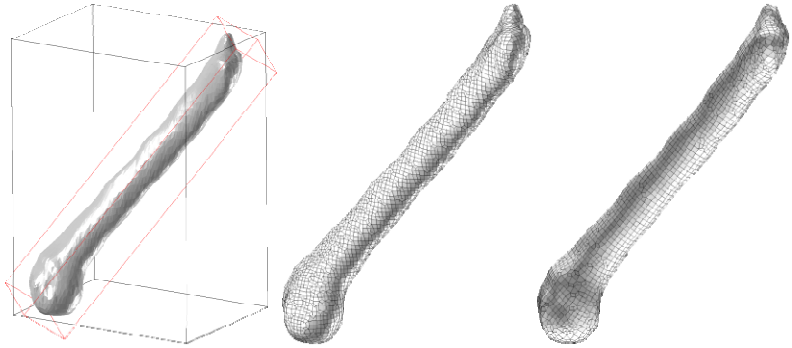
3.1 Temporal Rotation

Surface models are not always axially aligned, *i.e.*, their oriented bounding boxes (OBB) do not always align with the *xyz* axes. This means that the coordinate system of an octree needs to be selected carefully. The temporal rotation option is useful to create a mesh with fewer elements with the same resolution for a long object.

Figure 3 shows an example of a pediatric left humerus model. There are two bounding boxes in Figure 3a: the axial aligned bounding box (AABB; black) and the oriented bounding box (OBB; red) calculated based on the covariance matrix from vertex coordinates [9]. Our mesh generation method creates octants aligned with the AABB by default (Figure 3b). When the temporal rotation option is turned on, octants are automatically aligned with the OBB (Figure 3c). Fewer elements are needed with the same resolution. The quality of the meshes is almost the same. The lowest minimum scaled Jacobian of the hexahedra in the two meshes is 0.37. No adaptation is applied in this case.

3.2 Temporal Local Inflation

The temporal local inflation is a helpful option to represent thin regions with a fewer number of hexahedra. Suppose the local thickness of a surface model is τ_l



(a) (b) (c)

Fig. 3. Hexahedral meshes for a pediatric left humerus: (a) Surface model with AABB (black) and OBB (red); (b) Hexahedral mesh based on AABB (10.5K hexahedra); (c) Hexahedral mesh based on OBB (6.5K hexahedra)

and the local size of the octants (cubes) is ε_l. To represent the thin region without holes, $\varepsilon_l < k\tau_l$ $(k > 2)$) because the octants intersecting with the surface model are removed in our approach during the octree-based core mesh generation process. k is a constant, and the default value is 3.5. If $\varepsilon_l > k\tau_l$, the surface is locally inflated in the local outwardly normal direction so that the thickness temporarily becomes greater than $k\tau_l$. The inflation method is similar to the one used in near-field mesh generation for viscous flow simulations because the local outwardly normal direction and the moving distance may need to be slightly changed during the inflation to avoid creating intersections of the surface [10, 11]. Note that the initial ε_l should not be too big to represent certain details of the input geometry.

Figure 4 illustrates an example of a pediatric skull model. Figure 4a shows the original triangulated surface model. The size of octants can be automatically adjusted based on local thickness of the skull [12] and other user-specified parameters [8] during the octree-based hexahedral mesh generation process. Without the temporal local inflation option, the user needs to specify the parameters so that the size of leaf octants can be small enough to resolve all the thin regions. This induces two problems. If the size of the smallest leaf octants is too small, the resulting mesh may have too many hexahedra to perform computational simulations in limited computational resources (Figure 4b). If the size of the smallest leaf octants is not small enough, the resulting hexahedral mesh can have many holes (Figure 4c).

The temporal local inflation option resolves these problems at some level. When this option is turned on, octants are created inside the surface model after it is temporarily inflated like a balloon only at thin regions as shown in Figure 4d (*cf.* the nose and teeth in Figures 4a and d). The boundary surface of the hexahedral mesh is then recovered based on the original triangulated surface as usual. The resulting mesh shown in Figure 4d has 649K hexahedra. Although the dense mesh created with the temporal local inflation option off (Figure 4d) has 3.08M hexahedra, more elements are needed because there are still holes, especially around the eye sockets, where bones are very thin.

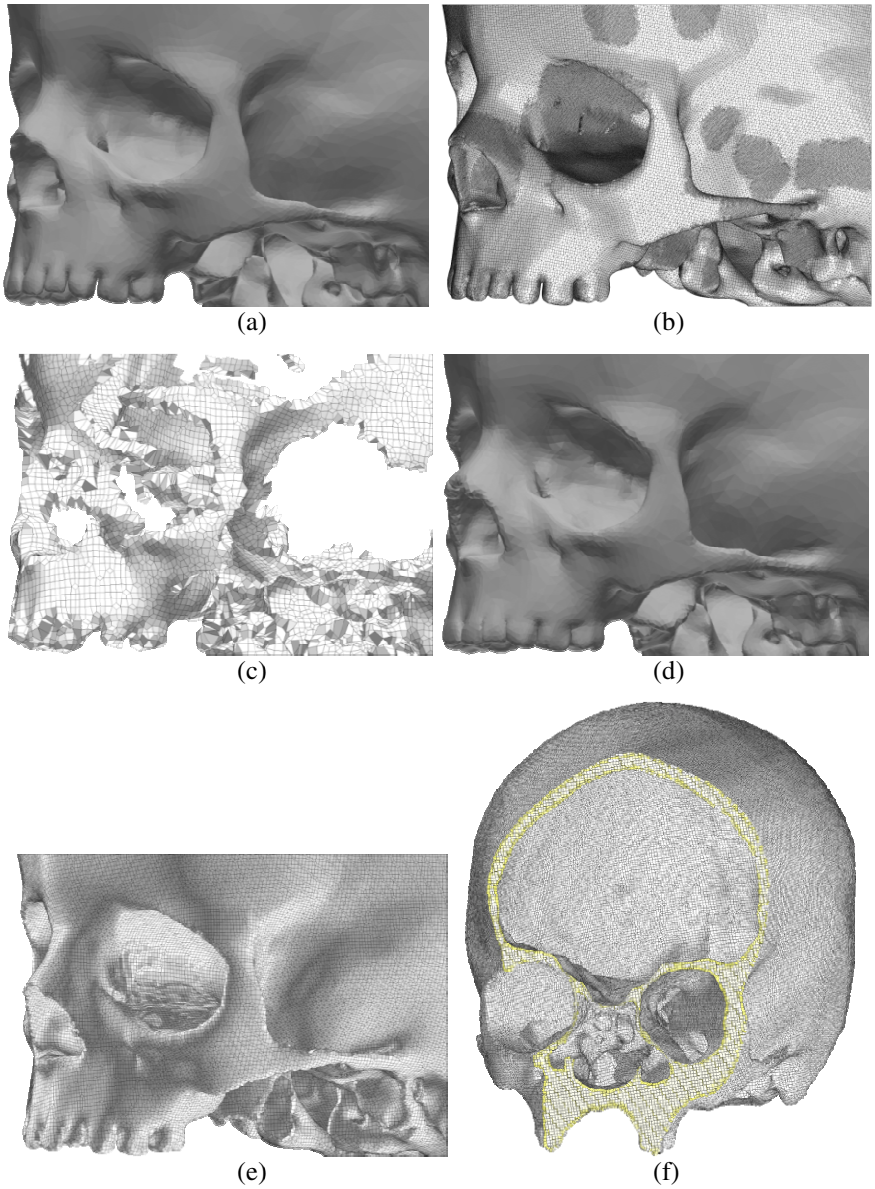


Fig. 4. Hexahedral mesh generation for a pediatric skull model: (a) Original surface model; (b-c) Hexahedral mesh with the temporal local inflation option off – two and three times of local refinement in b and c, respectively; (d) Locally inflated surface model; (e) Hexahedral mesh with the temporal local inflation option on (two times of local refinement); (f) Another view of e with a cross-section (yellow)

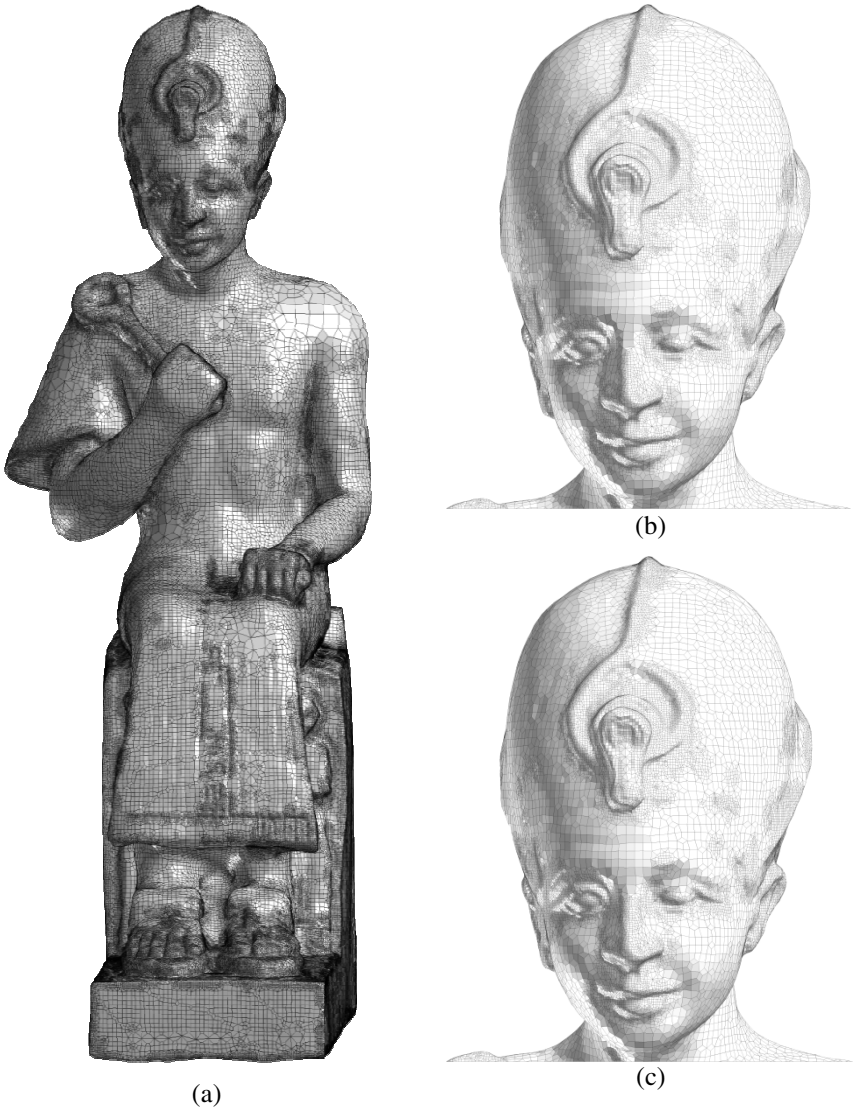


Fig. 5. Hexahedral mesh for a Ramses model (a-b) using the new set of the refinement templates (2.14M elements) and (c) the previous set (2.20M elements)

4 Applications

In this section, four geometries are shown to demonstrate our hexahedral mesh generation capability. The quality of hexahedral meshes is also discussed based on the scaled Jacobian metric.

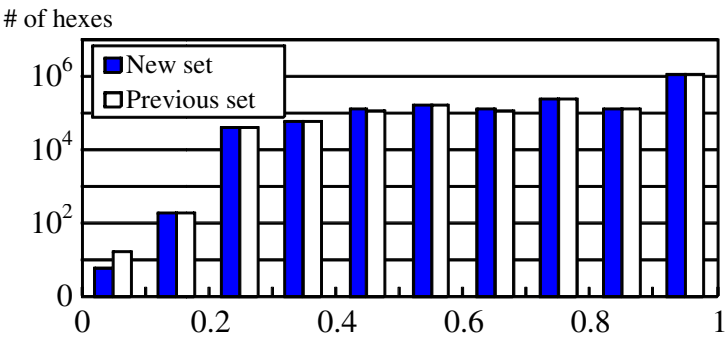


Fig. 6. Element quality distribution by the scaled Jacobian metric for the Ramses meshes

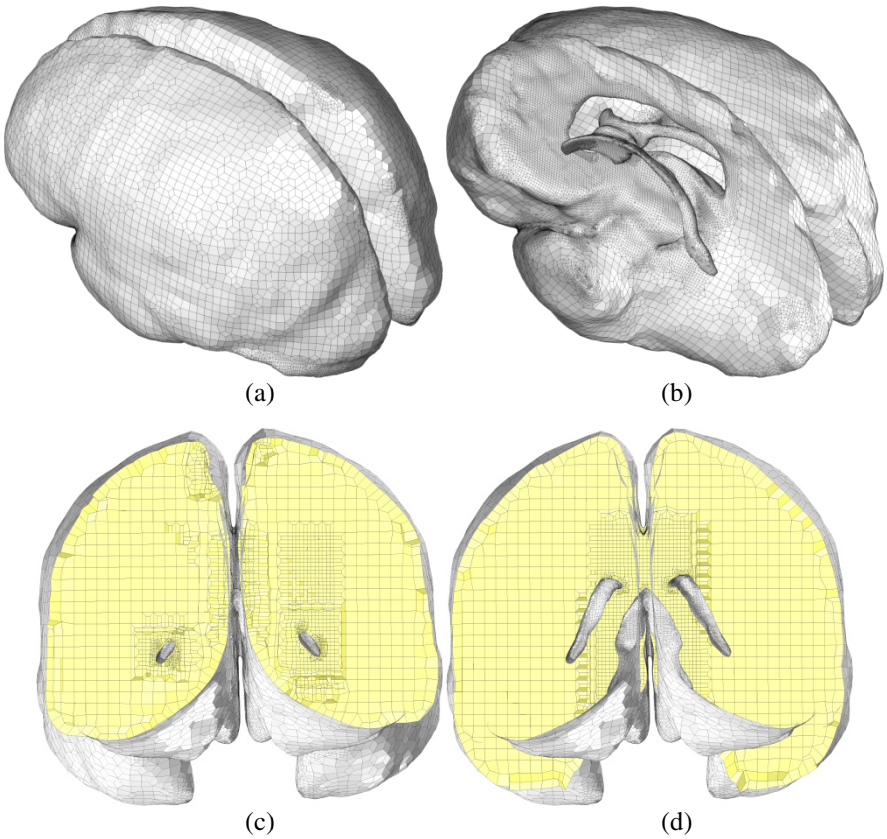


Fig. 7. Hexahedral mesh for a pediatric brain model: (a-b) Front and back sides of surface mesh; (c-d) Cross-sections of the mesh (yellow)

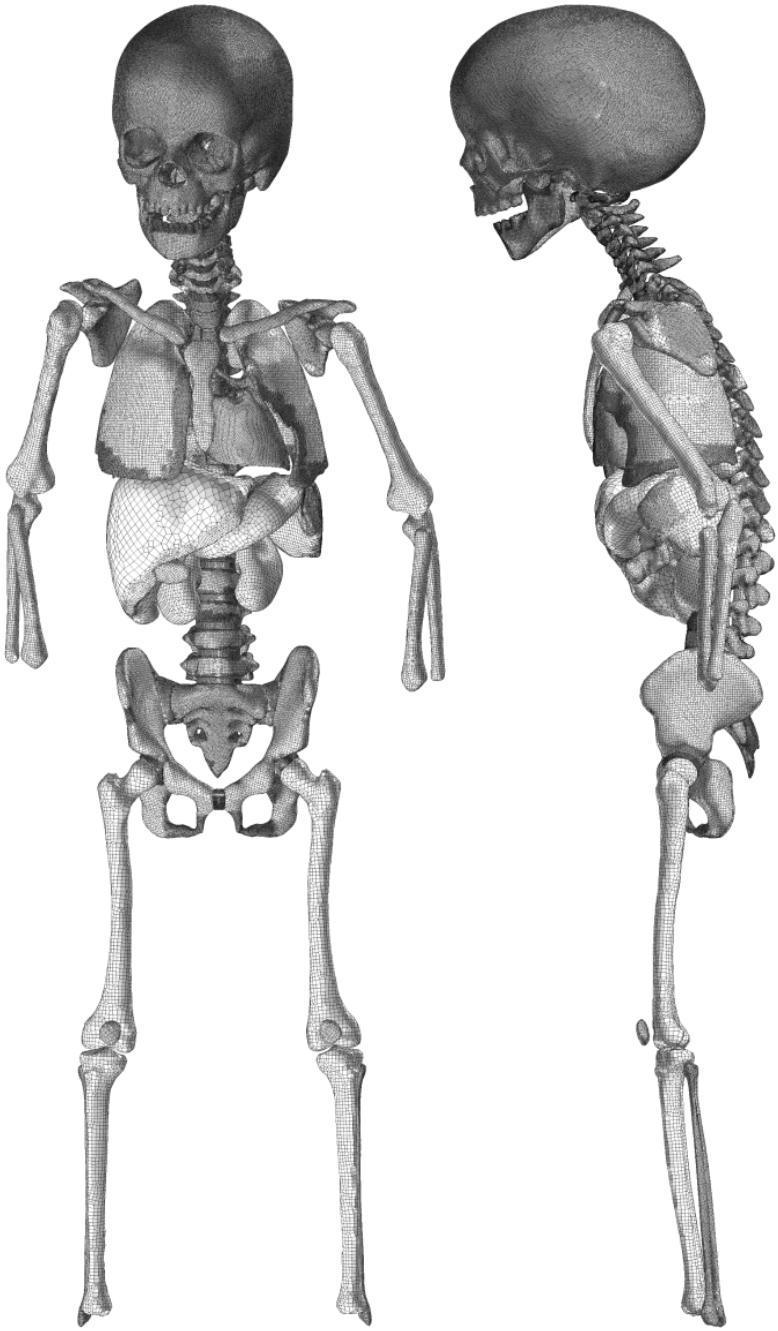


Fig. 8. Hexahedral mesh for a three-year-old child model

4.1 Ramses Model

Figure 5 shows hexahedral meshes for a Ramses model from the AIM@SHAPE Shape Repository. The initial octree was refined equally so that the size of octants becomes approximately 23 and then was locally refined three times based on α proposed in [8] ($\alpha = 50°$). Figures 5a and b show a mesh using the new template set, which has 2.14M elements. Figure 5c shows a mesh using the previous template set (without T_{4o}), which has 2.20M elements (2.8% more). Compared to the previous template set, the new set usually reduces the number of elements by a few percent while keeping almost the same or slightly better mesh quality. The actual reduction depends on each case, especially on the complexity of the refinement domain. The meshes shown in Figures 5b and c are similar except around the right eye.

Figure 6 shows the distribution of element quality of the two hexahedral meshes by the scaled Jacobian metric. The lowest minimum scaled Jacobian values are 0.015 for the mesh using the new set and 0.00079 for the mesh using the previous set. The two meshes have a small number of low quality hexahedra, but their overall quality is good.

4.2 Pediatric Brain Model

Figure 7 shows a hexahedral mesh for a pediatric brain model using the new set of the refinement templates. The initial octree was refined equally so that the size of octants becomes approximately 2.7, and then was locally refined twice based on α = 60°. The mesh has 340,917 hexahedra and is nicely adapted to the geometry. The lowest minimum scaled Jacobian of the hexahedra is 0.10. If the previous template set is used, a mesh with 355,904 hexahedra (4% more) is created.

4.3 Three-Year-Old Child Model

Figure 8 shows a hexahedral mesh for a three-year-old child model. Part of the components, left humerus, skull and brain, is already shown in Sections 3.1, 3.2 and 4.2, respectively. A hexahedral mesh is created for each of the body components separately using the new set of the refinement templates to apply the temporal rotation option to long bones and/or to apply the temporal local inflation option to components with thin regions. The final mesh has 2.79M hexahedra.

5 Conclusions

An octree-based mesh generation method has been developed to automatically create all-hexahedral meshes for complex geometries. An improved set of refinement templates allows the generation of geometry-adapted meshes with fewer elements while keeping almost the same or slightly better mesh quality. Two new options, temporal rotation and temporal local inflation, are useful to create meshes for long objects or thin regions. The capability of our mesh generation method is

demonstrated using a few triangulated surface models. The resulting meshes do not have any negative Jacobian elements even for complex geometries, and the quality of the meshes is good.

Acknowledgments

The initial triangulated surface model of the three-year-old child is provided courtesy of Dr. King Yang and Ms. Christina Huber of Wayne State University, Detroit, MI, USA. The Ramses model is provided courtesy of IMATI and INRIA by the AIM@SHAPE Shape Repository. This research is supported in part by the Southern Consortium for Injury Biomechanics (SCIB) No. DTNH22-01-H-07551.

References

1. Schneiders, R., Schindler, R., Weiler, F.: Octree-Based Generation of Hexahedral Element Meshes. In: Proceedings of the 5th International Meshing Roundtable, pp. 205–216. Sandia National Laboratories (1996)
2. Zhang, H., Zhao, G.: Adaptive Hexahedral Mesh Generation Based on Local Domain Curvature and Thickness Using a Modified Grid-Based Method. Finite Elements in Analysis and Design 43, 691–704 (2007)
3. Parrish, M., Borden, M., Staten, M., Benzley, S.: A Selective Approach to Conformal Refinement of Unstructured Hexahedral Finite Element Meshes. In: Proceedings of the 16th International Meshing Roundtable, Seattle, WA, pp. 251–268 (2007) doi: 10.1007/978-3-540-75103-8_15
4. Tchon, K., Dompierre, J., Camarero, R.: Automated Refinement of Conformal Quadrilateral and Hexahedral Meshes. International Journal for Numerical Methods in Engineering 59, 1539–1562 (2004)
5. Harris, N., Benzley, S., Owen, S.: Conformal Refinement of All-Hexahedral Element Meshes Based on Multiple Twist Plane Insertion. In: Proceedings of the 13th International Meshing Roundtable, Williamsburg, VA, Sandia National Laboratories, SAND #2004-3765C, pp. 157–168 (2004)
6. Shepherd, J.F., Zhang, Y., Tuttle, C.J., Silva, C.T.: Quality Improvement and Boolean-like Cutting Operations in Hexahedral Meshes. In: Proceedings of the 10th International Conference on Numerical Grid Generation in Computational Field Simulations, Crete, Greece (2007) (CD-ROM)
7. Merkley, K., Ernst, C., Shepherd, J., Borden, M.J.: Methods and Applications of Generalized Sheet Insertion for Hexahedral Meshing. In: Proceedings of the 16th International Meshing Roundtable, Seattle, WA, pp. 233–250 (2007), doi:10.1007/978-3-540-75103-8_14.
8. Ito, Y., Shih, A.M., Soni, B.K.: Octree-Based Reasonable-Quality Hexahedral Mesh Generation Using a New Set of Refinement Templates. International Journal for Numerical Methods in Engineering 77(13), 1809–1833 (2009)
9. Gottschalk, S., Lin, M., Manocha, D.: OBBTree: A Hierarchical Structure for Rapid Interference Detection. Computer Graphics (SIGGRAPH 1996) 30, 171–180 (1996)

10. Ito, Y., Nakahashi, K.: Improvements in the Reliability and Quality of Unstructured Hybrid Mesh Generation. International Journal for Numerical Methods in Fluids 45(1), 79–108 (2004)
11. Ito, Y., Shih, A.M., Soni, B.K., Nakahashi, K.: Multiple Marching Direction Approach to Generate High Quality Hybrid Meshes. AIAA Journal 45(1), 162–167 (2007)
12. Ito, Y., Shum, P.C., Shih, A.M., Soni, B.K., Nakahashi, K.: Robust Generation of High-Quality Unstructured Meshes on Realistic Biomedical Geometry. International Journal for Numerical Methods in Engineering 65(6), 943–973 (2006)