Subdivision Templates for Converting a Non-Conformal Hex-Dominant Mesh to a Conformal Hex-Dominant Mesh without Pyramid Elements

Soji Yamakawa[1], Iacopo Gentilini[2], and Kenji Shimada[3]

Department of Mechanical Engineering
Carnegie Mellon University
[1]soji@andrew.cmu.edu
[2]gentilini@cmu.edu
[3]shimada@cmu.edu

Abstract

This paper presents a computational method for converting a non-conformal hex-dominant mesh to a conformal hex-dominant mesh without help of pyramid elements. During the conversion, the proposed method subdivides a non-conformal element by applying a subdivision template and conformal elements by a conventional subdivision scheme. Although many finite element solvers accept mixed elements, some of them require a mesh to be conformal without a pyramid element. None of the published automated methods could create a conformal hex-dominant mesh without help of pyramid elements, and therefore the applicability of the hex-dominant mesh has been significantly limited. The proposed method takes a non-conformal hex-dominant mesh as an input and converts it to a conformal hex-dominant mesh that consists only of hex, tet, and prism elements. No pyramid element will be introduced. The conversion thus considerably increases the applicability of the hex-dominant mesh in many finite element solvers.

1. Introduction

This paper presents a computational method for converting a non-conformal hex-dominant mesh to a conformal hex-dominant mesh without introducing a pyramid element. The input hex-dominant mesh can include hex, tet, and prism elements (also known as wedge elements), and some quadrilateral faces

of the input mesh can be non-conformal; i.e., a triangular face can directly be connected to a quadrilateral face. The proposed method subdivides non-conformal hex-elements by conformal conversion templates and other elements by a conventional subdivision scheme. The conversion makes the hex-dominant mesh fully conformal without introducing a pyramid element, which is often rejected by a finite element solver. The output mesh thus can be used in a finite element solver even when it accepts neither a non-conformal face nor a pyramid element.

The accuracy of the finite element simulation depends on three factors: (1) mesh resolution, (2) mesh element quality, and (3) type of the mesh. A high-resolution mesh with good-quality elements yields an accurate solution. In addition, the mesh type also has a considerable impact on the accuracy of the finite element simulation. The type of a mesh depends on the type of elements included in the mesh. A volume mesh typically consists of four types of elements, tet, hex, prism, and/or pyramid elements as shown in Figure 1. Each type of element performs differently in the finite element simulation due to the difference of the shape functions, which interpolate physical quantity within the element.

A physical quantity within an element is interpolated as a linear combination of the shape functions, and higher-order shape functions can approximate a complex finite element solution better than lower-order shape functions. A shape function of an 8-node hexahedral element includes tri-linear, bi-linear, and linear terms, whereas 6-node prism and 5-node pyramid elements consist of bi-linear and linear terms, and a 4-node tetrahedral element consists only of linear terms. Therefore, a hexahedral element yields the most accurate solution, and an all-hexahedral mesh is preferred for a finite element analysis when available. Although a tetrahedral mesh can be created easily by an automatic mesh generation scheme [1-3], it requires more elements than an all-hexahedral mesh to obtain an equally accurate finite-element solution.

An all-hexahedral mesh, however, turned out to be difficult to create automatically for a complicated shape. Despite numerous attempts, none of the known methods can create an all-hexahedral mesh of adequate quality for an arbitrary geometric domain [4-18].

A hybrid mesh is an alternative to an all-hexahedral mesh and consists of hexahedral, tetrahedral, prism, and pyramid elements. The most common form of a hybrid mesh is a hex-dominant mesh [19-22], in which most of the volume is filled with hexahedral elements, and the rest with prism and tetrahedral elements, and pyramid elements are inserted between a quadrilateral face of a hexahedral or prism element and triangular faces of tetrahedral or prism elements as shown in Figure 2. However, some finite-element solvers do not accept a pyramid element. During the finite element calculation, an element is transformed into a master element, in which partial differential equations are numerically integrated [23]. Such a transformation utilizes Jacobian determinant, which is defined at every corner of an element where three edges con-

verge. However, four edges converge at one of the nodes of the pyramid, where Jacobian determinant becomes ambiguous. Due to this ambiguity, some finite element solvers do not accept a pyramid element, and it becomes necessary to connect two triangular faces to a quadrilateral face. Such a connection is called a non-conformal connection. Some solvers accept neither pyramid element nor non-conformal transition, and applicability of a hex-dominant mesh is significantly limited.

This paper proposes a set of mesh conversion templates that are applied to hex elements with non-conformal faces. The templates are based on HEXHOOP templates [13] and subdivide a non-conformal face into eight triangles (triangular pattern) and a conformal face into four quadrilaterals (rectangular pattern) as shown in Figure 3. Other elements are subdivided by a conventional subdivision scheme so that the elements become conformal after the subdivision. The conversion templates are constructed by a family of modular sub-templates that can be assembled to form conformal conversion templates for hex elements. The template design uses two types of modular sub-templates, called a *core* and a *cap*. A core module defines the subdivision patterns of two opposite faces of a hex element. Four caps define the subdivision patterns of the rest of the faces. The advantage of this method is that two subdivision patterns, rectangular and triangular patterns, can be mixed and matched freely on the exterior surfaces of a hex element.

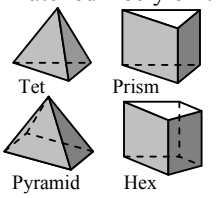


Figure 1 Four types of volume element

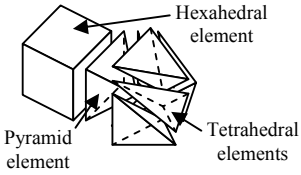


Figure 2 Inserting a Pyramid Element between Hex and Tet Elements

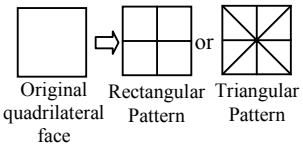


Figure 3 Two subdivision patterns of a quadrilateral face

The conversion makes a fully-conformal hex-dominant mesh that consists of tet, hex, and prism elements, and does not include pyramid elements. The mesh can be used by a finite element solver that rejects both non-conformal faces and pyramid elements. Hence, the conversion significantly increases the applicability of the hex-dominant mesh.

In the rest of the paper, Section 2 explains the construction of the subdivision templates, and Section 3 explains the non-template subdivision. Section 4 discusses the requirement of the input non-conformal hex-dominant mesh. Section 5 describes a post-process called cap-suppression. Section 6 shows some conversion examples, and Section 7 demonstrates the performance of the conformal hex-dominant mesh in finite element simulation. Section 8 discusses potential improvements, and Section 9 concludes the paper.

2. Construction of the conversion templates

The goal of the proposed method is to develop a system of conformal conversion templates for hex elements. The difficulty is that two face subdivision patterns, rectangular pattern and triangular pattern, need to be freely mixed on the exterior faces. Since a hex element consists of six faces, there are $2^6=64$ of the combinations of the subdivision patterns per element. Due to symmetry, the total number of combinations is reduced to ten as shown in Figure 4. Among the ten cases, there are three cases for which a known simple solution exists. H0 template can be achieved by a basic mid-point subdivision. H3 template can be constructed by sweeping a triangular pattern for two layers. H9 template can be constructed by adding a node in the middle of the template and making tet elements by connecting the mid node and exterior triangles. All other seven cases are non-trivial.

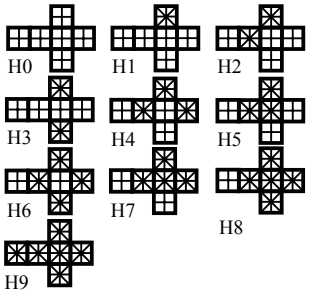


Figure 4 Ten required conversion templates

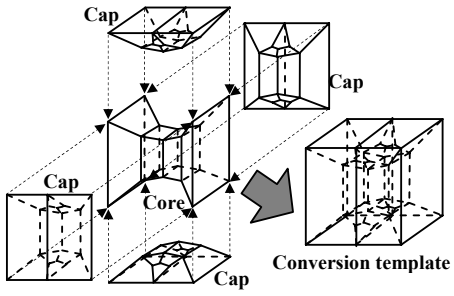


Figure 5 Modular approach to a all-hex template

To construct all non-trivial conversion templates, the proposed method constructs a conversion template by assembling sub-templates, each of which has either rectangular or triangular pattern on its external faces. The new template design uses two types of modular sub-templates, called *core*, and *cap*. A core module defines subdivision patterns of two opposite faces, and four caps define subdivision patterns of the other four faces as shown in Figure 5. Two subdivision patterns can be freely mixed and combined on the exterior faces of a hex element.

Construction of a Cap Module

Exterior faces of a cap module are grouped into four groups: T-face, which is exposed to the exterior after assembled together with other modules, B-face, which is connected to a slot of a core module, and F-face, which is connected to an adjacent cap module as shown in Figure 6.

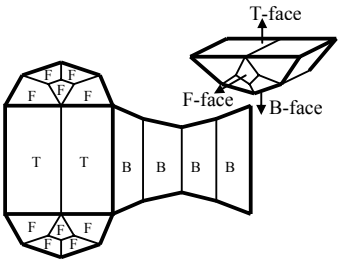


Figure 6 Faces of a Cap

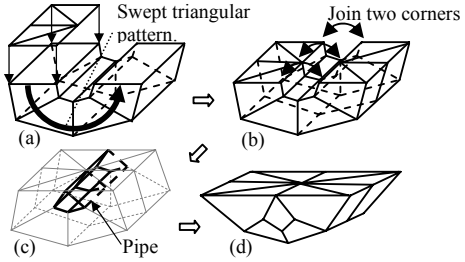


Figure 7 Construction of a triangular cap

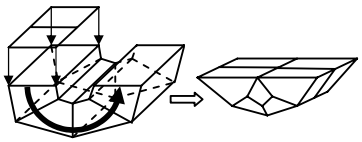


Figure 8 Construction of a rectangular cap

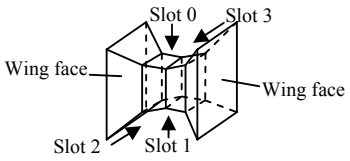


Figure 9 Two wing faces and four slots of a core module

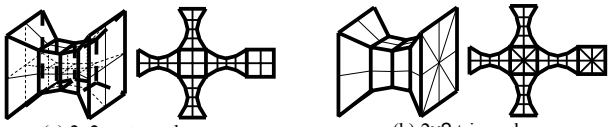
A cap that exposes a triangular pattern on the T-face can be constructed as follows. First, sixteen prism elements are created by sweeping four triangles as shown in Figure 7 (a). Next, the two corners of the two ends are joined together as shown in Figure 7 (b). Joining two corners yields a volumetric region surrounded by inside faces of the mesh as shown in Figure 7 (c). The region is called a *pipe* and is filled by two hex elements. The mesh is then deformed so that neither gaps nor overlaps remain when the cap is assembled with a core and other caps as shown in Figure 7 (d). This particular cap module consists of 16 prism elements and 2 hex elements. It is called a triangular cap and subdivides a corresponding quadrilateral face into eight triangles. Similarly, a rectangular cap, which exposes the rectangular pattern to the corresponding quadrilateral face, is constructed by applying a rectangular pattern to the T-face as shown in Figure 8.

Construction of a Standard Core

A core module has two *wing faces* and four *slots* as shown in Figure 9. Two wing faces are exposed to the exterior of the conversion template, and define subdivision patterns of the two opposite faces of the template. A slot is connected to a B-face of a cap module.

There are two types of standard cores, in which the subdivision patterns of the two wing faces are identical, called triangular core and rectangular core as shown in Figure 10. A standard core is created by sweeping a wing-face sub-

division pattern four times and then deforming the middle section so that a cap module fits in a slot. A rectangular core shown in Figure 10 (a) consists of 16 hex elements, and a triangular core shown in Figure 10 (b) consists of 32 prism elements.



(a) 2x2 rectangular core (b) 2x2 triangular core

Figure 10 Rectangular and Triangular cores

Assembling a Conversion Template

A core module and four cap modules can be assembled to form a conversion template for a hex element as shown in Figure 5. A B-face of a cap module is connected to a slot of a core module, and an F-face of a cap module is connected to an F-face of a neighboring cap module. After assembling, only T-faces of the four cap modules and wing faces of the core module are exposed to the exterior. This assembly is possible because the subdivision pattern of the B-face of a cap module matches the subdivision pattern of a slot of a core module, and the subdivision pattern of an F-face of a cap module always matches the F-face of the neighboring cap module regardless of the combination of four cap modules.

By combining appropriate core and caps, all templates except H5 shown in Figure 4 can be assembled. In H5 template, however, none of the opposite faces has an identical subdivision pattern, and it cannot be assembled with a standard core. H5 template requires a special type of core called a double-hoop core as explained in the next section.

Constructing a Double-Hoop Core

A standard core constrains at least one pair of opposite faces to have identical subdivision pattern. Since none of the opposite faces in H5 template shown in Figure 4 has identical subdivision pattern, standard core cannot be used for assembling H5 template. H5 template requires a core that has a triangular pattern on one of the wing faces and a rectangular pattern on the other.

A double-hoop core is created as follows. First, a H1 template is assembled from a rectangular core, a triangular cap, and three rectangular caps. The H1 template is oriented so that a triangular pattern is on the top face. The H1 template can be split into top and bottom halves as shown in Figure 11 (a). The cross-section between the top and bottom halves lies on a plane and ex-

poses interior faces of cap and core modules as shown in Figure 11 (b). Two layers of hex elements are then created by sweeping the cross-sectional pattern and inserted between top and bottom halves as shown in Figure 11 (c). The result is a conformal mesh consisting of hex and prism elements. The top face of the mesh has a triangular pattern, and the bottom face a rectangular pattern. Four side faces are also rectangular pattern, however, each face has four rectangles vertically and two horizontally as shown in Figure 11 (d). The mesh is then re-oriented so that the triangular pattern face to the left, and then the middle section of the mesh is shrunk so that four caps fit to the mesh as shown in Figure 11 (e). Interior nodes should be re-located by an appropriate mesh-smoothing scheme.

Since this double-hoop core exposes a triangular pattern on one of the two wing faces, and a rectangular pattern on the other, the H5 template can be assembled by a double-hoop core, two rectangular caps, and two triangular caps. This core is called a double-hoop core because two 'hoops' of pipe volumes exist in the H5 template as shown in Figure 12.

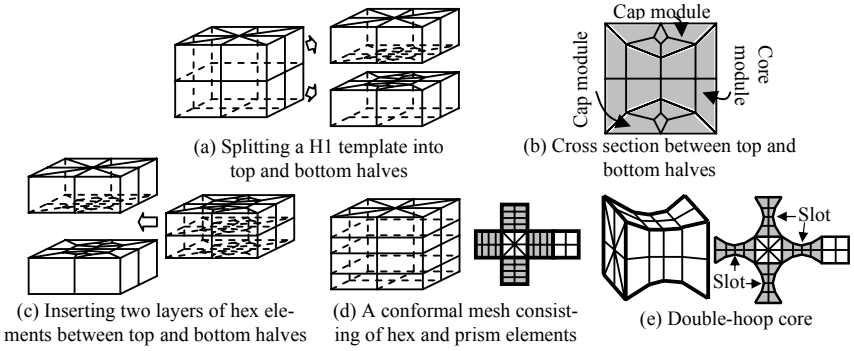


Figure 11 Inserting two layers of hex elements in the middle of the H1 template

3. Subdividing Other Elements

Subdividing Prism Elements

A prism element is subdivided into six hex elements as shown in Figure 13 (c), or eight prism elements as shown in Figure 13 (b) or Figure 13 (c). The subdivision pattern is chosen based on the constraint on the triangular faces. A triangular face of a prism element is constrained if it is connected to a non-

conformal face of the neighboring hex element or a triangular face of the neighboring tet element. Since the subdivision patterns on both triangular faces of a prism element needs to be identical, when one triangular face of a prism element is constrained, the opposite triangular face is also constrained. If multiple prism elements are chained by triangular faces, the constraint on one end of the chain propagate through the chain of the prism element to the other end, and all triangular faces of the prism chain are constrained.

If the two ends of a prism chain are exposed to the exterior of the mesh, each of the prism elements in the chain is subdivided into six hex elements as shown in Figure 13 (a). If one of the ends of a prism chain is connected to a tet element, and the other end is also connected to a tet element or exposed to exterior of the mesh, each of the prism elements in the chain is subdivided into eight prism elements as shown in Figure 13 (b).

If one of the ends of a prism chain is connected to a non-conformal face of a hex element, each of the prism elements in the chain is subdivided into eight prism elements as shown in Figure 13 (c). In this case, the subdivision pattern of the triangle connected to the non-conformal face must match half of the subdivision pattern of the non-conformal face as shown in Figure 14. However, due to the asymmetry of the triangle subdivision pattern, if both ends of a prism chain are connected to non-conformal faces, constraints from the two ends may conflict to each other as shown in Figure 15. Such a conflict can be resolved by subdividing one of the prism elements in the chain into two tet elements and three prism elements by adding two nodes inside the original prism element. However, this resolution often yields a low-quality element. The hex-dominant mesh generator thus needs to avoid such conflict when it creates an input hex-dominant mesh. Discussion on the requirement of the input hex-dominant mesh is also found in Section 4.

Subdividing Tet Elements

A tet element is subdivided by (1) adding a node at the center of the element, (2) subdividing triangular faces so that it satisfies the constraints imposed by the neighboring prism element or a non-conformal face, and then (3) connecting each triangle to the node created at the center of the original tet element.

A triangular face of a tet element can be constrained by the neighboring prism element, or a non-conformal face connected to the triangular face. All non-constrained faces of tet elements are subdivided into the pattern shown in Figure 16 (a), and constrained faces into the pattern shown in Figure 16 (b) so that the faces conform to the neighboring elements after the subdivision.

The nodes added at the center of the original tet element will be deleted by mesh smoothing [24-27], local transformation [28, 29], and edge collapse [30] as many as possible after the conversion.

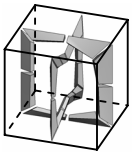


Figure 12 Double hoop in the H5 template

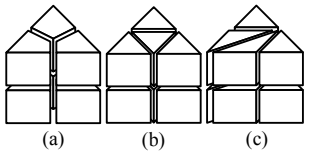


Figure 13 Three possible subdivision patterns of a prism element

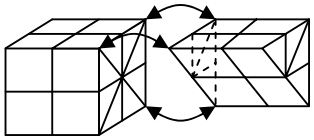


Figure 14 Subdivision pattern of a triangular face connected to a non-conformal face

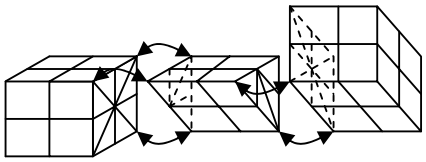


Figure 15 Conflict of triangle subdivision patterns from the two ends of a prism chain

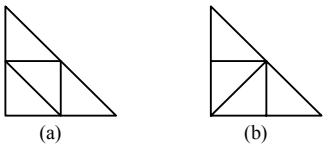


Figure 16 Two subdivision patterns of a face of a tetrahedral element

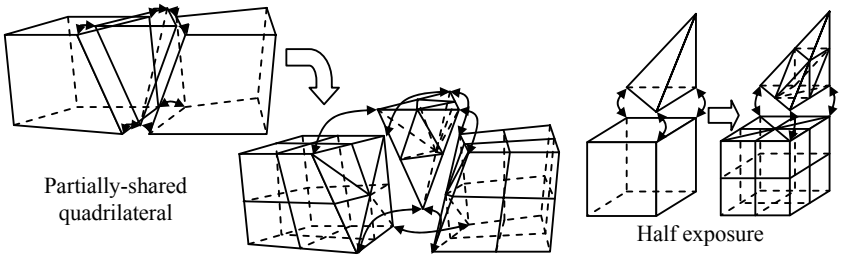


Figure 17 Permissible non-conformal connections

4. Requirement of the Input Hex-Dominant Mesh

The input to the proposed method is a non-conformal hex-dominant mesh, which can include a triangular face directly connected to a quadrilateral face of a hex element. The conversion templates are flexible and can convert even more complex connection such as a partially-shared quadrilateral and half-exposed quadrilateral to a conformal connection as shown in Figure 17. Such flexibility helps a hex-dominant mesh generator to create more hex elements in the input mesh.

Nonetheless, the input hex-dominant mesh needs to satisfy certain conditions discussed in this section. A hex-dominant mesh that satisfies those con-

ditions can easily be created by a conventional algorithm [19, 21, 31] with a slight modification.

Hex Element

A quadrilateral face of a hex element can be connected to a quadrilateral face of a neighboring hex or prism element, or one or two triangular faces of neighboring tet and/or prism elements.

However, when a quadrilateral face is sharing three nodes with a quadrilateral face of a neighboring hex element, two quadrilateral faces must share the diagonal. The connection shown in Figure 18 is thus impermissible.

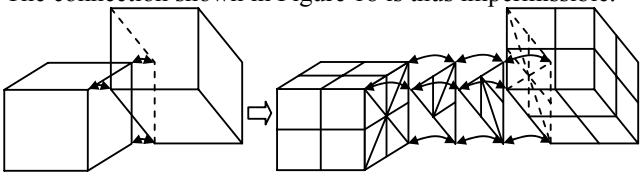


Figure 18 Impermissible connection: Two quadrilateral faces sharing three nodes are not sharing a diagonal.

Prism Element

All quadrilateral faces of prism elements need to be conformal. I.e., a quadrilateral face of a prism element needs to be completely shared by the neighboring prism or hex element, or completely exposed to the exterior. No triangular face can be connected to a quadrilateral face of a prism element.

A HEXHOOP template for a prism element can also be assembled as described in the original HEXHOOP paper [32], and such a template can subdivide a non-conformal prism element into conformal elements. However, the experiments performed in this research showed that applying a prism template often yields very low-quality elements. The possible reason why a prism template yields a low-quality element is because prism element are created in the difficult region where hex elements cannot be placed and the quality of the original prism element is often not high enough. The original element needs to be of reasonably high quality to have reasonably good elements after the conversion. Therefore, the current implementation does not utilize prism templates.

As already discussed in Section 1, the subdivision pattern of one of the two triangular faces of a prism element must match the pattern of the other triangular face. The hex-dominant mesh generator must avoid a prism element with two conflicting subdivision patterns on the two triangular faces.

If the input non-conformal hex-dominant mesh satisfies the above-mentioned conditions, the proposed conversion templates create a topologically-valid conformal hex-dominant mesh.

5. Cap Suppression

Some elements from cap modules may be deleted after the conversion. If two neighboring cap modules are sharing T-face as shown in Figure 19 (a), elements included in both cap modules can be eliminated by joining corresponding nodes as shown in Figure 19 (b). The subdivision pattern between the two conversion templates is shown in Figure 19 (c). Since the quality of the elements in cap modules tends to be relatively lower than for other elements, the overall mesh quality can be improved by eliminating cap modules.

(a) Two cap modules facing each other (b) Joining corresponding nodes (c) After suppressing caps

Figure 19 Suppressing two cap modules facing each other

6. Conversion Examples

Figure 20 and Figure 21 show conformal hex-dominant meshes created by the proposed method. The input non-conformal hex-dominant meshes are created by first converting a tet mesh into a prism-tet hybrid mesh by the method described in [33], and then merging pairs of neighboring prism elements into hex elements. Table 1 shows the mesh statistics, including Scaled Jacobian, which is defined at each node as a triple scalar product of the edge vectors of three converging edges of an element divided by the product of the lengths of the three vectors.

Table 1 Statistics of the sample meshes

	# node	# element			Volume Ratio (%)			Minimum Scaled Jacobian		Max Tet Radius Ratio	Max Quad Warpage
		Hex	Prism	Tet	Hex	Prism	Tet	Hex	Prism		
Leg-bone	10,001	3,288	3,872	16,352	36	23	41	0.20	0.17	59.32	46 deg
AAA	13,462	7,468	6,728	2,772	68	30	2	0.20	0.20	59.86	54 deg

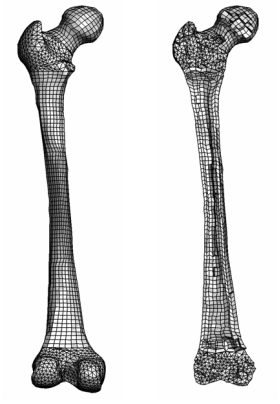


Figure 20 Leg-bone model

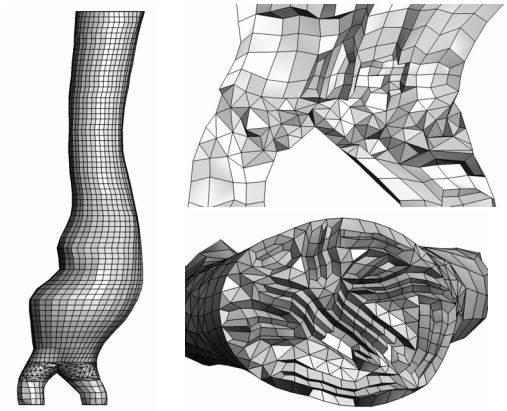


Figure 21 Abdominal aortic aneurysm model

7. Experimental Finite Element Analysis

This section presents results from a series of structural analyses of a connecting rod of an automotive engine. Four sets of analyses, each with (1) linear hex-dominant, (2) linear tet, (3) quadratic hex-dominant, and (4) quadratic tet meshes have been performed with different node density to compare the characteristics of different types of meshes. Statistics of the meshes used in the analyses are shown in Table 2.

The analyses are calculated by Abaqus/CAE Version 6.7 [34] on a PC with a Dual Core AMD Opteron Processor at 2.0GHz and with 3GB RAM. A load of 18,000N is applied to the internal surface of the crankshaft bearing housing in form of uniform pressure, and one half of the surface of the wrist pin seat is constrained against displacements in the XY-plane, and symmetry boundary conditions are also applied. Figure 22 shows a von Mises stress plot calculated with one of the hex-dominant meshes.

Figure 23 and Figure 24 plot in-body maximum displacement and von Mises stress vs. the number of nodes calculated by using linear and quadratic elements, respectively. In both plots, the solutions from tet meshes show a smoother convergence, and the solutions from hex-dominant mesh show an oscillatory behavior. Nonetheless, the solutions from tet meshes and hex-dominant meshes both converge to a similar values as the number of nodes increases, and virtually no difference was observed between tet mesh and hex-dominant meshes if the number of node reaches 25,000 (linear) and 110,000 (quadratic).

Figure 25 depicts von Mises stresses along a path starting in the middle of the rod and moving toward the surface at the fillet location, calculated from quadratic tet and hybrid meshes. The path plot also indicates that the results from a tet mesh and a hybrid meshes converge to the same solution as the mesh resolution increases.

The results imply that hybrid meshes exhibit unsmooth convergence pattern when the mesh resolution is relatively low. However, the solution becomes as accurate as tet meshes when node density becomes high.

When the number of nodes is equal, a hex-dominant mesh is expected to take shorter computational time to obtain a solution than a tet mesh. The dominant component of the finite element computation comes from the matrix solver. If the number of nodes is equal, the size of the stiffness matrix is equal. However, as can be seen from Table 2, average number of elements per node is much smaller in a hybrid mesh than a tet mesh, and the bandwidth of the stiffness matrix from a hex-dominant mesh is thus smaller. Hence, the matrix solver should take shorter time to solve a stiffness matrix from a hex-dominant mesh.

The expectation was confirmed for quadratic meshes. As can be seen from Figure 26, a quadratic hex-dominant mesh required shorter CPU time than a quadratic tet mesh. The difference becomes significant as the number of nodes increases. Results from the linear meshes, however, contradicted the expectation; a linear hex-dominant mesh took more CPU time than a linear tet mesh. It could be due to the characteristic of the solver used in the experiment. Further study is required to find the cause of the contradiction.

Table 2 Statistics of the meshes used in the analyses

Hex dominant mesh						Tet mesh				
# node		# elem	# elem. / # node		Hex vol. ratio	# node quad.		# elem	# elem. / # node	
lin.	quad.		lin.	quad.	(%)	lin.	quad.		lin.	quad.
4,196	17,850	5,571	1.33	0.31	57	4,182	28,260	17,588	4.21	0.62
5,208	22,151	6,894	1.32	0.31	57	5,522	37,785	23,844	4.32	0.63
6,075	25,599	7,778	1.28	0.30	59	7,355	50,998	32,689	4.44	0.64
8,692	37,909	12,517	1.44	0.33	51	10,156	71,456	46,556	4.58	0.65
9,977	42,406	13,166	1.32	0.31	64	12,927	91,542	60,106	4.65	0.66
12,849	54,304	16,591	1.29	0.31	67	14,616	104,280	69,014	4.72	0.66
15,835	66,332	19,804	1.25	0.30	75	16,783	120,133	79,835	4.76	0.66
16,173	69,292	21,900	1.35	0.32	62	19,371	139,655	93,528	4.83	0.67
16,456	70,053	21,824	1.33	0.31	66	22,504	162,980	109,694	4.87	0.67
23,751	101,915	32,367	1.36	0.32	68	24,075	174,708	117,851	4.90	0.67
24,456	104,610	32,930	1.35	0.31	70	26,249	190,928	129,169	4.92	0.68
29,991	125,860	37,610	1.25	0.30	75	28,979	211,373	143,414	4.95	0.68
30,914	130,977	40,222	1.30	0.31	73	30,126	220,152	149,666	4.97	0.68
31,952	135,154	41,306	1.29	0.31	75	31,171	227,889	155,031	4.97	0.68

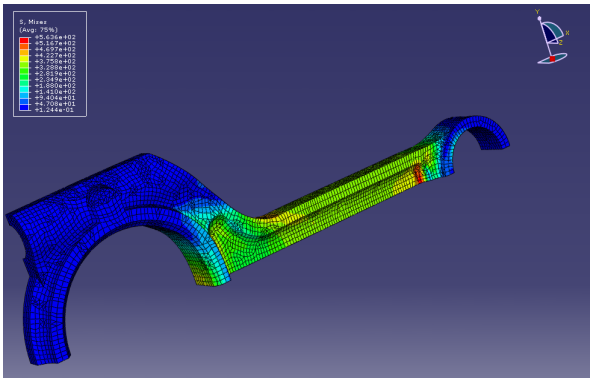


Figure 22 Connecting rod

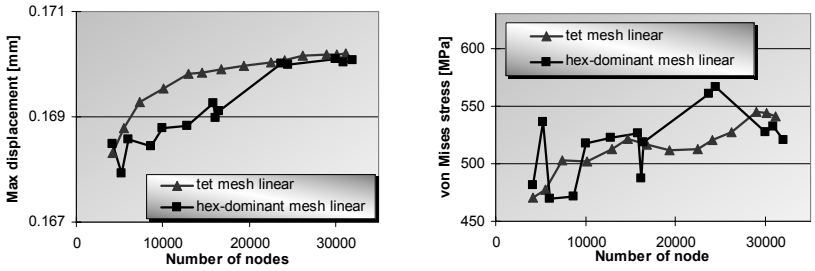


Figure 23 Results with linear elements

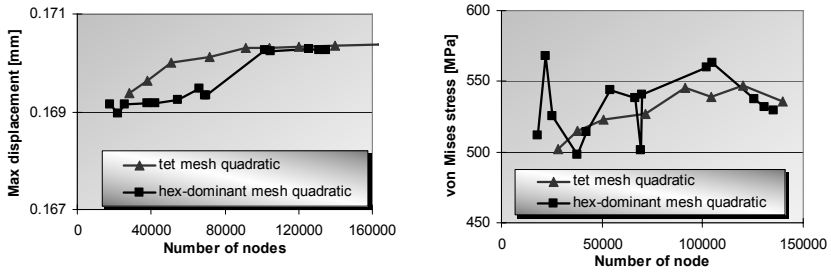


Figure 24 Results with quadratic elements

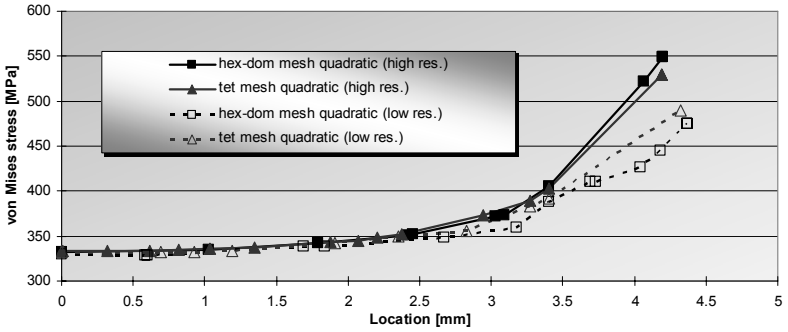


Figure 25 Path plot of the von Mises stress through the rod

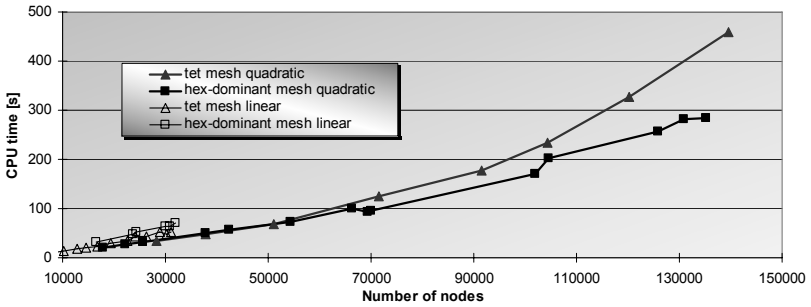


Figure 26 CPU time plot

The experimental results imply that a quadratic hex-dominant mesh is advantageous especially when the mesh resolution is high compared to a tet mesh. Since the difference between a linear element and a quadratic element becomes significant in a non-linear analysis, a quadratic hex-dominant mesh should be particularly effective for a large-scale non-linear analysis. Further research is needed to test this speculation.

8. Potential Improvements

The quality of the output conformal hex-dominant mesh can be further improved by taking as input a higher quality non-conformal hex-dominant mesh. If a subdivision template is applied to a low-quality element, the output mesh will include an even lower-quality element. Such a low-quality element should be avoided by eliminating low-quality elements included in the input non-conformal mesh.

The quality of the output mesh can also be improved by avoiding certain configuration of the input mesh can be avoided. The quality of the elements included in non-trivial templates (H1, H2, H5, H6, H7, and H8 templates) is relatively low due to the complex structure of the templates - the element quality of the non-trivial templates is shown in Table 3. Values in parentheses indicate the mesh quality after applying conventional smoothing schemes. In particular, H5 template has the lowest quality element among the non-trivial templates. The quality of the output mesh therefore can be improved by developing and applying a new non-conformal hex-dominant meshing scheme that reduces the usage of the H5 template.

The quality of the output mesh can also be improved by applying mesh-smoothing schemes. Note, however, that not all conventional mesh-smoothing schemes perform well on a hybrid mesh.

Table 3 The quality of the non-trivial templates

	H1	H2	H5	H6	H7	H8
Minimum Scaled Jacobian	0.34 (0.28)	0.34 (0.30)	0.20 (0.20)	0.34 (0.21)	0.34 (0.20)	0.34 (0.20)
Worst Aspect Ratio	2.9 (3.0)	2.9 (3.1)	5.3 (4.6)	2.9 (3.0)	2.9 (3.1)	2.9 (3.2)
Largest Quadrilateral Warpage	28.2deg (1.1deg)	28.2deg (1.9deg)	62.3deg (18.1deg)	28.2deg (1.4deg)	28.2deg (1.3deg)	28.2deg (0.8deg)

9. Conclusions

This paper has presented a method for converting a non-conformal hex-dominant mesh into a conformal hex-dominant mesh by applying a set of subdivision templates. The method always creates a conformal hex-dominant mesh if the input non-conformal hex-dominant mesh satisfies a certain condition, where a non-conformal hex-dominant mesh can be easily created by an existing non-conformal hex-dominant meshing scheme. Experimental results indicate the method creates a good-quality conformal hex-dominant mesh. A series of finite element simulations have been performed and the results imply that a conformal hex-dominant mesh can be effective for a large-scale non-linear analysis.

References

[1] P. L. George, "Tet Meshing: Construction, Optimization and Adaptation," Proceedings of 8th International Meshing Roundtable, pp. 133-141, 1999.

[2] S. Yamakawa and K. Shimada, "Anisotropic Tetrahedral Meshing via Bubble Packing and Advancing Front", *International Journal for Numerical Methods in Engineering*, vol. 57, pp. 1923-1942, 2003.

[3] J. Shewchuk, "Constrained Delaunay Tetrahedralizations and Provably Good Boundary Recovery," Proceedings of 11th International Meshing Roundtable, pp. 193-204, 2002.

[4] T. D. Blacker and R. J. Meyers, "Seams and Wedges in Plastering: A 3-D Hexahedral Mesh Generation Algorithm", *Engineering with Computers*, vol. 2, pp. 83-93, 1993.

[5] R. Schneiders, "A Grid-based Algorithm for the Generation of Hexahedral Element Meshes", *Engineering with Computers*, vol. 12, pp. 168-177, 1996.

[6] T. J. Tautges, T. Blacker, and S. A. Mitchell, "The Whisker Weaving Algorithm: A Connectivity-Based Method for Constructing All-Hexahedral Finite Element Meshes", *International Journal for Numerical Methods in Engineering*, vol. 39, pp. 3327-3349, 1996.

[7] F. Ledoux and J.-C. Weill, "An Extension of the Reliable Whisker Weaving Algorithm," Proceedings of 16th International Meshing Roundtable, pp. 215-232, 2007.

[8] M. L. Staten, S. J. Owen, and T. D. Blacker, "Unconstrained Paving & Plastering: A New Idea for All Hexahedral Mesh Generation," Proceedings of 14th International Meshing Roundtable, pp. 399-416, 2005.

[9] R. Taghavi, "Automatic Block Decomposition Using Fuzzy Logic Analysis," Proceedings of 9th International Mething Roundtable, pp. 187-192, 2000.

[10] D.-Y. Kwak and Y.-T. Im, "Remeshing for Metal Forming Simulations-Part II: Three-Dimensional Hexahedral Mesh Generation", *International Journal for Numerical Methods in Engineering*, vol. 53, pp. 2501-2528, 2002.

[11] M. Hariya, I. Nishigaki, I. Kataoka, and Y. Hiro, "Automatic Hexahedral Mesh Generation with Feature Line Extraction," Proceedings of 15th International Meshing Roundtable, pp. 453-467, 2006.

[12] L. Maréchal, "A New Approach to Octree-Based Hexahedral Meshing," Proceedings of 10th International Meshing Roundtable, pp. 209-221, 2001.

[13] S. Yamakawa and K. Shimada, "HEXHOOP: Modular Templates for Converting a Hex-Dominant Mesh to an ALL-Hex Mesh", *Engineering with Computers,* vol. 18, pp. 211-228, 2002.

[14] S. A. Mitchell, "The All-Hex Geode-Template for Conforming a Diced Tetrahedral Mesh to any Diced Hexahedral Mesh," Proceedings of 7th International Meshing Roundtable, pp. 295-305, 1998.

[15] D. R. White and T. J. Tautges, "Automatic Scheme Selection for Toolkit Hex Meshing", *International Journal for Numerical Methods in Engineering,* vol. 49, pp. 127-144, 2000.

[16] M. Lai, S. Benzley, and D. White, "Automated Hexahedral Mesh Generation by Generalized Multiple Source to Multiple Target Sweeping", *International Journal for Numerical Methods in Engineering,* vol. 49, pp. 261-275, 2000.

[17] J. Shepherd, S. A. Mitchell, P. Knupp, and D. White, "Methods for Multisweep Automation," Proceedings of 9th International Meshing Roundtable, pp. 77-87, 2000.

[18] W. R. Quadros and K. Shimada, "Hex-Layer: Layered All-Hex Mesh Generation on Thin Section Solids via Chordal Surface Transformation," Proceedings of 11th International Meshing Roundtable, pp. 169-180, 2002.

[19] S. Meshkat and D. Talmor, "Generating a Mixed Mesh of Hexahedra, Pentahedra and Tetrahedra from an Underlying Tetrahedral Mesh", *International Journal for Numerical Methods in Engineering,* vol. 49, pp. 17-30, 2000.

[20] R. J. Meyers, T. J. Tautges, and P. M. Tuchinsky, "The "Hex-Tet" Hex-Dominant Meshing Algorithm as Implemented in CUBIT," Proceedings of 7th International Meshing Roundtable, pp. 151-158, 1998.
[21] S. J. Owen and S. Saigal, "H-Morph: an Indirect Approach to Advancing Front Hex Meshing", *International Journal for Numerical Methods in Engineering,* vol. 49, pp. 289-312, 2000.
[22] S. Yamakawa and K. Shimada, "Hex-Dominant Mesh Generation with Directionality Control via Packing Rectangular Solid Cells," Proceedings of Geometric Modeling and Processing 2002, pp., 2002.
[23] E. B. Becker, G. F. Carey, and J. T. Oden, *Finite Elements: an Introduction,* vol. 1: Prentice Hall, 1981.
[24] P. M. Knupp, "Achieving Finite Element Mesh Quality via Optimization of the Jacobian Matrix Norm and Associated Quantities. Part II - A Framework for Volume Mesh Optimization and the Condition Number of the Jacobian Matrix", *International Journal for Numerical Methods in Engineering,* vol. 48, pp. 1165-1185, 2000.
[25] L. A. Freitag and P. E. Plassmann, "Local Optimization-based Untangling Algorithms for Quadrilateral Meshes," Proceedings of 10th International Meshing Roundtable, pp. 397-406, 2001.
[26] T. Zhou and K. Shimada, "An Angle-Based Approach to Two-dimensional Mesh Smoothing," Proceedings of 9th International Meshing Roundtable, pp. 373-384, 2000.
[27] N. A. Calvo and S. R. Idelsohn, "All-Hexahedral Mesh Smoothing with a Node-Based Measure of Quality", *International Journal for Numerical Methods in Engineering,* vol. 50, pp. 1957-1967, 2001.
[28] B. M. Klingner and J. R. Shewchuck, "Aggressive Tetrahedral Mesh Improvement," Proceedings of 16th International Meshing Roundtable, pp. 3-23, 2007.
[29] B. Joe, "Construction of Three-Dimensional Improved-Quality Triangulations Using Local Transformations", *SIAM Journal on Scientific Computing,* vol. 16, pp. 1292-1307, 1995.
[30] J. F. Molinari and M. Ortiz, "Three-Dimensional Adaptive Meshing by Subdivision and Edge-Collapse in Finite-Deformation Dynamic-Plasticity Problems with Application to Adiabatic Shear Banding", *International Journal for Numerical Methods in Engineering,* vol. 53, pp. 1101-1126, 2002.
[31] S. Yamakawa and K. Shimada, "Fully-Automated Hex-Dominant Mesh Generation with Directionality Control via Packing Rectangular Solid Cells", *International Journal for Numerical Methods in Engineering,* vol. 57, pp. 2099-2129, 2003.
[32] S. Yamakawa and K. Shimada, "HEXHOOP: Modular Templates for Converting a Hex-dominant Mesh to an All-hex Mesh," Proceedings of 10th International Meshing Roundtable, pp. 235-246, 2001.
[33] S. Yamakawa and K. Shimada, "Converting a Tetrahedral Mesh to a Prism-Tetrahedral Hybrid Mesh for FEM Accuracy and Efficiency," Proceedings of ACM Symposium on Solid and Physical Modeling 2008, pp. 287-294, 2008.
[34] "Abaqus," Dassault Systemes (http://www.simulia.com).