

Efficient Cutaway Model Generation for Interactive Cutaway Viewing System

Hiroaki Date ¹, Hiromu Furukawa ¹, Masahiko Onosato ¹

(1) : Kita14, Nishi 9, Sapporo 060-0814
 Phone +81-11-706-6447 Fax +81-11-706-7120
 E-mail : {hdate, onosato}@ssi.ist.hokudai.ac.jp,
 furukawa@dse.ssi.ist.hokudai.ac.jp

Abstract: Developing the techniques for an easy and intuitive understanding of complex 3D geometric models is one of the important issues for the 3D applications in several areas. In this paper, first, our interactive cutaway viewing system is introduced. In the system, an efficient understanding of complex 3D geometric models is realized by showing the cutaway model, which is the result of the Boolean difference operation on the given model and arbitrary cutting volume, using a 3D user interface which consists of 6DOF input devices and a glassless stereoscopic display. The efficient cutaway model generation method for the interactive cutaway viewing system is proposed in the following section. The method consists of the distance thresholding between the cutaway volume and the objects for reducing the collision and inclusion test, the use of time coherence for updating the object's information efficiently, and the model simplification and adaptive cutaway model generation for reducing several geometric calculations. The effectiveness of the method is shown thorough the experimental results using various types of complex geometric models.

Key words: Cutaway viewing, geometric models, Boolean operation, model simplification.

1- Introduction

An efficient understanding of the 3D geometric models is indispensable for several 3D applications in the field of CG, CAD/CAM/CAE, and GIS. Since the complexity of the geometric models used in the 3D applications becomes drastically high through the recent progress of 3D scanning and geometric modeling technology, the requirements for an easy and intuitive understanding are progressively increasing.

For geometric models with complex inner structures or a lot of parts, it is difficult to understand them by viewing exterior. Therefore, additional techniques for viewing the models are required. Currently, three kinds of techniques are used for understanding complex geometric models. 1) The transparency view shows the model by making the extraneous parts transparent in order to see the inner parts of interest [DW1]. 2)

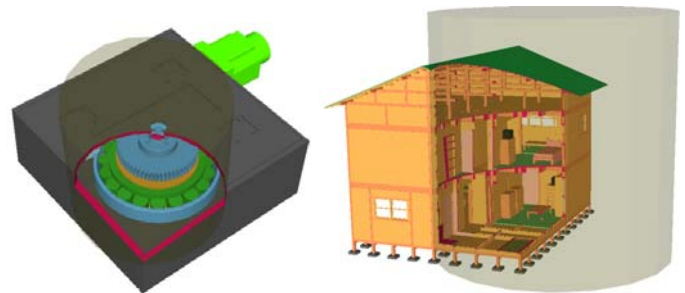


Figure 1: Cutaway views of 3D models with inner structures.

The exploded view shows the model by separating exterior parts away from inner parts of interest [LA1]. 3) Cutaway view shows the partial model cut with plane or arbitrary volume (cutting volume) [LR1, BF1, FD1], as shown in Figure 1. These viewing methods share a common issue: to determine which parts should be processed for seeing the inner part of interest, and what kind of process should be applied to the parts.

In our study, we focused on cutaway viewing. In cutaway viewing, it is important that the user can easily locate the cut which makes the region of interest visible. In addition, cutting the model with arbitrary shapes is effective in the cutaway view. For instance, cutting with a cube enables multiple cross-sections of the model to be seen at once.

From these backgrounds, we developed an interactive cutaway viewing system [FD1]. In order to realize an easy and intuitive location of the cut, three dimensional user interfaces are used. The user can determine the location of the cut through a simultaneous manipulation of the model and cutting volume using two 6DOF input devices and stereoscopic views. The system generates a cutaway model by Boolean difference operation on the two triangular meshes of a given model and cutting volume. Therefore, the cut is made with an arbitrary shape. Moreover, the system enables users to understand the model by touching the surface of the cutaway model using haptic force feedback.

In the interactive cutaway viewing system, real-time generation of the cutaway model is required. The cutaway model is the result of the Boolean difference operation on the input model and the cutting volume. Several Boolean operation methods are proposed for surfel-bounded solids [AD1], nef-polyedra [HK1], subdivision surfaces [BK1], and polygonal meshes [PC1]. However, these methods are not suitable for interactive viewing because of their high computational cost. On the other hand, there are some fast rendering techniques for the results of Boolean operations. Blister [HR1] renders the result of a Boolean using depth peeling on graphics hardware in real-time. Burns et al. [BF1] introduced a method for generating a view-dependent cutaway view with modifications to the polygonal rendering pipeline. However, in our interactive cutaway viewing system, the cutaway model consisting of the triangular mesh is required because of haptic force feedback. A straightforward method of the Boolean difference operation for triangular meshes is provided in our previous work [FD1]. Based on the method, in this paper, we propose some techniques of efficient cutaway model generation for interactive cutaway viewing.

This paper is organized as follows: In chapter 2, our interactive cutaway viewing system is introduced, and the basic method of the cutaway model generation is described in chapter 3. In chapter 4, a method for efficient cutaway model generation is proposed. The effect of our method is evaluated in chapter 5. Conclusions are described in chapter 6.

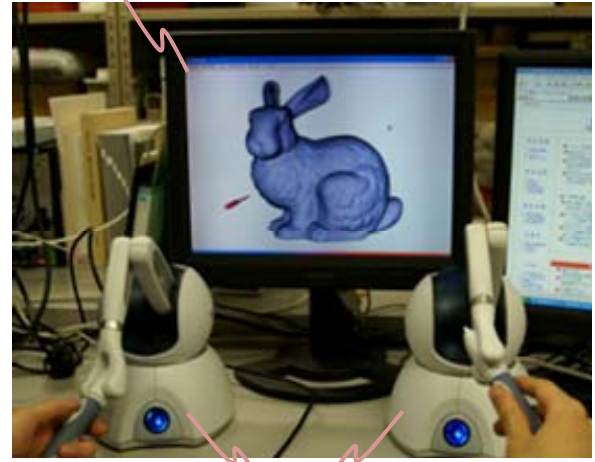
2- Interactive cutaway viewing system

Figure 2 shows the hardware configuration of the interface of our interactive cutaway viewing system [FD1]. Our interface consists of two haptic force feedback devices (PHANToM Omnis) and a glassless stereoscopic display (LL-151D). A PHANToM Omni has 6DOF for input (the position and orientation), and 3DOF for output (haptic force feedback).

Our system has two modes, location mode and focusing mode. In location mode, users locate the input model and the cutting volume to see the region of interest in the model. Locations of them can be done simultaneously using two 6DOF input devices, that is, a device is used for locating the input model, and another one is for the cutting volume. In focusing mode, users focus on the region of interest to better understand the model. Using a 6DOF input device, users can change the position and orientation of the cutaway model. In this mode, the relative location between the input model and the cutting volume is fixed. Using another input device with haptic force feedback, the users can touch the surface of the cutaway model. In the both modes, the model and cutting volume or the resulting cutaway model is always shown using the stereoscopic view. The stereoscopic view is effective for intuitive locations and easy understanding of the 3D models.

To evaluate the effectiveness of the interface system, an evaluation experiment was performed. The location task was configured taking the actual operations of the cutaway viewing system in account, which was to fit a cube to the given target location in 3D space. In our experiments, by using 3D interfaces (our configuration), the time of locating the cube

Glassless stereoscopic display [LL-151D (Sharp)]



Two input/force feedback devices (6DOF input and 3DOF force feedback) [PHANToM Omni (SensAble Technologies)]

Figure 2: Device configuration of our cutaway viewing system.

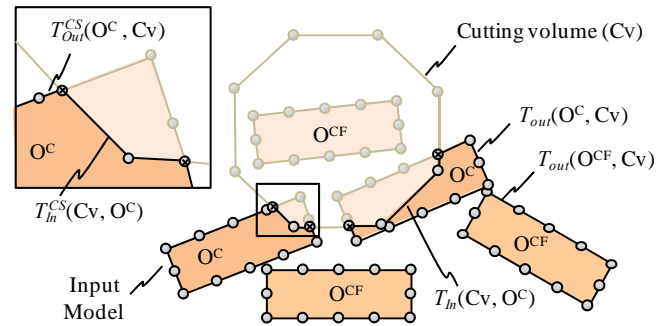


Figure 3: Cutaway model.

was shortened to one seventh on average (twelve subjects) in comparison with the case of using traditional interface configurations (2D mouse and display) [FD1]. The results reveal that our interface is suitable for the interactive cutaway viewing system because it can provide the easy and intuitive location of the cut.

3- Generation of the cutaway model

3.1 – Overview

Closed two-manifold triangular meshes are used to represent all objects in the input model and the cutting volume in our system. For a brief representation of the triangles constructing the cutaway model, we define some notations of the triangles as follows.

- $T_{In}(A, B)$: The triangles of the A, which are inside the B.
- $T_{Out}(A, B)$: The triangles of the A, which are outside the B.
- $T_C(A, B)$: The triangles of A, which collide with the B.
- $T_{In/Out}^{CS}(A, B)$: The split $T_C(A, B)$ along the intersection-lines inside/outside the B.

The cutting volume is shown by Cv. The object colliding with the Cv, and collision-free object are denoted by O^C and

O^{CF} respectively. As shown in Figure 3, using these notations, a set of triangles of a cutaway model is represented by

$$T^{Cutaway} = T_{Out}(O^{CF}, Cv) \cup T_{Out}(O^C, Cv) \cup T_{In}(Cv, O^C) \cup T_{Out}^{CS}(O^C, Cv) \cup T_{In}^{CS}(Cv, O^C).$$

The basic algorithm of the cutaway model generation is shown in Figure 4. First, collision detection between objects and the Cv are performed, and all objects are classified as collision-free objects (O^{CF}) and collision objects (O^C). Next, the inclusion test is applied to the O^{CF} , and $T_{Out}(O^{CF}, Cv)$ is obtained. For the O^C and the Cv , the intersection-lines are extracted from the collision triangle pairs. Then, each triangle of the O^C and the Cv except for collision triangles are classified into inner and outer triangles ones. As a result, $T_{Out}(O^C, Cv)$ and $T_{In}(Cv, O^C)$ are obtained. Finally, each collision triangle is split along the intersection-lines, which produces $T_{Out}^{CS}(O^C, Cv)$ and $T_{In}^{CS}(Cv, O^C)$. In this method, the axis-aligned bounding box (AABB) tree is created for each object in order to achieve efficient collision detection, the inclusion test and so on.

3.2 – Collision detection and inclusion test

Collision detection between the Cv and each object is done using the AABB tree and hierarchical collision tests [B1]. As a result, each object is classified into collision objects (O^C) and collision-free objects (O^{CF}), and all of the pairs of collision triangles are obtained. The inclusion test checks whether each O^{CF} is inside or outside of the Cv . In this test, the number of the intersections between the surface of the Cv and a ray shot from a vertex of the O^{CF} is counted. If the number of intersections is odd, the O^{CF} is inside the Cv , and if even number, it is on the outside. In this test, the AABB tree of the Cv is used for the efficient intersection detection of the lay and the Cv . As a result, $T_{Out}(O^{CF}, Cv)$ is obtained.

3.3 – Extraction of intersection lines and classification of triangles

Segments of intersection lines are extracted from collision triangle pairs, then the vertices of collision triangles $T_C(O^C, Cv)$ and $T_C(Cv, O^C)$ are classified as inside- or outside-vertices according to the normal direction of the triangle which generates the nearest intersection point on their incident edges. This inside/outside information is assigned from the vertices to their neighboring triangles. By propagating the information to their neighbors successively using mesh connectivity, all collision free triangles of the Cv and the O^C are classified into outside triangles ($T_{Out}(O^C, Cv)$ and $T_{Out}(Cv, O^C)$) and the inside ones ($T_{In}(O^C, Cv)$ and $T_{In}(Cv, O^C)$).

3.4 – Splitting collision triangles

For the triangle in the $T_C(O^C, Cv)$, the closed loops of the intersection lines are generated by connecting the segments of the intersection lines and the segments of the edges of the triangle. Then the closed-loops are triangulated. In our system, the triangulation is done by polygon tessellation of GLU (the OpenGL Utility library). A similar process is done for triangles in the $T_C(Cv, O^C)$. As a result, $T_{Out}^{CS}(O^C, Cv)$ and $T_{In}^{CS}(Cv, O^C)$ are obtained.

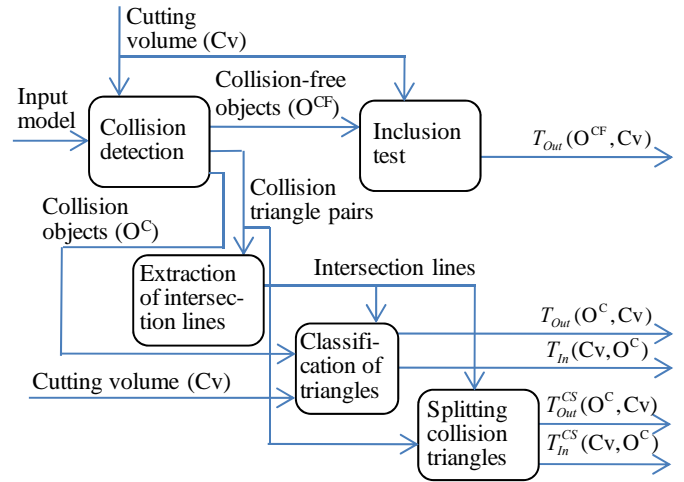


Figure 4: A cutaway model generation process.

Using the method described in this chapter, all triangles of the cutaway model are completed. Two examples of the cutaway models are shown in Figure 1. The time for model generation is shown in Table 1 and discussed in chapter 5 with the result of the extended method described in chapter 4.

4- Efficient cutaway model generation

In the interactive cutaway viewing system, keeping high frame rates is important. To reduce processing time for cutaway model generation and improve the frame rates, four methods are adopted in our system: 1) distance thresholding for reducing the number of collision and inclusion detection, 2) the use of time coherence for efficient triangle classification, 3) model simplification for accelerating many processes, and 4) adaptive cutaway model generation for reducing the number of splitting triangles.

4.1 – Distance thresholding

Inclusion and collision detections do not necessarily apply to the object far from the cutting volume. For the model with many parts, applying them at every frame causes the lower frame rate. In our system, thresholding for the distance between the object and cutting volume is used for finding the objects efficiently, which have the possibility of colliding with the cutting volume.

For making an efficient distance calculation, we use the bounding sphere centered at the barycenter of the bounding box of the object. The distance between the object and the Cv is calculated by using the distance between the centers of the bounding spheres of them. The object i has the possibility of colliding with the Cv , if $\|\mathbf{c}_{Cv} - \mathbf{c}_i\| \leq r_{Cv} + r_i$ is satisfied, where \mathbf{c}_{Cv} and \mathbf{c}_i are the centers of bounding spheres of the Cv and the object i , r_i and r_{Cv} are the radiuses of them.,

Distance thresholding is done before collision detection. All objects are classified into the distant objects without collision and the objects with the possibility of collision. Distant objects become a part of the cutaway volume as $T_{Out}(O^{CF}, Cv)$ without using an inclusion test.

4.2 – The use of time coherence

In triangle classification, first, the vertices of the collision triangles are classified into the inside and outside ones. Then, the in/out information of them is propagated through the neighboring triangles successively. This process is done in each frame and it is one of the causes of low frame rate in the model consisting of objects with a large number of triangles.

Time coherence occurs within changes of the triangles' in/out information between the sequenced frames. In other words, the number of the triangles whose in/out information change between the frame t and $t+1$ is small. Actually, such triangles are the ones passing the boundary of the cutting volume between the successive frames, and they exist near the collision triangles at each frame, as shown in Figure 5. Therefore, the in/out information of the triangles can be updated locally from the collision triangles. The sets of collision triangles at the current and previous frame are denoted as T_C^t and T_C^{t-1} .

First, the in/out test using ray intersections is applied to the vertices of the triangles in T_C^t . The resulting in/out information is assigned to their neighboring triangles. Then, the information is propagated to neighboring triangles which have different in/out information (Figure 5(b-c)). If there are non-updated triangles in the T_C^{t-1} , then the same process is used on them.

4.3 – Model simplification

A large number of triangles of the input models require many computations in almost every process, and reducing it is effective to improve the frame rates. In our system, the discrete LOD technique [LR2] is used. Some models with different resolutions and details are prepared for each object in advance. While the users view the cutaway model interactively, the models are switched according to the degree of interest and attention for the object.

4.3.1 –Simplification method

To the mesh with a relatively large number of triangles, the mesh simplification is applied for creating models with different resolutions. Although many methods for mesh simplification have been proposed [LR2], we adopted the quadric error based method [GH1] for mesh simplification. In the method, edge collapse operation is iteratively applied to the given mesh. An example of mesh simplification is shown in Figure 6.

For the primitives with a few number of triangles, such as cube with 12 triangles, the mesh simplification does not work well. Therefore, we use simple primitive simplification, which is simply done by removing the thin or small faces. The face removal rule for the rectangular parallelepiped is shown in the Figure 7. In our system, the rectangular parallelepiped is automatically recognized by checking the number of triangles and pairs of the parallel planes, and three distances between the parallel planes are extracted. By comparing the distances of planes with a given threshold, simplified versions are assigned to them. An example of primitive simplification is shown in Figure 8.

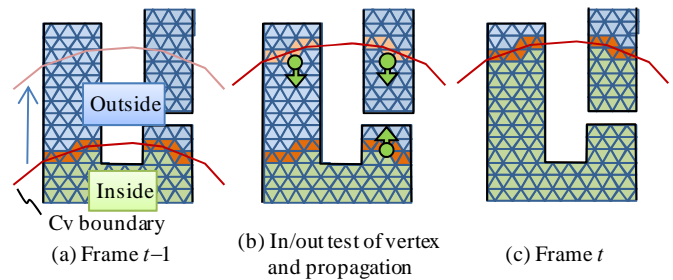


Figure 5: Classification of the triangles.

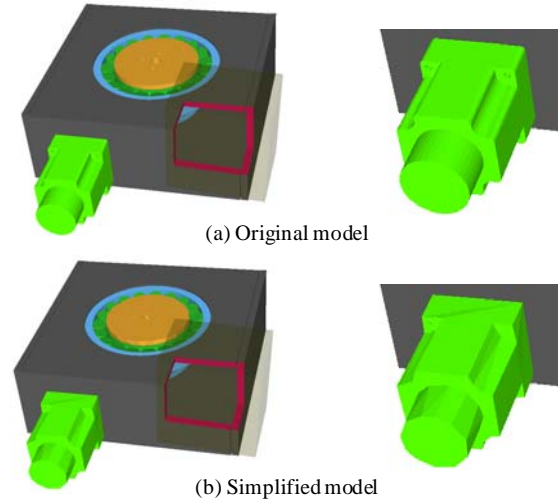


Figure 6: Simplification of the index table model.

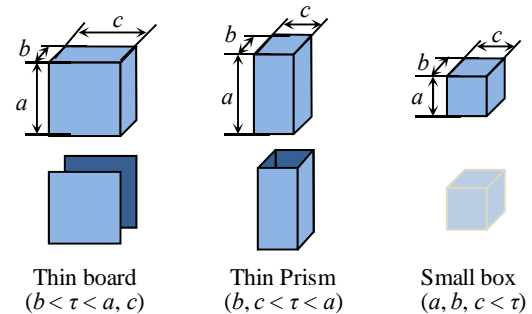


Figure 7: Simplified representations of the primitives.

4.3.2 –LOD control

In order to switch the models for changing LOD, the criterion is required. In our research, we define three measures of the user's attention and interest for the object i .

Distance from the cut: D_i

The region of interest of the user may be near the cutting portions. The distant region from there can be considered as the region with a low degree of attention. The distance measure D_i of the object i is defined as the distance between the centers of its bounding spheres and cutting volume, $D_i = \|\mathbf{c}_i - \mathbf{c}_{CV}\|$.

Area on the screen: S_i

The degree of attention may be low for the distant small objects from the viewpoint. We evaluate the degree of such attention for the object i by using the area of the bounding

sphere on the screen. It is calculated by $S_i = \pi (D_s r_i / D_i)^2$, where D_s is the distance between the screen and the viewpoint, and D_i is the distance between the object i and the viewpoint.

Relative velocity: v_i

When the user speedily moves the model and the cutting volume, the detailed cutaway model may not be needed to render because the users may find the appropriate location of the cut. We evaluate the relative velocity between the object i and cutting volume by taking the difference of the displacements of the centers of bounding spheres. It is calculated by $v_i = \|(\mathbf{c}_i^{t-1} - \mathbf{c}_i^t) - (\mathbf{c}_{CV}^{t-1} - \mathbf{c}_{CV}^t)\|$, where $\mathbf{c}_i^t, \mathbf{c}_{CV}^t$ represent the centers of bounding sphere of the object i and cutting volume at frame t .

Three measures are calculated for each object at the current frame. Then, the levels of the LOD for each measure, L_i^D, L_i^S, L_i^v are determined by comparing them with the given thresholds, where, the higher level corresponds to the detailed higher resolution object. Finally, the level of the LOD for the object i is determined by $L_i = \min(L_i^v, \max(L_i^D, L_i^S))$ (i. e., the higher level for distance and area is chosen, but if the movement of the model and the Cv is fast, the level becomes low). At each frame, the simplified version corresponding to the L_i is used for cutaway model generation.

4.5 – Adaptive cutaway model generation

Splitting triangles is one of the most time consuming processes. In order to reduce the number of splitting, it is adaptively applied to the triangles in the region used for the render and haptic force feedback, that is, the cutaway model is generated partially.

In the location mode, the model is used for only rendering in the display, without haptic force feedback. Therefore, only visible collision triangles are split. The visible collision triangles are front-facing triangles of the input model and back-facing triangles of the Cv. They are obtained by a visible test, which checks the normal direction of each triangle and the eye direction. Visible collision triangles are split and rendered, and the others are not split.

In the focusing mode, the haptic force feedback is also required for touching the surface of the cutaway model. For efficient haptic force feedback, it is effective and enough to use the triangles near the device's point. Therefore, only collision triangles near the device's point are split. In our system, such triangles are found using the search-sphere centered at the device's point, and the AABB trees of each object and the Cv. Collision triangles in the search-sphere are split and used in the force feedback together with the collision free triangles in the search-sphere.

When the mode switches from the location to the focusing, the partial cutaway model at the final state of the location mode becomes the initial model in focusing mode. In focusing mode, the cutaway model is progressively grown from the initial model, depending on the change of the viewpoint, orientation of the model and position of the device's point.

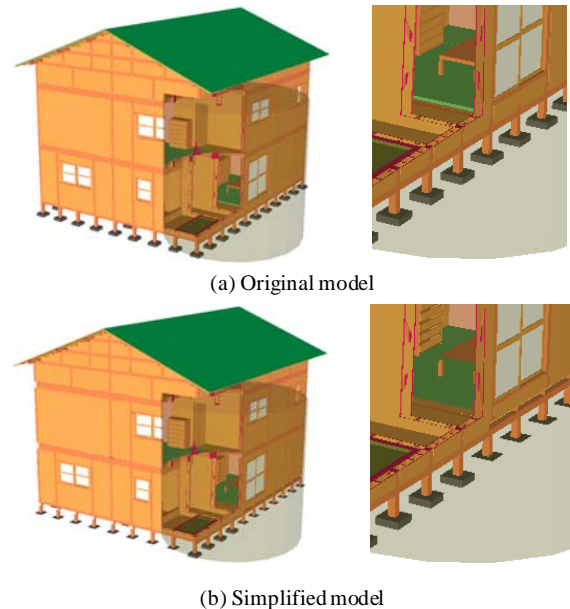


Figure 8: Simplification of the house model.

5- Results and discussions

The cutaway model generation method is applied to three types of the models shown in Figure 9. Eight bunnies model consists of relatively small numbers of the objects (eight) and each object has relatively large number of triangles (twenty thousands, high resolution) and high shape complexity. The house model consists of relatively large number of objects (eight thousands), but each part is very simple, rectangular parallelepiped with twelve triangles (low resolution). The index table model has the intermediate properties of these two models. Using these models, the influence of the differences of the scene complexity and the type of objects (natural or artificial, shape complexity and mesh resolutions) on our approach is shown. Cutting volumes used in the experiments are also shown in Figure 9 (semi-transparency, cubes for the eight bunnies and the index table models, and a cylinder for the house model).

In the LOD using mesh simplification, three simplified versions are made for each object. The numbers of triangles of each model are determined by the maximum of the normalized approximation error e_s , which is the ratio between the quadric error of edge collapse and the radius of the bounding sphere. For example, simplified versions of the bunny have 2,000 triangles at $e_s = r_i/500$ and 888 triangles at $e_s = r_i/50$. The thresholds of the each measure for switching the models are determined experimentally.

Models and cutting volume were moved along a given user-specified path and orientation, and processing times for cutaway model generation and the FPS of cutaway viewing were measured on a windows PC (Win-XP) with Xeon3GHz CPU, 2GB RAM, and ATI Quadro FX3500 graphics card. Their averages are shown in Table 1.

For every model, it was observed that the average frame rates of cutaway viewing were improved (120~351%) by using the methods described in section 4 compared with the previous

method described in section 3. The results are summarized as follows: the distance thresholding is effective for the model consisting of the large number of the objects, but the time of shortening is short. Local in/out information updates, using the time coherent, is effective especially for the model involving the objects with a large number of triangles. However, for the model consisting of a large number of simple objects, it may take additional time because the method references the state at the previous frame. Adaptive cutaway model generation is effective for any type of the model. The model simplification can improve the overall process in the cutaway model generation and rendering. Mesh simplification is especially effective. The effect will depend on the number of pre-defined simplified versions, and thresholds for the LOD. Finding the appropriate execution parameters (thresholds, number of simplified versions) is included in the future works.

6- Conclusions

An interactive cutaway viewing system is developed to efficiently understand the complex 3D geometric models. The interface of the system consists of two 6DOF input devices and a glassless stereoscopic display. The features of our system are summarized as follows: the simultaneous positioning of the model and cutting volume, and the stereoscopic display allows the user to intuitively determine the location of the cut. The input model can be cut using a cutting volume with arbitrary shape by applying the Boolean difference operation on the triangular meshes. The user can understand the model by using stereoscopic view and touching the surface of the cutaway model with haptic force feedback.

In this paper, four methods for improving the frame rate of the cutaway viewing were proposed. Through the experiments using different types of the model, followings were observed; Distance thresholding is effective for the model consisting of a lot of parts. A local update of the triangle's in/out information based on the time coherence is effective for the model including objects with a large number of triangles. Model simplification and partial model generation can stably improve the frame rate regardless of the type of the model. In our experiments, the rate of the improving the frame rate was 120 - 351%.

Future work includes determining the appropriate execution parameters automatically, and applying the system to the dynamic scenes with deformable objects.

7- References

- [AD1] Adams, B. and Dutre, P. Interactive Boolean Operations on Surfel-bounded Solids. *ACM Transactions on Graphics*, 22 (3): 651 – 656, 2003
- [B1] Bergen G. Collision Detection in Interactive 3D Environments, Morgan Kaufmann, 2004
- [BF1] Burns, M. and Finkelstein, A. Adaptive Cutaways for Comprehensible Rendering of Polygonal Scenes. *ACM Transactions on Graphics*, 27(5): 124:1-124:9, 2008
- [BK1] Biermann, H., Kristjansson, D. and Zorin, D. Approximate Boolean Operations on Free-form Solids. *Proc. ACM SIGGRAPH2001*, 185 – 194, 2001

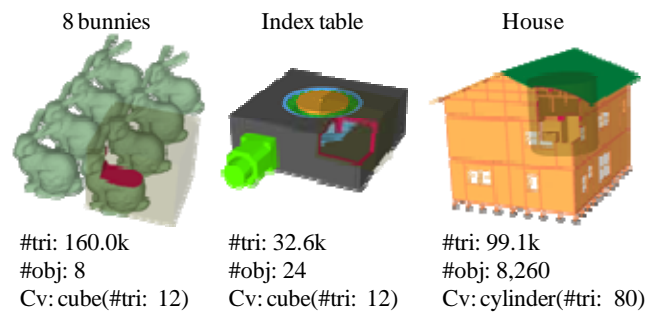


Figure 9: Geometric models used for evaluation.

Table 1 : Average processing times (msec.)

Process	8 bunnies		Index Table		House	
	previous method	proposed method	previous method	proposed method	previous method	proposed method
Distance thresholding	---	0.004	---	0.007	---	2.194
Collision detection	0.613	0.349 (57.0%)	0.360	0.238 (65.9%)	19.174	15.213 (79.3%)
Inclusion test	0.009	0.009 (98.5%)	0.018	0.017 (94.6%)	6.858	6.582 (96.0%)
Extraction of intersection lines	1.329	0.713 (53.7%)	0.675	0.392 (58.2%)	11.248	11.006 (97.8%)
Classification of triangles	5.012	1.967 (39.2%)	0.749	0.124 (16.6%)	0.229	0.295 (128.8%)
Splitting collision triangles	14.068	4.806 (34.2%)	5.849	2.013 (34.3%)	14.951	9.787 (65.5%)
Total time of Cut operation	21.031	7.849 (37.3%)	7.670	2.791 (36.4%)	52.460	52.077 (99.3%)
Rendering and haptics	173.08	47.388 (27.4%)	34.768	14.079 (40.5%)	121.000	99.111 (81.9%)
FPS	5.154	18.104 (351.3%)	23.54	59.275 (251.6%)	5.765	6.935 (120.3%)

[DW1] Diepstraten, J., Weiskopf, D. and Ertl, T. Transparency in Interactive Technical Illustrations. *Computer Graphics Forum*, 21(3): 317-325, 2002

[FD1] Furukawa, H., Date, H. and Onosato M. An Interactive Cutaway Model Representation System using a Three-Dimensional User Interface, *Proc. Design Engineering Workshop 2009*, 138-142, 2009

[GH1] Garland M. and Heckbert P.S. Surface Simplification using Quadric Error Metrics, *Proc. SIGGRAPH'97*, 209-216, 1997

[HK1] Hachenberger, P. and Kettner, L. Boolean Operations on 3D Selective Nef Complexes: Optimized Implementation and Experiments. *Proc. the 2005 ACM symposium on Solid and physical modeling*, 163 – 174, 2005

[HR1] Hable, J. and Rossignac, J. Blister: GPU-based rendering of Boolean combinations of free-form triangulated shapes. *Proc. ACM SIGGRAPH 2005*, 1024 – 1031, 2005

[LA1] Li, W., Agrawala, M., Curless, B. and Salesin, D. Automated Generation of Interactive 3D Exploded View Diagrams. *ACM Transactions on Graphics*, 27 (3): 1-7, 2008

[LR1] Li, W., Ritter, L., Agrawala, M., Curless, B. and Salesin, D. Interactive Cutaway Illustrations of Complex 3D Models. *Proc. ACM SIGGRAPH 2007*, Article No. 31, 2007.

[LR2] Luebke D., Reddy, M., Cohen, J., Varshney, A., Watson, B. and Hoebner, R. Level of Detail for 3D Graphics, Morgan Kaufmann, 2003

[PC1] Pavic, D., Campen, M. and Kobbelt, L. Hybrid Booleans, *Computer Graphics Forum*, 29(1): 78-87, 2010