

RibbonView: Interactive Context-Preserving Cutaways of Anatomical Surface Meshes

T. McInerney and P. Crawford

Dept. of Computer Science, Ryerson University, Toronto, ON, Canada, M5B 2K3

Abstract. We present an interactive visualization tool that provides users with the capability of cutting away the surfaces of enclosing objects to reveal interior or occluded objects. This tool, known as *RibbonView*, is applied by mimicking the sweeping action of a paint roller across the surface of a smooth object. This virtual analogy of a familiar real-world action not only removes the occluding surface but also generates a series of polygonal strips or *ribbons* that provide an effective contextual outline view of the removed material. The cutaway tool can be used for many types of cuts, uses a single, consistent interaction style, and the cut region is easily editable. We apply RibbonView to several human anatomy data sets to demonstrate its ease of use and effectiveness.

1 Introduction

Visualizing, examining and measuring anatomical structures embedded in 3D medical images are key tasks in radiology. In surgical planning, surgeons must ascertain the complex shape and organization of these structures to find optimal approaches to a target structure or to plan a series of surgical actions. In general, the explosion of large 3D datasets in medicine and other application domains such as architecture, entertainment, engineering and manufacturing, has resulted in the need for effective visual communication styles as well as simple, intuitive interactive data exploration tools which allow users to not only view the data, but comprehend it.

When inspecting and manipulating complex systems of solid 3D objects, it is critical to provide users with the capability of seeing through the surfaces of enclosing objects to reveal interior objects or parts. Users also need to understand and measure the spatial relationship between objects in the system and may need to spatially overlay two or more similar objects in order to compare their shape and size. Providing these capabilities is a challenging interactive 3D visualization problem. Ideally one would like to mimic the real world actions of simply “reaching out” and “cutting away” occluding material in order to examine a hidden region or object sub-part. Furthermore, it is imperative that this region *focusing* capability is not achieved at the expense of a *contextual* view of the surrounding or occluding structures.

While the use of semi-transparency on an outer surface can reveal the interior of an object, the generated images tend to be noisy and do not effectively convey the spatial relationship or depth of interior objects, especially if there are multiple layers of transparency. Simply cutting away parts of the occluding objects is debatably a more effective approach, but the user is forced to mentally “fill in the gap” of the removed

material and estimate its shape. Ideally it would be desirable to cut away the outer surface while still rendering some outline of it that simultaneously minimizes occlusion and provides effective visual depth and spatial relationship cues of the interior. Furthermore, if this outline is aligned with the natural geometry of the outer object surface (e.g. along primary medial axes, ridge lines, primary curvature lines etc.) then the user can more readily mentally reconstruct the missing geometry.

Although these classic focus-plus-context visualization issues are important, allowing users to easily *generate* object-aligned cutaway regions is equally important. Radiologists, surgeons and medical technicians are very knowledgeable about anatomy. However, given their considerable workloads, using the complex and restrictive interfaces too often associated with common medical visualization software is counterproductive. The goal is to have these expert users focus on their specific visualization task and not on how to use the visualization tools. To realize this goal requires the creation of simple but powerful interaction metaphors.

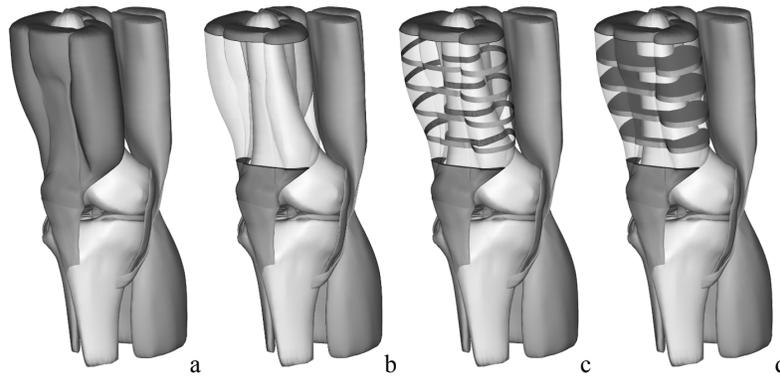


Fig. 1. Knee data set showing: a) surfaces of muscles, tendons, and bone, b) cutter extruded along muscle and resulting cutaway material shaded transparently, c) ribbon view of cutaway material, d) solid “slice” view of cutaway material

In this paper we describe RibbonView - a cutaway visualization tool that mimics the sweeping action of a hand-held paint roller across the surface of a smooth object. In RibbonView, this virtual analogy of a familiar real-world action does “double duty”. Sweeping the cutaway tool along an object surface not only removes the occluding surface but also generates a series of polygonal strips or *ribbons* outlining the removed material (Fig. 1c). The sweeping action is simple and intuitive - a user simply pulls a rectangular strip over the surface of the object along a curving path aligned with the natural shape lines of the object. As the strip is pulled across, it is continuously extruded to form a quadrilateral cutter mesh orthogonal to this path. Thus, the very action of using the cutaway tool, coupled with its explicit quadrilateral mesh-based representation, automatically establishes a contextual outline view of the enclosing object. The cutaway tool can be used for many types of cuts while adhering to a single, consistent interaction model. Furthermore, since the cutter mesh corres-

ponds directly to a cutaway region, cuts are not only easily created, but easily edited as well. In the remainder of this paper, we describe RibbonView and its interaction model in more detail. We apply it to several human anatomy data sets to demonstrate its effectiveness and provide a discussion of its ease of use and efficiency compared with other techniques.

2 Related Work

Anatomical structures often have complex shapes and systems of these structures have complex spatial interrelationships. Medical volume images containing these structures are commonly viewed through direct volume rendering or by segmenting and extracting the surfaces of the anatomical structures from the volume, generating polygonal surface meshes. Although we concentrate on polygonal meshes in this paper, it is nonetheless insightful to first briefly review cutaway techniques and context-preserving volume rendering algorithms for volume images. We will then review recent cutaway techniques for polygonal meshes.

2.1 Volume Image Cutaway and Rendering

A simple and common technique to reveal hidden structures in medical volume images is the use of one or multiple clip planes, generating image slices. Radiologists are adept at scanning through volume images on a slice by slice basis and mentally reconstructing 3D shape. While the use of 2D slice images obviously does not directly convey 3D shape, slices are useful for exact measurements of depth and/or distance between objects.

Clip planes support only a limited range of cut geometries. Weiskopf et al. [1] extend the range of volume clipping by interactively applying convex and concave clipping objects, carving the away chunks of the volume image. Konrad-Verse et al. [2] present a virtual resection tool based on a deformable cutting mesh. Chen et al. [3] proposes a variety of complex volume manipulation tools that allow drilling, lasering, and peeling operations.

Another approach to viewing interior structures in volume images is through the use of illustrative volume rendering techniques [4], [5], [6]. Essentially these techniques attempt to reduce the opacity in less important regions in order to visualize interior and exterior structures of a focal region while also preserving and enhancing important shape cues. This "context-preserving" volume rendering model is often a function of shading intensity, gradient magnitude, distance to the eye point, and previously accumulated opacity.

Several research groups have also been exploring the use of deformations to visualize volume images [7], [8], [9]. McGuffin et al. [7] proposed an interactive system to browse pre-labeled iso-surfaces in volume data by deforming them according to simple interaction metaphors. Users can cut into and open up, spread apart, or peel away parts of the volume in real time using widgets, attempting to retain surrounding context. Mensmann et al. [8] also use a deformation approach to simulate surgical cutting

of tissue to reveal tissue underneath. Finally, Correa et al. [9] use a space warping technique to simulate peelers, retractors, pliers, and dilators.

2.2 Polygonal Mesh Cutaway

Many segmentation algorithms are commonly used to label the voxels associated with anatomical structures embedded in medical volume images. The surfaces of these structures can then be extracted; resulting in systems of 3D closed (i.e. "solid") polygonal surface meshes. Another class of segmentation algorithms known as deformable models, can typically directly generate polygonal meshes.

The primary techniques used to examine occluded interior structures, whether through direct volume rendering of the volume images or through the use of extracted polygonal meshes, are cutaway views and ghosted views (i.e. semi-transparency). For example, Diepstraten et al. [10] used transparency for technical illustrations.

The most common technique to implement cutaways of polygonal meshes is Constructive Solid Geometry (CSG). RibbonView uses the GTS library [11] to perform CSG operations as it allows us to maintain explicit mesh representations of both the object and the cutaway piece.

Several researchers have developed cutaway tools for surface meshes. Coffin et al. [12] present an interactive technique to cut holes in occluding geometry using a user-defined cutout shape or a standard shape. Diepstraten et al. [13] presented different methods to create cutaways in polygonal data. In Knodel et al. [14] users generate cutaways using simple sketching actions and then refine the shape of the cut using widgets. Li et al. [15] presented a system for authoring and viewing interactive cutaway illustrations of complex 3D models, targeted towards medical and technical education rather than for radiology or surgical planning.

3 RibbonView

In our cutaway approach we attempt to provide a single intuitive interaction metaphor that can be used for many different types of cuts. We employ a paint roller metaphor (Fig. 2) where the user grabs the "handle" and pulls the "roller" across the surface of the object, *extruding* a quadrilateral cutter mesh to define the cutaway region. This surface-constrained interaction is simple, fast and comfortable.



Fig. 2. A paint roller is used as the underlying interaction metaphor of the RibbonView tool

One advantage of the extrusion interaction model is the user is provided with immediate visual feedback of the cutter mesh – and hence the cutaway region – as it is created. Furthermore, users can stop at any time, back up (un-extrude) or make adjustments to the cutter height and depth (see Sect. 2.3).

However, the primary advantage of the extrusion process is the leading quadrilateral edge (the “roller”) of the cutter is constrained to remain orthogonal to the extrusion path defined by the cursor (the “handle”) (Fig. 2, right) as the user pulls it along with the mouse. The extrusion path can be easily made to coincide with medial axes of objects or ridge lines, or other curving feature lines. The result is a cutter mesh that essentially consists of a connected series of quadrilateral polyhedrons (loosely referred to as “boxes”) oriented orthogonally with respect to these shape feature curves. These boxes provide a convenient framework for the construction of the context preserving ribbons.

The ribbon alignment and ribbon shading provide an effective visual outline of the cutaway region, allowing users to more easily mentally reconstruct the original surface while they are simultaneously viewing the interior. In the following sections, we describe the extrusion process and the subsequent construction of our cutter mesh. We then describe the various types of cuts that can be performed as well as the cutaway region rendering styles supported. Finally, we describe the editing operations that can be performed on the cutter mesh.

3.1 Cutter Model Representation and Construction

The cutter mesh is represented as a 3D closed mesh of quadrilaterals (Fig. 3d). It can be visualized as a connected series of polyhedrons (“boxes”) (Fig. 6a). The user also has the option of using the cutter as a control mesh for an interpolating subdivision surface [16] (Fig. 3e). The subdivision surface is used when performing freeform cuts (Fig. 3h) or rounded cuts. Maintaining an explicit mesh representation of the cutter as well as of the cutaway region allows us to maintain complete control over the rendering styles and also allows for subsequent manipulation and measurements (Sect. 3.3).

To create a cutter – and hence define the cutaway region – the user first establishes the height of the “roller” by clicking on a surface point of the object¹ and stretching out an initial narrow and thin rectangular box (Fig. 3a). The user can slide the free end of the initial box around (Fig. 3a–b) to fine tune the roller height and orientation.

¹ The user can also click on a point off of the object to create a cutter that completely surrounds a region of the object.

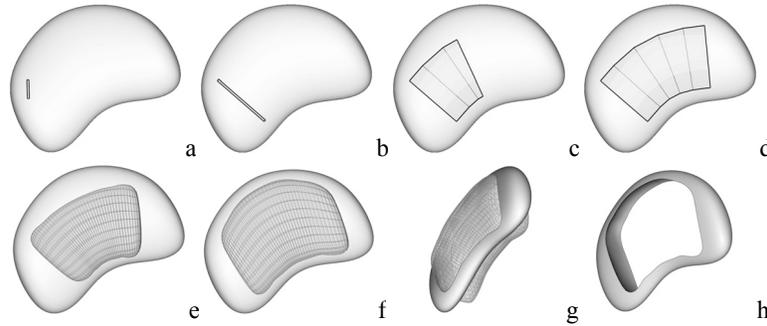


Fig. 3. To create a cutter mesh, the user extends and orients an initial thin rectangular box “strip” (a–b) and then extrudes it by pulling the right face across the object surface (c–d). New boxes are automatically created and connected to the cutter mesh. The mesh is thickened and can be subdivided and edited, if desired, resulting in a freeform shape cut (e–h)

Once the initial cutter box is established, the user pulls on the right-hand face and smoothly widens it (Fig. 3c). When the box reaches a user-defined width (set via a slider), a new box is created and connected to the cutter mesh (Fig. 3c–d). As the new box is pulled and widened across the surface of the object, the front face of the box may “collide” and intersect with the object surface. This collision is detected and the front face is rotated around its shared front edge until it no longer collides. In effect, this action causes the front face to be constrained to “stick” to the object surface. This widening, rotation and box generation process happens quickly and smoothly - the user is unaware of the underlying collision detection and response algorithm and sees only the continuous cutter extrusion.

Once the cutaway region has been defined with the cutter mesh, the mouse scroll wheel is used to “thicken” the cutter to the desired depth (Fig. 3g). The cut is then activated with a key press (Fig. 3h). Since parts of the object may be thicker than others, a small amount of unwanted object material may remain after the cut is made. The user can easily edit the depth of the cutter mesh in the thicker region and re-cut.

3.2 Cutaway Types Supported

Various cuts are naturally supported by the combination of the extrusion process, the quadrilateral cutter mesh and the subdivision process. Each cut is described below:

Curving Path: The default cutaway supported is a curving path cut. As described previously, the curving path cut is used to cut away material along the natural shape lines of an object. The cut can either wrap around a section of the object (Fig. 1), cut away part of the object (Fig. 4a, b), or cut a “window” in the object (Fig. 3h).

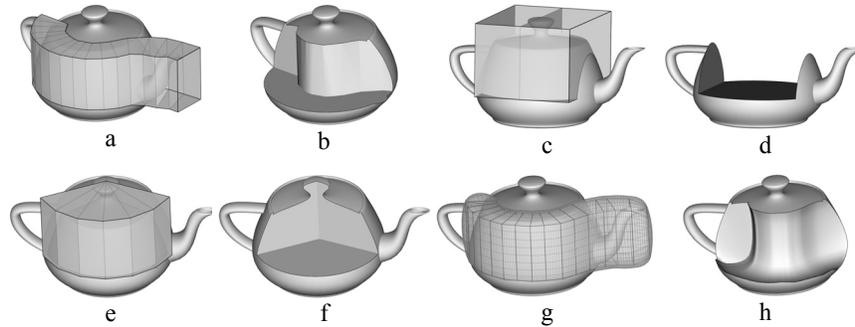


Fig. 4. Cut types supported: (a,b) curving path cut, (c,d) straight path cut (“box” cut), (e,f) surface of revolution cut (“wedge” cut), (g,h) freeform cut

Straight Path: With the click of a button, the surface-sticky curving cutter extrusion path can be constrained to continue along a 3D straight line. This simple constraint supports the creation of standard volumetric cuts such as box cuts (Fig. 4c,d). The cutter can also switch at any time between curving paths and straight line paths.

Wedge Cut: Many anatomical structures are disc-shaped or tubular, with a well-defined single primary medial axis. By simply clicking and constraining one end of the initial cutter strip, surface of revolution or “wedge” cutters can be extruded to open up these structures (Fig. 4e,f). The free end of the cutter strip can be swept over an arbitrary range of angles, from narrow wedges to complete surfaces of revolution.

Freeform Cut: Freeform cuts are performed using a subdivision surface cutter (Fig. 3h). Rather than using a tedious and error-prone tracing process, freeform cuts are performed by first extruding a rough cutter shape across the target region of an object surface as usual (Fig. 3d). With the click of a button, the cutter mesh is then used as a control mesh and automatically subdivided with a tightly-interpolating subdivision process. The control points of the mesh can then intuitively pulled, in a constrained manner, to quickly refine the freeform shape (Fig. 3f).

Curved or “Rounded” Cut: The subdivision surface can also be used to generate cutter meshes that are rounded (Fig. 4g, h). As the depth of the cutter can be locally controlled along its length, rounded cuts can be generated such that an interior object is fully exposed without cutting all the way through the exterior object.

3.3 Context-preserving Rendering Styles of the Cutaway Region

The extrusion/sweeping interaction model combined with the explicit quadrilateral mesh representation of the cutter and the use of GTS, provide the means for very flexible, context-preserving rendering styles. For example, since GTS outputs an explicit mesh representation of the cutaway region, we can render the cutaway region as semi-transparent (Fig. 5c). In addition, GTS partitions the cutaway region into several components, including the part of the object contained within the cutter mesh and vice versa. This flexible scheme allows us to render each of these components

separately or in combination. For example, the part of the object contained within the cutter mesh – the “walls” – can be shaded and outlined in a different color (Fig. 5a).

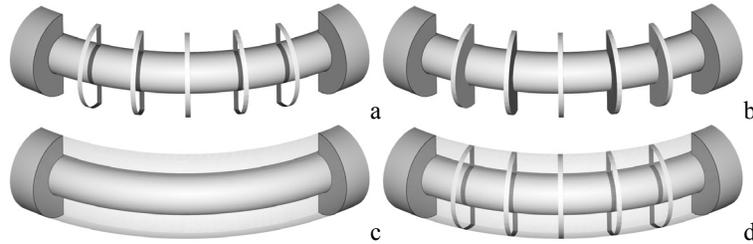


Fig. 5. The extrusion/sweeping interaction model combined with the explicit quadrilateral mesh representation of the cutter and the use of GTS provides a convenient framework upon which to construct different context-preserving rendering styles

The quadrilateral parameterization of the cutter mesh allows us to construct a series of ribbons. Ribbons are constructed by interpolating cross-sectional “slices” between the cutter mesh boxes and subsequently constructing thin, quadrilateral box meshes centered on each slice. The width of the box mesh can be varied to control ribbon width. The thin box meshes and the object are then fed to GTS, causing only the part of the object contained within each of these boxes to be output. This has the effect of removing only ribbon-shaped sections of the geometry. These ribbons can then be rendered alone (Fig. 5a) or combined with a transparent intersection surface (Fig. 5d). Furthermore, the explicit mesh representation of the ribbons provides a means for separately rendering the outside and inside of the ribbon mesh (Fig. 5a).

Finally, we are also able to generate solid cross-sectional slices (Fig. 5b). Solid slices are constructed in a manner similar to ribbons except that the complete intersection geometry is computed by GTS, rather than just the outermost surface.

In summary, maintaining explicit representations of both the input and output of the CSG operations, combined with the parameterization of these representations, provides a framework upon which effective visual communication styles can be built. We also plan to explore non-photorealistic rendering styles for additional visual cues.

3.4 Cutter Mesh Editing

As described in section 3.1, the cutter is a closed quadrilateral mesh and can be loosely thought of as a series of joined boxes (Fig. 6a). This representation supports constrained editing of the cut region, preventing the user from creating ill-defined cutters while still providing cutter shape flexibility. For example, to locally modify the height of the cutter, the user selects a cutter cross-sectional “slice”, corresponding to a box inner boundary face (or end-cap face), and drags the top edge of the slice to extend, shrink, or reposition it. During this interaction the slice rectangle is constrained such that it cannot intersect a neighboring slice rectangle, while at the same time always remaining planar.

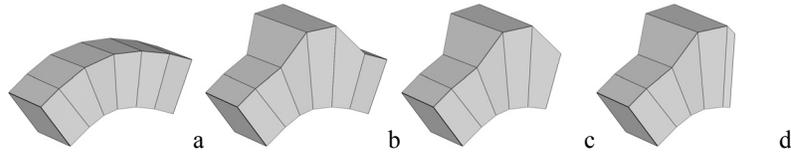


Fig. 6. The shape of the cutter mesh (a) can be intuitively edited in a constrained fashion to prevent the creation of degenerate cutters. In b) the height and depth have been locally modified. In (c–d), the cutter has been un-extruded

Similarly, the user can use the mouse scroll wheel and quickly “thicken” the entire cutter mesh, cutting more deeply into, or entirely through, the object. The cross-sectional slices of the cutter may be angled with respect to each other (if, for example, the cutter was extruded along a sharply curving path). As the cutter is thickened these faces may intersect, resulting in a non-simple closed cutter mesh. An iterative procedural algorithm is continually invoked as the cutter is thickened to prevent cutter self-intersection. This algorithm detects intersecting slice rectangles and rotates them slightly away from each other, using the front edge of each slice rectangle as its rotation axis. In addition to global cutter thickening, the user is also able to select a cutter slice and thicken locally (Fig. 6b). Slice orientation and position may also be adjusted.

A critical factor of any cutaway tool is the ability to quickly and easily undo input actions. The RibbonView tool supports a simple but effective un-extrude action. The user simply presses a key and then un-extrudes the cutter by pulling the leading edge of the cutter in the opposite direction. The orientation and width of the current box can be adjusted or the entire cutter can be undone, box by box. A quick undo option uses the Ctrl-z keys to remove the last box of the cutter.

4 Experimental Results and Discussion

We have performed a series of experiments to demonstrate the effectiveness and ease of use of the RibbonView tool. All region selection operations (initialization, extrusion and thickening) were performed in real-time and most take only a few seconds to generate (Table 1). Fine tuning the region of interest adds a few more seconds. Once the region has been selected (and edited if desired), a key press activates the actual cut. Currently the GTS-supported cutaway operations are not GPU accelerated. However, a spatial data structure is used to speed up the cutting. Only a few seconds are required to perform complex cuts on an object with up to 500000 triangles.

In the first experiment we cutaway a portion of a muscle to expose the femur bone (Fig. 1). Unlike the use of transparency alone (Fig. 1b), note how the ribbons clearly show the wrapping of the muscle around the thigh bone (Fig. 1c). In the second experiment we perform a freeform cut of the skin to expose the skull (Fig. 7a). We then perform another freeform cut to cutaway part of the skull. This type of cut is used for planning some cranio-facial surgical procedures. In the third experiment we have extruded the cutter along the primary medial axis of the jaw and combined ribbons

with transparency to show the shape of the mandible (Fig. 7b). Solid slices can also be generated and used to compare and measure the shape of this jaw with another jaw data set or with the same post-surgical jaw. In the final experiment we create a wedge cut of the vertebra to reveal the disc underneath (Fig. 7c).

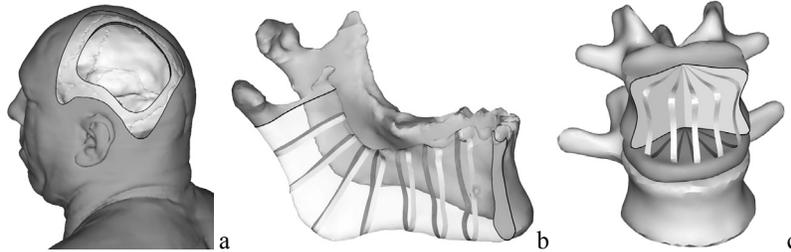


Fig. 7. Results of different cuts using the RibbonView tool. In a) a freeform cut of the skin and skull has been performed. In b) a curving cut along the primary medial axis of the jaw is demonstrated. In c) a wedge cut exposes the relationship of the vertebra to the disc underneath.

Table 1: User interaction times required to generate results in Figures 1 and 7

Fig.	Extrusion (sec)	Edit Time (sec)	No. Edits	Total Time (sec)
1	5	6	2	11
7-a	10	17	10	27
7-b	8	8	4	17
7-c	4	3	1	7

We also performed an informal user study on the ribbons and solid slices versus transparency. The study asked users to choose the visualization style that best illustrated the shape of the cutaway occluding surface, as well as the spatial interrelationship between the occluding and occluded surfaces. Several sets of images were presented to approximately 40 participants. Each set contained the original uncut image of a unique system of anatomical structures, and three cutaway images – each using one of transparency, ribbons, or solid slices. Overall, ribbons or solid slices were preferred on average 80 percent of the time.

4.1 Discussion

In many cut away algorithms, the ability to easily, efficiently and effectively construct, define and modify the cut region to obtain the optimal view of the interior is often inadequate. Our contention is that this inability is due to the choice of interaction model. For example, cutting- or tracing-based interaction models attempt to mimic the real world action of scissors (or a marker) to cut out (or outline) the region of interest. While this type of interaction may be useful for freeform cutout shapes, it is difficult to perform more structured, symmetric or volume-based cuts. Editing or modifying the cut region is also problematic. Backtracking can be performed or it

may be possible to convert the traced region to a spline curve for further editing. Tracing with a mouse can also be tedious – pen based systems or touch screens may be more efficient but there is the new problem of occlusion by the user’s hand.

Sketch- or gesture-based interaction is another metaphor that has been used in cutaway systems. In this scenario the user draws different curve shapes on the object surface and each curve is then interpreted by the algorithm to generate different cuts. For example, Knödel et al. [14] define four different curve gestures: line, circle, corner, and ribbon. One of the basic problems with this interaction metaphor is that users can’t immediately see the shape of the cutaway region they are creating. Another problem is after the initial cutting phase, the modification of a cutaway region is typically performed using a widget that is constructed from the sketched input curve. The implication is that users are now faced with two interaction models. Furthermore, widgets typically have rather restrictive editing capabilities.

Sculpting based interaction is another popular interaction model [1], [3]. Sculpting may be better suited to the shaping (i.e. generation) of objects, for example by digital artists, rather than the visualization of existing structures. The difficulty aligning or positioning the sculpting tool along natural shape lines of the object and the subsequent inability to easily or efficiently edit the cutaway region may limit the effectiveness of this interaction model. For example, it may be tedious to interactively explore/remove a region of the data around a target anatomical structure using unconstrained 3D positioning, and each cutting action is independent of the others.

Finally, widget based interaction models [7] are often limited to cutting away simple convex shapes and it may be difficult to produce elongated, curving cuts that are aligned with the natural shape of the object. Furthermore, several widget types are often provided and the user is forced to shift their focus away from the visualization task to decide which widget is most appropriate – an often nontrivial task.

The RibbonView interaction model is an attempt to design a model that minimized the deficiencies noted above. In our opinion, the two most critical design factors are the ability to easily position the cutter along natural shape features of the object using simple constrained input actions, and the ability to easily modify/fine tune the cut.

5 Conclusion

Using either semi-transparency or cutaways separately to examine occluded interior structures often results in an ineffective visual communication style. Some synthesis of the two techniques is perhaps necessary in order to understand the spatial interrelationships of a complex system of anatomical structures. In this paper, we have utilized ribbons to achieve this synthesis. The shaded ribbons coupled with a semi-transparent cutaway region not only maintains a contextual view but also clearly shows the spatial relationship of the back of the occluding surface and the front of the interior surface.

Furthermore the simple, fast and intuitive generation of a desired data view is equally as important as the visual communication style itself. In this paper, we have adhered to a single, consistent and intuitive extrusion interaction model and realized this interaction model by sliding, expanding, creating, and joining boxes along the surface of an occluding object. This design allows us to efficiently generate a cutaway

that is aligned with the natural shape features of an occluding object, with most cutaway operations completed within 10 – 15 seconds (Table 1).

We are currently exploring several improvements and extensions to the Ribbon-View system. For example, we are experimenting with the use of a multi-touch input device for RibbonView such that the user can literally reach out with their hands and sweep their fingers across the object surface to indicate the cutaway region. We also plan to transfer the cutting operation to the GPU to provide real-time cutting.

References

1. Weiskopf, D., Engel, K., Ertl, T., Interactive clipping techniques for texture-based volume visualization and volume shading, *IEEE Transactions on Visualization and Computer Graphics*, vol. 9, no. 3, 2003, 298–312.
2. Konrad-Verse, F.O., Preim, B., Littmann, A., Virtual Resection with a Deformable Cutting Plane, In *Proceedings of Simulation und Visualisierung 2004*, 203–214.
3. Chen, H.L. J., Samavati, F.F., Costa Sousa, M., GPU-based Point Radiation for Interactive Volume Sculpting and Segmentation, *The Visual Computer*, v24, no. 7–9, 2008, 689–698.
4. Zhou, J., Döring, A., Tönnies, K.D., Distance based enhancement for focal region based volume rendering, In *proceedings of Bildverarbeitung für die Medizin'04*, 2004, 199–203.
5. Viola, I., Kanitsar, A., Gröller, M.E., Importance-driven feature enhancement in volume visualization, *IEEE Trans. on Visualization and Comp. Graphics*, v11, n4, 2005, 408–418.
6. Bruckner, S., Grimm S., Kanitsar, A., Gröller, M.E., Illustrative context-preserving exploration of volume data, *IEEE Trans. on Vis. and Comp. Graphics*, v12, n6, 2006.
7. McGuffin, M.J., Tancu, R., Balakrishnan, R., Using Deformations for Browsing Volumetric Data, In *proceedings of IEEE Visualization*, Seattle, Wash., 2003, 401–408.
8. Mensmann, J., Ropinski, T., Hinrichs, K.H., Interactive Cutting Operations for Generating Anatomical Illustrations from Volumetric Data Sets, *Journal of WSCG -- 16th International Conference in Central Europe on Computer Graphics, Visualization and Computer Vision*, 2008, vol. 16, no. 1–3, 89–96.
9. Correa, C., Silver, D., Chen, M., Feature Aligned Volume Manipulation for Illustration and Visualization, *IEEE Trans. on Vis. and Computer Graphics*, v12, n5, 2006, 1069–1076.
10. Diepstraten, J., Weiskopf, D., Ertl, T., Transparency in Interactive Technical Illustrations, *Computer Graphics Forum* 21, 2002, 317–326.
11. SourceForge, The GNU Triangulated Surface Library, <<http://gts.sourceforge.net>>, 2006.
12. Coffin, C., Höllerer, T., Interactive Perspective Cut-away views for general 3D scenes, In *Proc. of IEEE Symposium on 3D User Interfaces*, Alexandria, VA, 2006, 25–28.
13. Diepstraten, J., Weiskopf, D., Ertl, T., Interactive Cutaway Illustrations, *Computer Graphics Forum* 21, v22, n3, 2003, 523–532.
14. Knödel, S., Hachet, M., Guitton, P., Interactive Generation and Modification of Cutaway Illustrations for Polygonal Models, In *Proc. Of the 10th Int. Symposium on Smart Graphics*, Salamanca, Spain, 2009, 140–151.
15. Li, W., Ritter, L., Agrawala, M., Curless, B., Salesin, D., Interactive Cutaway Illustrations of Complex 3D Models, In *Proc. of Int. Conf. on Computer Graphics and Interactive Techniques ACM Siggraph*, San Diego, CA, 2007.
16. Schaefer, S, Warren, J., A Factored Interpolatory Subdivision Scheme for Quadrilateral Surfaces, *Curve and Surface Fitting: Saint Melo 2002*, 2003, 373–382.