Aperio: A System for Visualizing 3D Anatomy Data Using Virtual Mechanical Tools

T. McInerney(✉) and D. Tran

Department of Computer Science, Ryerson University,
Toronto, ON M5B 2K3, Canada
tmcinern@ryerson.ca

Abstract. We present Aperio - an interactive real-time system for visualizing complex organic-shaped 3D models such as anatomy data or medical data. Aperio employs an interaction model based on a mechanical tool analogy via a small set of virtual "metal" tools, such as rods, rings, cutters, and scalpels. The familiar and well-differentiated tool shapes, combined with their initial pose and metallic appearance, suggest a tool's function to the user. Cutter tools are designed to create easily-understood cutaway views, and rings and rods provide simple oriented path constraints that support rigid transformations of models via "sliding", including interactive exploded view capabilities. GPU rendering provides realistic real-time "solid cut" previewing of surface-mesh models. We demonstrate Aperio using a human anatomy data set and present user studies to provide supporting evidence of Aperio's interaction simplicity and its effectiveness for visualizing model spatial interrelationships.

1 Introduction

Perceiving the spatial relationships between 3D geometric models and between model subsystems, and perceiving the hidden internal structure of complex geometric models are essential tasks in fields such as engineering, architecture, and medicine. Specifically, in medical education or surgical planning, anatomy data or medical data may consist of numerous model subsystems each of which may be composed of many individual curving anatomical structures that are often closely adjoined, intertwined, or enclosed. Therefore, in order to actively explore the data, objects and their constituent parts must be quickly and repeatedly rearranged relative to one another to reveal obscured relationships, and hidden objects must be revealed by creating contextual cutaway views.

This paper presents Aperio[1] (Fig. 1), an interactive system for exploring spatial relationships between geometric models (i.e. surface meshes) that uses an interaction metaphor based on a familiar mechanical tool analogy. Aperio utilizes a small collection of virtual metal tools, such as rods, rings, "cookie cutters", a scalpel and spreader (Fig. 1 left), to support a coherent set of intuitive, simple

[1] A video of Aperio can be viewed at www.youtube.com/watch?v=KcGfVjDlnpU.

G. Bebis et al. (Eds.): ISVC 2015, Part I, LNCS 9474, pp. 797–808, 2015.
DOI: 10.1007/978-3-319-27857-5_71

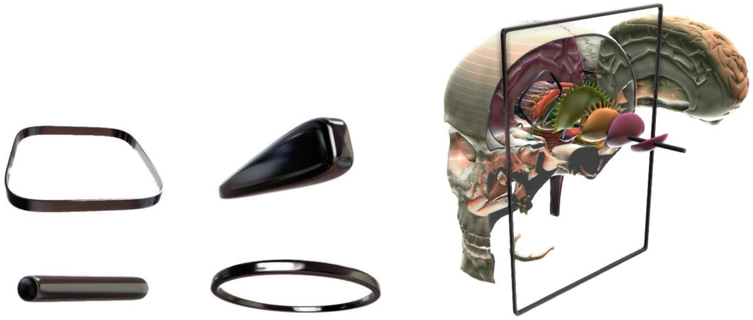

Fig. 1. Left: Aperio tools - “cookie” cutter, knife, ring, rod. Right: Using a combination of virtual mechanical tools to reveal spatial relationships between model parts.

and fast model rearrangements and model cutaways. The familiar shapes and metallic appearance of the tools provide strong visual cues to the user allowing tools to be quickly positioned and oriented unambiguously while also providing the user with a clear choice of which tool to use. Tools can be smoothly moved around the surface of the object models and dynamically oriented, scaled, and shaped. Model rearrangement, including exploded views, is performed by *sliding* models along the tools, akin to a beads on a wire. In the remainder of this paper, we describe Aperio, its interaction model, a description of each tool and its implementation. We demonstrate Aperio using a human anatomy data set and also present user studies to provide supporting evidence of Aperio’s interaction simplicity, controllability and effectiveness for visual analysis of this data type.

Aperio presents several contributions to occlusion management of curving, twisting and intertwining organic shape models created by digital artists or derived from 3D medical images. Firstly, the system combines model cutaway and real-time cutaway previewing, constrained rigid transformation control of individual models, model parts, and model subsystems, and exploded view capabilities, all under a single, coherent interaction model based on a mechanical/surgical tool analogy. The highly-controllable tools can be flexibly combined and repeatedly applied to model parts cut away from the original. Secondly, Aperio supports user-defined dynamically configurable curving explosion paths, enabling clear views of the spatial relationship between closely adjoined curving objects by not only moving them apart but also “opening” them up relative to each other. Thirdly, unlike many other systems, we maintain a rendering of the tools/tool outlines to visually reinforce the tool operations and to aid in the overall perception of model part relationships. A user study suggests this type of visual cue aids in visual analysis and also helps the user maintain an understanding of what operations have been performed, thereby allowing users to more easily undo, extend, and modify the operations. Finally, the single, compact underlying mathematical formulation of the tools enables specialized, dynamically-configurable cutaways views such as ribbons.

2 Related Work

A wide range of novel visualization techniques have been proposed for managing 3D scene occlusion [1], for both volume data and surface mesh data (which is the focus here). The majority of these techniques fall into one (or more) of the following categories: transparency, cutaway, explosion and deformation. Aperio supports cutaway, transparency and explosions, as well as model rearrangement via rigid transformations. In this section we review related work in the cutaway, explosion and deformation categories only. We also briefly compare and contrast these works with Aperio to elucidate the contributions.

Cutaway views have a long history and are a widely used technique for revealing hidden objects [2–7]. In Knodel et al. [2] users generate cutaways using simple sketching actions and then refine the shape of the cut using widgets. Li et al. [3] present a system for experts/artists to interactively author cutaway illustrations of complex 3D models. Users can then use a pre-authored viewer application for exploring the data. To minimize the loss of contextual information, McInerney and Crawford [4] remove only part of the occluding geometry and retain polygonal strips called "ribbons" or solid thick "slices". Pindat et al. [5] use a moveable lens that combines a cut-away technique with multiple detail-in-context views of the data. Trapp and Döllner [6] generalize *clip-planes* to *clip-surfaces* that support the generation of curving cut surfaces with user definable contours. Burns and Finkelstein [7] uses a depth texture along with a depth parameter to generate view dependent cutaways. The cutter tool in Aperio is most similar to Pindat et al. [5], and Li et al. [3]. Pindat et al. [5] use a cone shaped cutter "lens" whereas Aperio uses a more flexible superellipsoid. Unlike the pre-authored system in Li et al. [3], which restricts the user's control over view generation, Aperio's real-time previewing cutter "lens" tool is controlled by the end user. However, Li et al. are currently able to generate a wider shape range of cutaways. Aperio also supports "ribbons" but unlike the statically generated ribbons of McInerney and Crawford [4], they are generated in real-time and are interactively configurable.

Exploded views attempt to reveal hidden surfaces or spatial relationships by interactively spreading objects apart along a path [8–11]. Radial explosion paths are common [8] but this simple strategy does not provide the user with much control and may result in visual clutter. Tatzgern et al. [9] introduced an automatic approach where only subsets of model part assemblies are exploded in an attempt to reduce visual clutter. Li et al. [10] generate exploded views by automatically creating an explosion graph that encodes how parts are moved. Many of the above techniques may be better suited to machine part models than the curving, twisting and closely adjoining or intertwined object mesh models found in anatomy/medical data. That is, the simple radial paths or the pre-computed/automatically generated explosion graphs/paths may make assumptions about the shape and spatial arrangement of the models. For example, Li et al. [10] assume that parts can be separated via linear translations, that models are two-sided and that parts fit together without interference. These assumptions may not apply to many human anatomy structures and subsystems. Furthermore, pre-computed/automated explosion graphs/paths restrict the

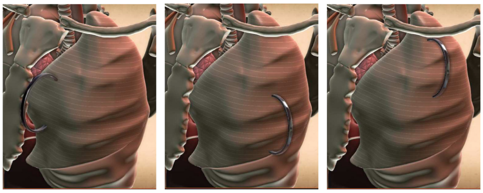

Fig. 2. Ring tool sliding along a model surface. The ring partially penetrates the model surface and automatically aligns itself with the model surface normal vector.

ability to explore individual/group relationships in model subsystems. Aperio was expressly designed to be less automatic and more user-controllable and flexible, so as to better handle systems of curving/twisting models. User-configurable path orientation, curving paths, and model sliding control (with/without "explosion") of individual models or model groups can be dynamically created with rods/rings.

Unlike exploded views, deformation techniques employ nonrigid transformations [12–14], such as peeling, bending, and retracting objects. Correa et al. [13] coin the term "Illustrative Deformation" to describe an approach to volume and surface data visualization that uses 3D displacement maps to perform a wide range of deformations. Birkeland and Viola [12] present a view-dependent peel-away technique for volume data. McGuffin et al. [14] propose an interactive system for browsing pre-labeled iso-surfaces in volume data where users can perform deformations to cut into and open up, spread apart, or peel away parts of the volume in real time. When dealing with anatomy mesh data, the complex spatial interrelationships of curving/twisting models may complicate deformation operations and result in unrealistic model interpenetration. Geometric-based deformation techniques may result in model transformations that are physically realistic (such as peeling) but typically do not consider real tissue deformation properties and may therefore generate un-intuitive and unexpected deformations of some structures if used improperly. Furthermore, for multiple adjoining subsystems, such as arteries, muscles and bones, it may be difficult to quickly and simply apply geometric-based deformations. For these reasons, we decided to support constrained *rigid* transformations combined with cutaways. Rigid transformations are efficient, familiar and easily understandable, enable simple, precise controllability, are flexible and easily combined, and result in predictable *geometric* behavior that is not restricted by expectations of physical realism.

3 Aperio Tool Interaction

Currently Aperio uses a mouse and modifier keys to position tools. A tool is instantly instantiated and made active by clicking on a tool icon from a control panel. The user can then smoothly slide the active tool along the surface of a model or (optionally) smoothly slide it (partially) off of the model surface by switching to a mode that uses a pre-computed oriented bounding box (OBB) of

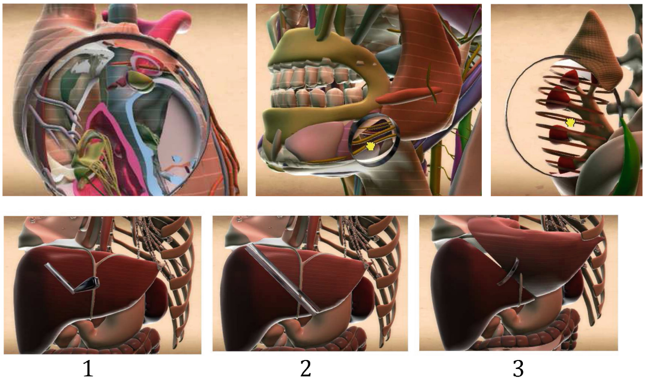

Fig. 3. Top row: cutter tool cutting into several objects in real-time: (1) the heart, (2) head with facial nerve is exposed, (3) kidney cutaway using "ribbons". Bottom row: knife and spreader tools are used cut a liver model into two pieces. A ring tool has been added to "open up" the cut liver (Color figure online).

the model. The tool will automatically orient itself to match the normal vector of the current model surface (Fig. 2) or OBB. The active tool can also be optionally "planted" at any time to establish a view, and "picked up" at a later time to modify the view. We use modifier keys and the mouse wheel to quickly and fluidly change the active tool size, orientation and depth without disturbing its current position. For tool parameters changed less often, GUI sliders are provided.

3.1 Cutting Tools

The ("cookie") cutter tool (Fig. 3) can interactively slide along (multiple) selected model surfaces and cutaway the parts of models inside the cutter boundaries, to a user-defined depth, in real-time. The selected model's cut-surface is automatically "capped" by the renderer so that the object model always appears solid. Furthermore, the user can double click as they move the mouse, selecting deeper objects visible inside the cutter. These objects are instantly cut away providing the ability to dynamically "drill-down" into the data and reveal inner layers. A cutter can optionally cut away a pattern of the occluding model, forming surface "ribbons" [4] contained within the superellipsoid cutter boundaries (Fig. 3 top row, right). The user can dynamically control ribbon orientation by spinning the cutter with the mouse wheel, and properties such as ribbon width, frequency and "tilt" can also be dynamically modified using GUI sliders.

The knife tool (Fig. 3 bottom row) is analogous to a surgical scalpel and is designed to automatically cut an object model into two separate pieces. The knife is rendered as a tapered, flattened superquadratic cylinder that resembles a knife blade. The user simply draws the knife across a selected object surface[2] and a shallow, narrow "surgical incision" cutaway region is rendered in real-time

[2] It is not necessary to draw completely across the selected object.

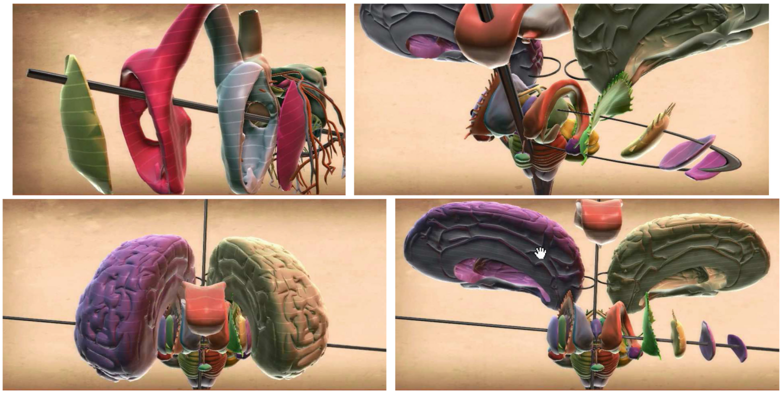

Fig. 4. Top left: rod tool used to create exploded view of heart model parts. Top right: ring tool used for "opening" models like pages in a ringed notebook or exploding models along a curving path. Bottom: two rings and two rods are used to open up and explode several brain parts.

to provide visual reinforcement. Once the cut action is finished, an automatically generated narrow rectangular cutter, known as a "spreader" tool, is instantiated and rendered (Fig. 3 bottom row center) inside the incision. A cutter scaling algorithm is executed based on the model's OBB, as is a highly optimized computational solid geometry (CSG) difference operation using the Carve library that typically completes in under a second. The result is the model is divided into two pieces, with a slight separation between them, along the "incision" path. The spreader tool can be used to interactively widen the incision cut to any desired degree. The user can treat the two pieces of the model in the same manner as any other object model. For example, rods and rings may be used to translate/rotate each piece and each piece can be divided with knife cuts.

3.2 Rearrangement Tools

The rod tool is a user-extendable cylinder-shaped superquadric (Fig. 4). Upon selection, it is oriented along the model surface normal at the current mouse point and partially penetrates the model surface. The simple cylindrical shape, pose, metallic appearance and partial model surface penetration suggests to the user that a model can "slide" along the rod. The user positions a rod along a model surface until it penetrates all target selected models[3]. GUI sliders allow the user to control model sliding (and, optionally, model fanning) back and forth along the rod, and to restore the models to their original position. Models can be individually translated along the rod or translated in groups. Furthermore, an additional "Spread" GUI button switches to an exploded view. In this mode,

[3] If a selected model is not penetrated, a point on the rod closest to the center of the model's OBB is used.

the back and forth movement of the GUI slider will cause the penetrated objects to automatically slide apart and together relative to each other.

The ring tool is represented using a supertoroid and is primarily suggestive of rotation and secondarily of translation (Fig. 4 upper right). Like the rod tool, each model that is selected by the user and penetrated by the ring can be individually, or in combination, slid back and forth along the ring with a GUI slider, and each model is automatically aligned with the normal vector of the ring at the current ring-model intersection point. Ring shape is controlled using a separate GUI slider. If the ring is set to a circular shape, models slide and rotate in a manner similar to turning pages on a ringed notebook. Similar to the rod, the "Spread" button can be used to "explode" models along the ring, "opening" them up with respect to each other. If a selected model does not intersect the ring, we use the center of the model's OBB and compute the closest point on the ring from this center point.

In the bottom row of Fig. 4 we show two stages of iterative model rearrangement (i.e. sliding apart and back together) when exploring a multi-part brain model. Two rings and two rods are used to open up the brain. If the user selects all penetrated models then a single GUI slider will slide all models back and forth along their respective rods/rings. In this example, establishing the initial exploded view takes approximately 30 s for an experienced user.

4 Implementation

This section will provide a brief, high-level description of Aperio's implementation. A detailed description can be found in [15]. In addition, Aperio software is open source and is available at github.com/eternallite/Aperio. Aperio is written in C++ and is constructed using the Visualization Toolkit (VTK) [16]. It uses OpenGL and GLSL shaders for rendering and Carve CSG, a constructive solid geometry library for performing cutting and splitting mesh models. Aperio uses superquadrics [17] to represent all tools - compactly defined geometric shapes that resemble ellipsoids and toroids but with a more expressive shape range. VTK contains classes for creating, transforming, and rendering superquadrics. A superquadric [15] has an implicit function formulation that provides a simple test to determine if a point is inside, outside, or on its surface, as well as a corresponding parametric function formulation. The vectors $(\mathbf{right}, \mathbf{up}, \mathbf{forward})$ form the basis of the superellipsoid coordinate system (CS) (Fig. 5) and we convert points between this local CS and world coordinates using a transformation matrix constructed from these vectors.

Cutaway and Capping Algorithm. We use VTK's multi-pass rendering pipeline for "capping" the cut surface of the "hollow" mesh models to create the illusion of solid models. Our cut-surface capping algorithm requires two render passes: a pre-pass to render information into texture images of a Frame Buffer Object (FBO) and a subsequent main-pass that reads in texture data and renders the final scene. Each of these render passes triggers the execution of a special

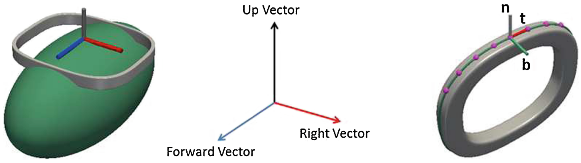

Fig. 5. Left, middle: A local superquadric coordinate system is constructed from the camera's "view up" vector and the data surface normal at the current mouse position. Right: A coordinate system is constructed at each sampled point along the middle of the ring's outer surface and is used to orient a model during sliding.

GPU fragment shader. In both passes, both fragment shaders first discard all selected mesh fragments that are inside the boundaries of the superellipsoid cutter, making use of the convenient superellipsoid implicit inside-outside function. The removal of these fragments will expose back-facing fragments of a cut model. We must replace these fragments with back-facing cutter fragments to achieve a solid cut (Fig. 3 upper left). In the pre-pass, the fragment shader outputs the depths and colors of all selected model front and back facing fragments, as well as the depth of all back facing cutter fragments into an FBO. Depths are encoded using an RGBA color vector. The pre-pass fragment shader also sets the color of all discarded/non-filled fragments in the texture to the RGBA color $(1,1,1,1)$ (i.e. "infinite" depth or pure white) so that they are distinguishable from actual depths. In the main-pass we read in the FBO textures generated by the pre-pass and check if a fragment was discarded/non-filled. If so, it likely requires capping. For every back-facing fragment of the cutter, the main-pass fragment shader tests if the front-facing fragment of a selected model is discarded/non-filled, and the depth of the back-facing cutter fragment is less than (i.e. closer) the depth of the corresponding back-facing selected model fragment. If true then a selected model fragment color is output but using the back-facing cutter fragment depth. This algorithm will only render cap fragments that are within the bounds of a selected model.

Ring and Rod Path Generation. The parametric surface representation of a superquadric provides a basis for easily definable, unambiguously oriented sliding paths. To implement sliding along a ring, we use VTK's superquadric class to generate a position and normal vector **n** for points sampled along a path of the supertoroid that symmetrically divides the supertoroid in half (Fig. 5 right). We can then determine what orientation to give the models (at each point along the path) as they slide on the ring by constructing a local CS at each path point. The tangent vector **t** of a path-point CS is calculated by subtracting the current point from the next path point. The bi-tangent vector **b** is simply the cross-product of the normal and tangent vectors. The normal, tangent and bi-tangent vectors are used to construct a rotation matrix which is then multiplied by all mesh model points to orient it at the current path point. To determine the initial path point for a selected model, we use VTK to determine the intersection point between the ring and the model. If a selected model does not intersect the ring, we use the center of the model's OBB and compute the closest point on the ring from this center point. For the rod tool, the path generation is greatly simplified as the rod central axis is a line segment. Path points are generated evenly along this line segment and the rod central axis is used as the rotation axis to spin models. We again calculate the intersection point between the rod central axis and a selected mesh for the initial path point.

5 Validation

We performed a qualitative user study to gather some supporting evidence of the effectiveness of the visual cues provided by the tools, and the intuitiveness and controllability of user interactions. Eighteen participants each had a one hour session where they were asked to perform visualization tasks using Aperio. The 18 participants were aged 19–34 with an average age of roughly 26 years; there were 14 males and 4 females. The user study consisted of two trials and one practice trial. In each trial participants were shown several scene images depicting a subsystem of the anatomy data set in various "stages" of model rearrangement. Participants were asked to use Aperio to match each stage[4]. The matching task also included restoring the position and orientation of models to their original state. Participants were informed that their performance was not being measured. To begin the study, each participant was shown the functionality and interface of Aperio and were then allowed to play with the system using a demonstration data set. They were then asked to perform a practice trial followed by the matching task in two subsequent trials, with all trials using different anatomical model subsystems. Finally, once the scene matching tasks were completed, participants were asked to play with the cutter tool and ribbon view cutter on one of the trial data sets.

After the trials the participants filled out a questionnaire and indicated their level of agreement or disagreement (using a 7-point Likert scale) with statements pertaining to the perception of Aperio tools (Fig. 6 left) and to Aperio's user interface (Fig. 6 right). As is evident from the bar graphs (Fig. 6), the results were very positive in this qualitative study. In addition, with respect to the cutter tool versus the ribbon cutter, 15 users thought the ribbon cuts were an effective alternative to a full cutaway (for preserving the shape of the cut object), 2 did not think it was effective and 1 chose "OK". With respect to Aperio's user interface, of the 18 participants, 14 found the Aperio's tool interface intuitive, 4 did not. For tool preference, 9 users preferred the rod tool, 8 users preferred the ring tool and 1 user did not like either.

Finally, we performed two additional online perception studies using Survey Monkey [18]. In each study, 50 people participated and the two groups did

[4] We also initially used a separate system with a simplified user interface, similar to the interface of common object modeling packages, as a control for the study but it was found to be too simplistic and was rated poorly by almost all users. The results for this simplified system are not included in this paper but can be found in [15].

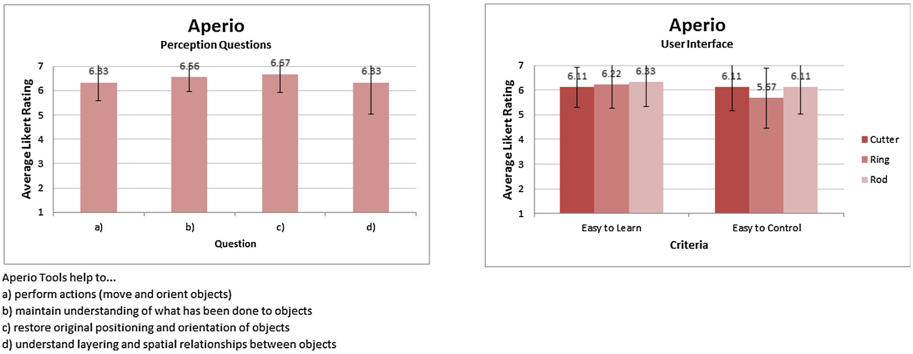

Fig. 6. Left: results of user study related to perceiving model spatial relationships using Aperio tools. Right: results of user study related to Aperio's user interface.

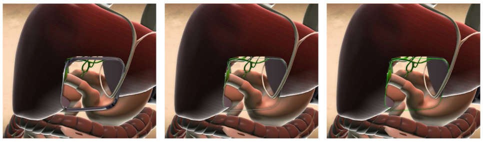

Fig. 7. Example of images used in the online user study using a visual cue of (a) metal rings, (b) gray curve, (c) bright green curve (Color figure online).

not overlap. The ages of the participants ranged from 18 to 60+ and all had at least four years of college level education. In these studies the goal was to determine if visual cues (such as Aperio tools) were useful for understanding the spatial interrelationships of models after the tools had been applied, as well as for understanding how the model parts had been moved apart or cut away. We created a series of static images of various model subsystems (e.g. the parts of the brain) with our tools to move apart or cut models (Fig. 7). We asked several multiple choice questions related to the understanding of the model rearrangements. In the first study, we presented multiple sets of four images to the participants. The first image showed a pre-transformed view and three images showed the result of the tool action with rings/rods/cutters as visual aids, with simple gray lines/curves as visual aids with each having the same shape and dimension as a corresponding rod/ring/cutter, and with no visual aids. We chose simple line/curve visual cues as a comparison to determine if the realistic shiny metal surface renderings of the Aperio tools were advantageous in these visual analysis scenarios. The second study was identical to the first except the gray lines/curves were replaced with bright green lines/curves that visually stood out more strongly against the data models.

In both studies the results showed that for rod/ring tool operations, on average, the participants preferred some visual aid (56.18 %) to no visual aid (28.64 %), while some had no preference (15.18 %). For cutter tool operations, on average the participants preferred some visual aid to help them determine what model region had been cut (35.85 %) to no visual aid (21.36 %). However, the results also showed that, on average, while more participants preferred the metal tools (34.88 %) to gray lines/curves (18.42 %), more participants preferred the bright green lines/curves (43.85 %) to the metal tools (24.42 %). We conjecture that when performing a visual analysis on images of 3D models that have already been transformed, as long as the visual cues are clearly visible and their shape/curvature discernable, then the least distracting visual cue (i.e. with the smallest visual impact with respect to the data models) is preferable.

6 Conclusion

User interaction models are an integral part of effective interactive visual exploration of 3D data. If the interaction model is too complex, for example one based on a widget with many handles and controls and artificial appearance, the user may spend too much time learning and/or applying the model. On the other hand, if the interaction model is too simple, then tool/option proliferation may result. This complicates the interface, reduces interaction consistency and forces the user to perform too much work. Furthermore, automated techniques or pre-computed views may overly restrict the user's control over view generation. Aperio lets the user decide which data models to rearrange and how to rearrange them. This approach may result in a heavier cognitive load with respect to which tools to use and how to apply and combine them to obtain the desired view. However, by adhering to this simpler approach of highly controllable model rearrangements and cutaways, users may discover/view relationships between complex organic-shaped multi-part models in ways not predicted or not possible by more automatic techniques. To help offset the increased cognitive and interaction load on the user, Aperio uses a small set of easily-applied tools with familiar shapes, appearance and affordance, to guide and visually reinforce user actions. For expert users who may want to create more complex views such as free-form shaped cutaways or custom curving explosion paths, it may be desirable to allow tool constraints to be "loosened". For example, superquads can be interactively deformed using global deformation functions, such as bending and tapering [17], expanding their range of possible tools, tool paths, and cutaway shapes. In addition, multiple superquads can be blended to form complex shapes while still providing a well-defined inside-outside function. These features are the subject of future research.

References

1. Elmqvist, N., Tsigas, P.: A taxonomy of 3d occlusion management for visualization. IEEE Trans. Visual Comput. Graphics **14**, 1095–1109 (2008)
2. Knödel, S., Hachet, M., Guitton, P.: Interactive generation and modification of cutaway illustrations for polygonal models. In: Butz, A., Fisher, B., Christie, M., Krüger, A., Olivier, P., Therón, R. (eds.) SG 2009. LNCS, vol. 5531, pp. 140–151. Springer, Heidelberg (2009)

3. Li, W., Ritter, L., Agrawala, M., Curless, B., Salesin, D.: Interactive cutaway illustrations of complex 3d models. ACM Trans. Graph. **26**, 31:1–31:11 (2007)
4. McInerney, T., Crawford, P.: RibbonView: interactive context-preserving cutaways of anatomical surface meshes. In: Bebis, G., et al. (eds.) ISVC 2010, Part II. LNCS, vol. 6454, pp. 533–544. Springer, Heidelberg (2010)
5. Pindat, C., Pietriga, E., Chapuis, O., Puech, C.: Drilling into complex 3d models with gimlenses. In: VRST 2013, pp. 223–230 (2013)
6. Trapp, M., Döllner, J.: 2.5d clip-surfaces for technical visualization. J. WSCG **21**, 89–96 (2013)
7. Burns, M., Finkelstein, A.: Adaptive cutaways for comprehensible rendering of polygonal scenes. ACM Trans. Graph. **27**, 154:1–154:7 (2008)
8. Sonnet, H., Carpendale, S., Strothotte, T.: Integrating expanding annotations with a 3d explosion probe. In: Proceedings of the Working Conference on Advanced Visual Interfaces, AVI 2004, pp. 63–70. ACM, New York (2004)
9. Tatzgern, M., Kalkofen, D., Schmalstieg, D.: Compact explosion diagrams. In: Proceedings of the 8th International Symposium on Non-photorealistic Animation and Rendering, NPAR 2010. ACM, New York, pp. 17–26 (2010)
10. Li, W., Agrawala, M., Curless, B., Salesin, D.: Automated generation of interactive 3d exploded view diagrams. ACM Trans. Graph. **27**, 101:1–101:7 (2008)
11. Bruckner, S., Gröller, M.E.: Exploded views for volume data. IEEE Trans. Visual Comput. Graphics **12**, 1077–1084 (2006)
12. Birkeland, S., Viola, I.: View-dependent peel-away visualization for volumetric data. In: Hauser, H., Spencer, S.N. (eds.) Spring Conference on Computer Graphics SCCG, pp. 121–128. ACM (2009)
13. Correa, C.D., Silver, D., Chen, M.: Feature aligned volume manipulation for illustration and visualization. IEEE Trans. Vis. Comput. Graph. **12**, 1069–1076 (2006)
14. McGuffin, M.J., Tancau, L., Balakrishnan, R.: Using deformations for browsing volumetric data. In: Proceedings of the 14th IEEE Visualization 2003 (VIS 2003), p. 53. IEEE Computer Society, Washington, D.C. (2003)
15. Tran, D.: Aperio: Managing 3d scene occlusion using a mechanical analogy for visualizing multi-part mesh data. Master's thesis, Dept. of Computer Science, Ryerson University, Toronto, ON, Canada (2015)
16. Schroeder, W.J., Martin, K., Lorensen, W.: The Visualization Toolkit: An Object-Oriented Approach to 3d Graphics, 4th edn. Kitware Inc., Clifton (2006)
17. Barr, A.H.: Superquadrics and angle-preserving transformations. IEEE Comput. Graph. Appl. **1**, 11–23 (1981)
18. SurveyMonkey Inc.: SurveyMonkey. www.surveymonkey.com. Accessed 08 October 2015