Eurographics Conference on Visualization (EuroVis) (2013)
M. Hlawitschka and T. Weinkauf (Editors)



# 3D Strokes on Visible Structures in Direct Volume Rendering

Alexander Wiebel[1,4], Philipp Preis[2], Frans M. Vos[3], and Hans-Christian Hege[1]

[1]Zuse Institute Berlin (ZIB), Germany, [2]FU Berlin, Germany, [3]TU Delft, Netherlands, [4]Coburg University of Applied Sciences, Germany

**Abstract**

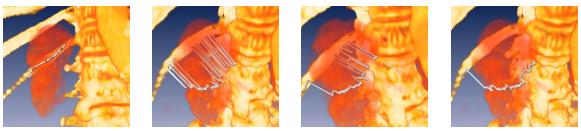
*In this paper we describe VisiTrace, a novel technique to draw 3D lines in direct volume rendered images. It allows to draw strokes in the 2D space of the screen to produce 3D lines that run on top or in the center of structures visible in the rendering. It is able to ignore structures that shortly occlude the structure that has been visible at the start of the stroke. For this purpose a shortest path algorithm finding the optimal curve in a specially designed graph is employed. We demonstrate the usefulness of the technique by applying it to image data from medicine and engineering, and show how it can be used to mark structures in the example data, and to automatically obtain good views toward these structures enabling faster navigation in the rendering.*

Categories and Subject Descriptors (according to ACM CCS): I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques

## 1. Background

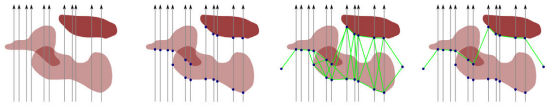
In the 2D setting of image editing, obtaining strokes and points from user interactions is straightforward. When dealing with direct renderings of volumetric data (DVR) the transformation is complicated because of the additional third dimension. If a user draws a stroke in the 2D screen space it is usually not obvious at which depth this line should run through the volume. This problem is further complicated by the application of a transfer function (TF): should the depth of the line depend on the original data or the visibility induced by the TF? For single points, solutions for both variants, i.e. using the original data resp. using the visible rendering, have been conceived recently: [KBKG09] resp. [WVFH12]. For strokes, the previous work has mainly focused on considering the original data, e.g. [ONI05], [DR12]. Both techniques, i.e. picking *visible* structures and selecting *lines*, are well justified and good implementations exist. Yet there is no literature describing a combination of both techniques, i.e. a technique for tracing *lines* lying in or on top of 3D structures *visible* in a DVR. The present paper aims at filling this gap by introducing such a technique, called VisiTrace, an extension, and an exemplary application of the technique to automatic view selection. This combination enables faster and more intuitive navigation in DVR.

As our approach is designed to select 3D structures from 2D screen locations it is slightly related to volume picking (see [WVFH12], [AA09] and references therein). However, most volume picking techniques select only single points whereas VisiTrace aims at line structures. Owada et al. [ONI05] present a volume segmentation technique that uses 2D strokes on DVR images to generate 3D constraints for the segmentation. Another volume data segmentation technique using brushes in 2D has been introduced by Wan et al. [WOCH12]. Chen et al. [CSS08] describe interactive volume sculpting using stroke input for DVR exhibiting completely opaque surfaces. The Volume Cutout technique by Yuan et al. [YZNC05] also segments 3D structures using 2D sketches, but the sketches remain completely 2D in their approach. Yu et al. [YEII12] introduce a technique selecting regions in 3D point clouds using sketched lassos. Kohlmann et al. [KBKG09] use their contextual 3D picking technique also to allow the user to draw lines in or on top of volumetric structures. The main differences of these approaches and our work is that their resulting 3D curves are either not tailored to lie on the most visible structures or that the approaches are only applicable to specific types of data. In the context of vessel segmentation and DVR, Diepenbrock and Ropin-



(a) (b) (c) (d)

**Figure 1:** *DVR of CT data set. (a) Line, as drawn onto the DVR image in two dimensions. (b)-(d) Other perspective. (b) 3D line using WYSIWYP directly. (c) 3D line using WYSIWYP with median filter. (d) 3D line using VisiTrace.*



(a) Step 1 (b) Step 2 (c) Steps 3&4 (d) Steps 5&6

**Figure 2:** *The steps of the VisiTrace algorithm. See Sec. 2 for a detailed description of the algorithm and its steps.*

© The Eurographics Association 2013.

DOI: 10.2312/PE.EuroVisShort.EuroVisShort2013.091-095

delivered by EUROGRAPHICS DIGITAL LIBRARY www.eg.org diglib.eg.org

ski [DR12] described a technique to trace a centerline in a rendering of a vessel by simply drawing a line on top of the vessel. They use a path with lowest cost together with an active contour approach to detect the correct intended 3D line of interest. The description of the weights (or costs) used for the lowest cost path, however, remains vague.

## 2. VisiTrace

To obtain the desired 3D curve, one could be tempted to simply apply visibility oriented picking [WVFH12] for each of the positions along the stroke drawn by the user. However, this can result in very jagged lines (see Fig. 1(b)) because different structures touched by neighboring pixels might exhibit a very similar visibility. In such a case, the picked position will repeatedly jump between the two (or more) similarly visible structures. Application of a median filter to the 3D curve does not resolve this problem (see Fig. 1(c)). To avoid such jumping VisiTrace considers more than one visible location for each pixel and tries to find the most plausible path along these locations. VisiTrace consists of six main steps which will be described in the following.

**Drawing (Step 1)** In this first step the only user interaction, namely the user drawing on top of the DVR image with the mouse (Fig. 1(a)), is performed and recorded. First, the stroke generated by the user is recorded as array of pixel coordinates returned by mouse events. The line strip obtained by connecting the pixels is then resampled to provide a continuous path of pixels. The resulting 2D samples are transformed to 3D world coordinates $P^i, i \in \{0,\dots,N-1\}$ and stored for further processing. For each $P^i$ a viewing ray $R^i$ is traced through the volume (Fig. 2(a)).

**Detecting Visible Structures (Step 2)** With the viewing rays $R_i$ at hand we extract the most visible volumetric structures along each ray. For this purpose we employ a modified version of the visibility oriented picking (WYSIWYP) described by Wiebel et al. [WVFH12]. While WYSIWYP extracts the single most visible structure along the ray, we aim at having a relatively smooth 3D curve and thus will consider also less visible structures. Hence, the locations $v^i_k, k \in \{1,\dots,J\}$ of the $J$ highest jumps of accumulated opacity along each $R^i$ are stored (dots in Fig. 2(b)). The locations are candidates for being members of the final 3D curve.

**Graph Generation (Step 3)** Steps 3 to 6 try to find the best 3D line that runs through exactly one location $v^i_k$ of each ray and contains as many "most visible" locations as possible without creating many spikes due to negligible visibility changes in the volume rendering. The optimization is carried out by a shortest path algorithm on a specially defined graph (Fig. 2(c)). The graph consist of the $v^i_k$ as nodes and directed, weighted edges $e^i_{j,k}$ connecting nodes $v^i_j$ to $v^{i+1}_k$ (see Fig. 2(c)). A path along the directed edges will represent the final 3D curve. To provide an equally probable origin and target for all paths we add artificial start and end nodes.

**Edge Weights (Step 4)** A clever choice of the weights assigned to the edges is crucial for the result of the shortest path algorithm applied to the graph. Our choice, a function based on location and opacity, and the reasoning for its design are described below. The reasoning is based on a number of requirements (R.1-R.7) for the behavior of the final 3D curve. We derive these requirements from the abstract yet basic application scenarios show in Figs. 3(a)-(d).

**R.1** *If two path variants differ in magnitude of their opacity jumps, but are equivalent otherwise (e.g. edge lengths), the one with the higher jumps should be chosen. It is more visible and thus probably intended by the user. (Fig. 3(a))*

We do not prefer structure changes depending on their position along the curve (start/end/in between).

**R.2** *If the opacity jumps of path variants are similar, the variants should be equally likely.*

**R.3** *Avoidable structure changes (with possibility to stay on same structure) should only be performed if the opacity jump of the continued structure is much smaller than the one of the newly appearing structure. (Fig. 3(b))*

We assume that users wants to select continuous structures.

**R.4** *Changing the structure should be less likely than staying on it (when opacity values are equal). For this reason, the spatial position of the nodes must be considered.*

Thus, two structures having the same opacity jumps can be weighted differently in order to achieve a *smoother* path.

**R.5** *The weight of straight structures should be very low to avoid making costs of a structure change less significant.*

**R.6** *Unambiguous structure changes should not be weighted by the depth difference of the structures. (Fig. 3(c))*

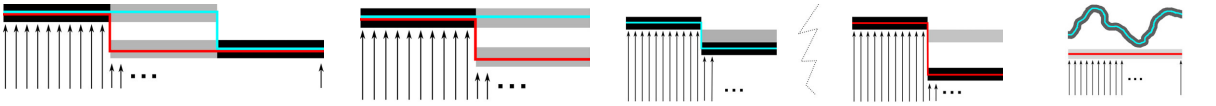
It would be desirable that the selected structure in Fig. 3(c) only depends on the opacity jumps because it unambiguously specifies what is visible here.

**R.7** *The weight of slightly bent structures should not be much higher than that of a straight structure because a followed structure might be oblique in space and might thus cause a longer path without a structure change. (Fig. 3(d))*

**Opacity Weighting** Above, we mentioned that it is desirable to let the magnitude of opacity jumps for candidate locations have an influence on the final weight of an edge (R.1-R.3). For an increasing opacity jump magnitude, the goal is an increased probability for an edge to be in the shortest path and thus a decreasing edge weight. Therefore, and because the magnitude of opacity jumps is in $[0,1]$, we set $w_\alpha(v^i_j, v^{i+1}_k) := 1 - (\text{magnitude of opacity jump } v^{i+1}_k)$. A further weight modification of the weight does not seem necessary because the opacity influence of WYSIWYP working in the same way has proven to be effective.

**Distance Weighting** R.4-R.7 imply that the edge weights should also depend on the relative positions of the graph nodes. We considered different distance-based weights. We chose the Euclidean distance between the nodes, because the resulting weight grows only slowly for small depth variations (R.7). Furthermore, it assigns sufficiently small weights to very small distances. As the rays are very close

© The Eurographics Association 2013.



(a) Unavoidable change (b) Avoidable change (c) Two cases with unambiguous change (d) Curved structure

**Figure 3:** *Basic scenarios of structure selection for partly occluding semi-transparent structures. Color of structures indicates magnitude of corresponding opacity jump (**high**, low). Blue/red lines are possible variants for resulting curve. Details in text.*

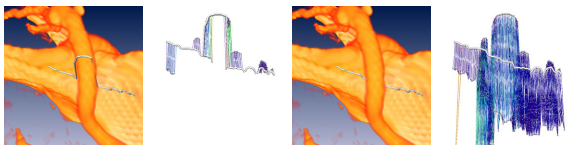
due to the resampling, the fact that the Euclidean distance also depends on the distance between rays does not violate R.5. We obtain the weights $w_d$ from the Euclidean distances by normalizing all distances to lie in $[0,1]$. The weight function $w_d$ has been devised in an effort to fulfill most requirements related to distance. However, in favor of an intuitive and reliable weight function we decided to violate R.6.

**Combined Weight** The partial weights incorporating spatial distance and opacity jumps have been described above. They can be combined to form the final weight function $w(v_j^i, v_k^{i+1})$. In summary, the combined function should have the following properties: 1) $w$ must strictly increase with $w_\alpha$, and $w_\alpha \approx 0 \Rightarrow w \approx 0$, so that opaque structures are chosen with high probability. 2) $w$ must be monotonically increasing with $w_d$, and $w_d \approx 0 \Rightarrow w \approx 0$, so that close jump locations are chosen with very high probability. A straight forward choice for the combined function which fulfills the criteria would be $w_1 = w_\alpha \cdot w_d$. This function works well in many cases but leaves room for improvements regarding R.5 and R.7. In order to give very small differences in distance (which might result from an oblique structure) an even lower influence on the final path we use $w_2 = w_\alpha \cdot {w_d}^2$.

**Shortest Path and 3D Curve (Steps 5&6)** In the fifth step Dijkstra's shortest path algorithm [Dij59] is applied to the graph equipped with the edge weights described above. The algorithm searches for a path between the artificial start and end node. Because the graph contains $\mathcal{O}(N)$ edges, a time complexity of $\mathcal{O}(N \log N)$ can be achieved [FT84]. The shortest path obtained in this step is easily translated into the final 3D curve in step six. The artificial start and end nodes are removed from the obtained path.

## 3. VisiTrace Extended

The algorithm as described up to this point achieves the goal of tracing lines on top of visible structures. In some cases, however, the user might be interested in tracing a line that stays on the object where it started as long as possible (Fig. 4(c)) and ignore crossing objects. This is not always possible when applying VisiTrace. An example is a crossing object which is completely opaque. Only one candidate will be found along the ray in this case (see Figs. 4(a), 4(b)). The desired result, shown in Figs. 4(c), 4(d), can be achieved by using VisiTrace with down-scaled opacities. Down-scaling, i.e. a division of the opacity of all samples by the maximum number of samples possible per ray (depending on the



(a) (b) (c) (d)

**Figure 4:** *Selected curves (a),(c), and corresponding graph representations (b),(d). Edge weights are color coded. (a),(b) without opacity scaling. (c),(d) with opacity scaling.*

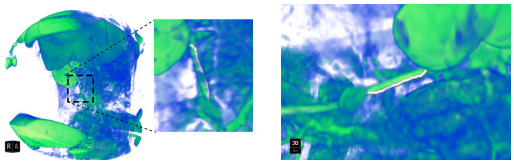


**Figure 5:** *Left: Original perspective when stroke was drawn. Right: Corresponding reoriented view providing best view.*

sampling distance of the DVR), makes all structures transparent to some degree, leading to candidate points also behind originally opaque structures. In other words, the scaling allows the picking algorithm to look through the originally completely opaque crossing structure (see Fig. 4(d)). The resulting opacities do not correspond to the actually perceived opacities anymore, but lead to the desired 3D curve which intentionally does not run only on the visible structures. Fig. 4 also shows the generated graphs for VisiTrace with and without opacity scaling. It is obvious that the relatively smooth line in Fig. 4(c) is not achievable using the graph in Fig. 4(b).

We expect the user to point intentionally on visible structure first. Thus we deactivate down-scaling for the first position to get the actually visible depth there. In practice, this ensures that we obtain the desired 3D curve. The usual result of VisiTrace is a 3D line running on top of structures of the DVR. If the center of the jump intervals is used, VisiTrace allows to trace lines *in* visible volumetric structures. This can be especially interesting for tubular structures.

## 4. Automatic Viewpoint Selection Using VisiTrace

The 3D curve resulting from VisiTrace can be used in various ways. In general, its use only depends on the application context of the DVR it is applied to. Here, we describe how it can be used for automatic view point selection. An automatic selection of good view points on specific structures in

© The Eurographics Association 2013.

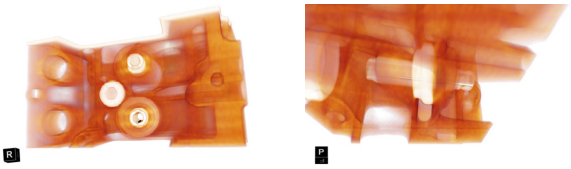


**Figure 6:** *Original perspective while drawing (left) and reorientation based on opacity and direction weighting.*

DVRs can aid fast navigation and data exploration. A number of different techniques to determine good views has been presented in the past (see [ZAM11] and references therein, e.g. [KBKG07], [BS05], [VMN08], [KUBS12], [TFTN05]). In contrast to previous methods, we propose to use the 3D curve obtained by VisiTrace to mark the desired features in the DVR and then compute a viewpoint providing the best view toward the curve and thus the feature. The best view has two balanced properties: the curve should be occluded by the DVR as little as possible, and the structure of the curve should be perceivable as good as possible. In order to determine how strongly the curve is occluded if viewed from possible view points, we perform the following procedure for candidate points $c_i$, $i \in \{0,\ldots,C-1\}$ equally distributed on a sphere (with radius twice the diagonal of the curve's bounding box) surrounding the whole 3D curve. A ray $cR_i$ is cast from the current $c_i$ to every point on the curve. For each ray the opacity is accumulated using the same sampling rate as the DVR. The accumulated opacities for the rays are summed up and represent the visibility of the curve at $c_i$. Choosing the candidate with the lowest sum of accumulated opacities, satisfying results can be achieved in most cases. However, in some cases the direction yielding least occlusion may not be optimal. Consider a relatively straight curve. If the best viewing direction was nearly parallel to this curve then the selected view would not provide much insight because the curve would cover only a small portion of the screen space. In order to avoid this, we weight the summed opacities using the overall shape of the curves. The shape can be characterized by the eigenvalues $e_1 \geq e_2 \geq e_3$ and eigen vectors $\mathbf{e}_1, \mathbf{e}_2, \mathbf{e}_3$ obtained using a PCA of the curve vertices. The relations of the eigenvalues can be used to differentiate between three different shapes: linear, planar (e.g. circular 3D curve on a flat object) or spherical (e.g. jumpy 3D curve obtained from spiral stroke) [WPG*97]. For each of the three shape types a different weighting is used: *Linear:* Weighting by $\|cR_i \cdot \mathbf{e}_1\|$. This disqualifies directions (nearly) parallel to the curve. *Planar:* Weighting by $(1-(cR_i \cdot \mathbf{e}_3))$ (all vectors normalized). This promotes directions pointing toward the "flat" part of the shape. *Spherical:* No weighting. For the final view the camera is located at the best $c_i$ and is looking toward the center of the sphere. As an alternative to using the best viewing angle, one can choose the viewing angle, e.g. out of the best five, lying closest to the initial viewing direction. This maintains a kind of *view coherence* which can be made even more effective by the straight-forward implementation of a *smooth* transition between the viewpoints.

## 5. Results and Discussion

Using the shortest path obtained by VisiTrace, one can achieve plausible 3D curves running on top of relatively transparent volumetric regions. The result is depicted in Fig. 1(d). Instead of repeatedly jumping between the organ in the foreground and the bone in the background as in Figs. 1(b) and (c), the 3D curve stays on top of the organ structure. More examples for this effect are shown in the video accompanying this article. Fig. 4 illustrates how opacity-scaling enables the user to select continuous structures shortly being in the background. Fig. 5 demonstrates the usefulness of the view selection approach. In the left image the front of a visible structure has been selected by a stroke. For the right image the view has been automatically reoriented to provide a view toward the structure and the 3D curve on top of it. The effect of the shape-based weighting is shown in Fig. 6. Although the curve is least occluded when looking *into* the tube-like engine part, our method chooses a different viewing direction (right) to convey the shape of the curve as well as maintaining visibility of the curve itself. In the video accompanying this article, we demonstrate the interactive nature of our approach. VisiTrace provides the final 3D curve almost instantly. With activated opacity scaling, producing more candidate points per ray, long strokes can lead to a comparatively large graph and thus to a longer computation for obtaining the shortest path. Nevertheless, computing the 3D curve is still faster than carefully drawing the original stroke, that is in the order of a few seconds. As standard VisiTrace is focused on *visible* structures, it is intentionally dependent on what is visible, i.e. on the chosen TF and the viewing direction when drawing the 2D stroke.

## 6. Conclusion and Future Work

In this paper, we have introduced a new way of interacting with DVR. The presented approach allows the user to mark visible structures in DVR by translating 2D strokes into 3D curves. The 3D curves are extracted in *soft* real-time and can thus be integrated into an every-day work-flow dealing with DVR. Based on the 3D curves we generate optimized view points allowing for an improved navigation in DVR. Possible directions for future work, additional application scenarios, some additional minor implementation details, as well as a more detailed discussion of the related work can be found in a technical report [WPVH13].

**Acknowledgments** Based on one author's thesis [Pre12]. We use data from volvis.org. Parts of the research leading to these results has received funding from the European Commission's 7th Framework Programme (FP7/2007-2013) of the VIGOR++ Project under grant agreement nr. 270379. The information presented is provided as is and no guarantee or warranty is given that the information is fit for any particular purpose. The user thereof uses the information at its sole risk and liability. The opinions expressed in the document are of the authors only and in no way reflect the EC's opinions.

© The Eurographics Association 2013.

## References


[AA09] ARGELAGUET F., ANDUJAR C.: Efficient 3d pointing selection in cluttered virtual environments. *IEEE Computer Graphics & Applications 29*, 6 (Nov. 2009), 34–43. doi:10.1109/MCG.2009.117. 1

[BS05] BORDOLOI U. D., SHEN H.-W.: View selection for volume rendering. In *IEEE Visualization Conference* (2005), IEEE Computer Society, pp. 487–494. doi:10.1109/VIS.2005.110. 4

[CSS08] CHEN H.-L. J., SAMAVATI F. F., SOUSA M. C.: GPU-based point radiation for interactive volume sculpting and segmentation. *The Visual Computer 24*, 7-9 (2008), 689–698. doi:10.1007/s00371-008-0249-5. 1

[Dij59] DIJKSTRA E.: A note on two problems in connexion with graphs. *Numerische Mathematik 1* (1959), 269–271. doi:10.1007/BF01386390. 3

[DR12] DIEPENBROCK S., ROPINSKI T.: From Imprecise User Input to Precise Vessel Segmentations. In *Eurographics Workshop on Visual Computing for Biology and Medicine* (Norrköping, Sweden, 2012), Eurographics Association, pp. 65–72. doi:10.2312/VCBM/VCBM12/065-072. 1, 2

[FT84] FREDMAN M., TARJAN R.: Fibonacci heaps and their uses in improved network optimization algorithms. *Foundations of Computer Science, IEEE Annual Symposium on 0* (1984), 338–346. doi:10.1109/SFCS.1984.715934. 3

[KBKG07] KOHLMANN P., BRUCKNER S., KANITSAR A., GRÖLLER M. E.: LiveSync: Deformed viewing spheres for knowledge-based navigation. *IEEE Trans. on Visualization and Computer Graphics 13*, 6 (Oct. 2007), 1544–1551. doi:10.1109/TVCG.2007.70576. 4

[KBKG09] KOHLMANN P., BRUCKNER S., KANITSAR A., GRÖLLER M. E.: Contextual picking of volumetric structures. In *Proceedings of the IEEE Pacific Visualization Symposium 2009* (May 2009), Eades P., Ertl T., Shen H.-W., (Eds.), IEEE Computer Society, pp. 185–192. doi:10.1109/PACIFICVIS.2009.4906855. 1

[KUBS12] KIM H. S., UNAT D., BADEN S. B., SCHULZE J. P.: Interactive data-centric viewpoint selection. 829405–829405–12. doi:10.1117/12.907480. 4

[ONI05] OWADA S., NIELSEN F., IGARASHI T.: Volume catcher. In *Proceedings of the 2005 Symposium on Interactive 3D Graphics and Games* (New York, NY, USA, 2005), I3D '05, ACM, pp. 111–116. doi:10.1145/1053427.1053445. 1

[Pre12] PREIS P.: *Sichtbarkeitsorientiertes Picking in Direct Volume Renderings mit Beleuchtungsmodellen*. Master's thesis, Freie Universität Berlin, 2012. 4

[TFTN05] TAKAHASHI S., FUJISHIRO I., TAKESHIMA Y., NISHITA T.: A feature-driven approach to locating optimal viewpoints for volume visualization. In *IEEE Visualization* (2005), IEEE Computer Society, pp. 495–502. doi:10.1109/VISUAL.2005.1532834. 4

[VMN08] VÁZQUEZ P.-P., MONCLÚS E., NAVAZO I.: Representative views and paths for volume models. In *Proceedings of the 9th International Symposium on Smart Graphics* (Berlin, Heidelberg, 2008), SG '08, Springer-Verlag, pp. 106–117. doi:10.1007/978-3-540-85412-8_10. 4

[WOCH12] WAN Y., OTSUNA H., CHIEN C.-B., HANSEN C.: Interactive extraction of neural structures with user-guided morphological diffusion. In *Proceedings of IEEE Symposium on Biological Data Visualization (BioVis'12)* (2012), pp. 1–8. doi:10.1109/BioVis.2012.6378577. 1

[WPG*97] WESTIN C.-F., PELED S., GUDBJARTSSON H., KIKINIS R., JOLESZ F. A.: Geometrical diffusion measures for MRI from tensor basis analysis. In *ISMRM '97* (Vancouver Canada, April 1997), p. 1742. 4

[WPVH13] WIEBEL A., PREIS P., VOS F. M., HEGE H.-C.: *Computation and Application of 3D Strokes on Visible Structures in Direct Volume Rendering*. Tech. Rep. 13-20, Zuse Institute Berlin (ZIB), Takustr. 7, 14195 Berlin, 2013. 4

[WVFH12] WIEBEL A., VOS F. M., FOERSTER D., HEGE H.-C.: WYSIWYP: What you see is what you pick. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (Dec. 2012), 2236–2244. doi:10.1109/TVCG.2012.292. 1, 2

[YEII12] YU L., EFSTATHIOU K., ISENBERG P., ISENBERG T.: Efficient structure-aware selection techniques for 3D point cloud visualizations with 2DOF input. *IEEE Transactions on Visualization and Computer Graphics 18*, 12 (Dec. 2012), 2245–2254. doi:10.1109/TVCG.2012.217. 1

[YZNC05] YUAN X., ZHANG N., NGUYEN M. X., CHEN B.: Volume cutout. *The Visual Computer (Special Issue of Pacific Graphics 2005) 21*, 8-10 (2005), 745–754. doi:10.1007/s00371-005-0330-2. 1

[ZAM11] ZHENG Z., AHMED N., MUELLER K.: iView: A feature clustering framework for suggesting informative views in volume visualization. *IEEE Transactions on Visualization and Computer Graphics 17* (2011), 1959–1968. doi:10.1109/TVCG.2011.218. 4


© The Eurographics Association 2013.