DOI: 10.1111/cgf.12621
Eurographics Conference on Visualization (EuroVis) 2015
H. Carr, K.-L. Ma, and G. Santucci
(Guest Editors)

Volume 34 (2015), Number 3

Guided Volume Editing based on Histogram Dissimilarity

A. Karimov[1], G. Mistelbauer[1], T. Auzinger[1] and S. Bruckner[2]

[1]Institute of Computer Graphics and Algorithms, Vienna University of Technology, Vienna, Austria
[2]Department of Informatics, University of Bergen, Bergen, Norway

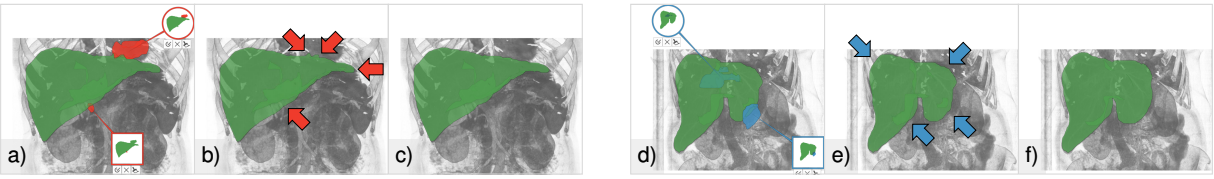

Figure 1: *We propose a novel guided volume editing approach for improving the quality of segmented medical data (Jaccard coefficient in percents). (a) Two suggestions to rectify over-estimation defects, with an initial quality of 88%, (b) after applying the suggestions, 91%, (c) after applying four more suggestions, 92%. (d) Two suggestions to fix under-estimation defects, with an initial quality of 80%, (e) after applying the suggestions, 85%, (f) after applying six more suggestions, 94%.*

Abstract

Segmentation of volumetric data is an important part of many analysis pipelines, but frequently requires manual inspection and correction. While plenty of volume editing techniques exist, it remains cumbersome and error-prone for the user to find and select appropriate regions for editing. We propose an approach to improve volume editing by detecting potential segmentation defects while considering the underlying structure of the object of interest. Our method is based on a novel histogram dissimilarity measure between individual regions, derived from structural information extracted from the initial segmentation. Based on this information, our interactive system guides the user towards potential defects, provides integrated tools for their inspection, and automatically generates suggestions for their resolution. We demonstrate that our approach can reduce interaction effort and supports the user in a comprehensive investigation for high-quality segmentations.

Categories and Subject Descriptors (according to ACM CCS): I.3.3 [Computer Graphics]: Picture/Image Generation—Display algorithms, I.3.6 [Computer Graphics]: Methodology and Techniques—Interaction techniques, I.4.6 [Image Processing and Computer Vision]: Segmentation—Edge and feature detection

1. Introduction

Many analysis tasks require the segmentation of volumetric data. Common examples include medical diagnosis and treatment planning, where individual organs or pathologies are labeled in order to compute quantitative information such as diameter or volume, or the segmentation of cell bodies in microscopy data. While there is a plethora of automatic and semi-automatic segmentation algorithms, they are frequently restricted to very specific scenarios and often require manual verification and adjustment. An example of typical segmentation defects in medical imaging is depicted in Figure 1. Most commonly, the user coordinates segmentation correction, actively searching for defects and editing them. We propose a novel approach for simplifying and accelerating correction. Our method extracts structural information from any initial segmentation and identifies potential defects automatically. These potential defects can be inspected by the user in 3D using integrated views and corrected by automatically provided suggestions. Our method enables **1)** *guidance towards an overall better quality* of the result by suggestions, based on a novel histogram dissimilarity analysis, **2)** *simple, yet effective volume editing* by reducing and unifying user interaction, and **3)** *structure-awareness* by accounting for the object's shape by means of the skeleton.

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd. Published by John Wiley & Sons Ltd.

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

2. Related Work

Many different volume editing techniques exist, but most of them do not actively guide the user throughout the correction process. Instead, the user pro-actively searches for the defects, corrects them and evaluates the results. The live wire approach, proposed by Mortensen and Barrett [MB95] and enhanced by Hastreiter and Ertl [HE98], allows the user to define and refine contours of the object, while investigating the volumetric data slice by slice. Sketch-based methods employ simple and intuitive sketching operations on slices, which modify the contours. One of these methods is proposed by Heckel et al. [HMTH13], where radial basis functions are used to construct an implicit smooth surface of the object out of several sketches. The work of Salz et al. [SRW*12] includes sketches over the object and the background, which are then used in geodesic segmentation of heart and lungs. Our previous work [KMS*13] propagates user selections on the skeleton towards the object voxels and allows the following editing operations: removing or adding the selected volume, peeling or growing the surface layer at the selection, and smoothing the selected surface patch. Graph cuts are widely used in volume editing [APB07, BFL06, LRN08]. In such techniques the user interactively marks a few parts of the object and the background. Then the object's boundary is found by a graph cut optimization procedure with respect to spatial and data value characteristics. Approaches that employ deformable 3D meshes, like the one proposed by Proksch et al. [PDP10], provide easy to use and efficient interaction tools for volume editing. Random walk methods [EZJS13, GFL06] detect the object's boundary from a small amount of seed points, efficiently improving the segmentation and requiring little user interaction. Heckel and Braunewell [HB14] propose a segmentation editing concept, based on interactive watershed transformation [HP03] combined with analysis of foreground and background markers provided by the user. Two recent methods [PRH10, THA11] direct the user to possible segmentation defects, but do not suggest how to correct them. Praßni et al. [PRH10] employ a random walk method and present ambiguous regions with high segmentation uncertainty in a 3D overview rendering and detailed uncertainty iso-lines in slice views. Top et al. [THA11] determine ambiguous regions of segmentation and display them in a slice view where the user can perform corrections with live wire editing. For an extensive survey on volume editing techniques in medical image processing, we refer the reader to the works of Heckel et al. [HMM*14, HMTH13].

Data collection and aggregation are common ways of summarizing data values over spatial regions of volume data. The idea is to obtain a discriminative entity that characterizes each such region. Cubes and spheres centered at a voxel are commonly used as regions of aggregation. Generic aggregation functions, which represent the whole region by a single data value, e.g., minimum, maximum and average, are used, for example, on vessel data by Mistelbauer et al. [MMV*13].

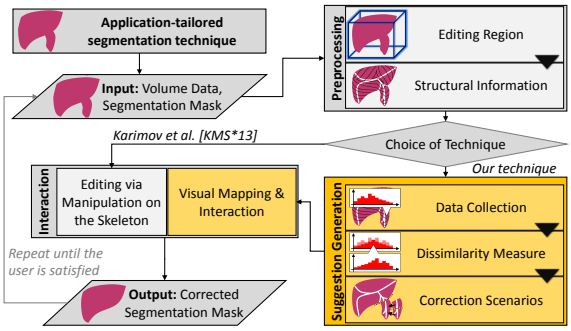

Figure 2: *Our main contribution is the integration of the suggestion generation and the visual mapping into the workflow of volume editing.*

A more general approach is to fit the collected data values to a known model such as a normal distribution. First, the fitting procedure checks whether the collected values correspond to the model by statistical tests and then determines the parameters of the model. A potential problem of such an approach is that the distribution model may be insufficient to describe parts of the object where segmentation defects are present. Haidacher et al. [HPB*10] propose to collect the data within a growing sphere around each voxel and fit it to a normal distribution. The statistical test they used can distinguish between a region with a single material and a region with multiple materials and stop the growing process accordingly. The statistics, however, become biased at the boundary of the object.

3. Guided Volume Editing

Motivated by the fact that volume editing is a laborious task with the last percentages of quality consuming the most of the time, we propose a technique that unifies the interaction and guides the user through the editing process. The workflow of our technique is depicted in Figure 2. We automatically generate suggestions (*correction scenarios*) about where (*correction region*) and how (*correction operation*) to correct potential segmentation defects. With these suggestions we guide the user towards the desired improved segmentation. An application-tailored segmentation technique, applied prior to the editing process, provides an approximate object segmentation as an input for our technique. We require this segmentation to roughly resemble the correct object in terms of major features. Hence, the basic assumption of our method is that defects in the segmentation are localized (e.g., an over-estimation into another organ) and we utilize the fact that the skeleton of the initial segmentation partially corresponds to the shape of the correct object. To simplify the explanation, we restrict our discussion to a binary segmentation, which labels the voxels of a volume as either object or background. Despite this, our approach equally supports scenarios where multiple objects are labeled.

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

In principle, we distinguish between the following two types of defects of a segmentation (see Figure 3):

1. *Over-estimation defects* (O) are regions of the volume that are incorrectly classified as belonging to the object.
2. *Under-estimation defects* (U) are parts of the object that are misclassified as being background.

Both classes of defects may be caused by low contrast between the object and the background or inhomogeneities of data values. Our approach aims to detect such inhomogeneities by analyzing data value distributions and incorporating the shape information provided by the initial segmentation. Hence, our method assumes that detectable differences of data values could be the cause of a misclassification that can be resolved by incorporating the domain knowledge of the user. We distinguish between defects that occur in the object's boundary layer (L) and those that affect whole parts of the object (P). This leads us to a total of four different types of defects (O_P, O_L, U_P, U_L). In order to identify them and provide automatic suggestions for their resolution, we analyze changes in the data value distribution along the structure of the initial segmentation using a histogram dissimilarity measure. Based on the nature of these changes, we identify the most likely type of defect and its corresponding correction scenario at multiple levels of detail. Overall, we provide an interactive visual interface that guides the user through the correction process. We now introduce the concepts of our method in detail.

3.1. Editing Region

The global *editing region* defines the domain of every subsequent editing operation. For O defects it is the set of all object voxels. To deal with U defects one could be tempted to just choose the whole background as the editing region. However, from a usability perspective this is problematic, as the multitude of structures in the background would generate a large amount of possible correction scenarios. Thus, we restrict the editing region to the vicinity of the object by constructing its convex hull and growing it away from its barycenter by a fixed amount. Removing the object voxels from this set yields the editing region for U defects, which we refer to as the *complement of the object*. Note that very large defects could reach beyond this region and would have to be corrected in several editing iterations. We found that a growth by 10% of the maximal dimension of the volume provides a good compromise. It is important that the correction of U defects of the object is conceptually identical to the correction of O defects in its complement; a fact that significantly simplifies the subsequent description of our method. The user chooses the editing region to correct, either the object or its complement.

3.2. Structural Information

We assume that the object has a certain stereotypical shape (e.g., because the object is a particular organ with known characteristics), so variations in its skeleton can provide valuable information about potential defects. Hence, our approach for determining correction scenarios operates on the skeleton of the editing region. We do not perform any skeleton post-processing, e.g., pruning, as it may destroy details of the shape, caused by segmentation defects. We adopt the concept of a skeleton distance field and influence zones, introduced in our previous work [KMS*13]. The *skeleton distance field* provides distances from all non-skeleton voxels to the closest skeleton voxels. For a given skeleton voxel, its associated *influence zone* can be thought of the set of all non-skeleton voxels that lie closest to this skeleton voxel, similar to a Voronoi region. The editing region is partitioned into influence zones, which emanate orthogonally from the region's skeleton. A synthetic dataset is illustrated in Figure 4. For a discussion on skeletonization, we refer the reader to the work of Reniers [Ren09]. We denote the influence zone of a skeleton voxel q as $IZ(q)$, the distance of a voxel p from the skeleton as $D_S(p)$, and the largest occurring distance as $I_{\max}$. Details can be found in our previous work [KMS*13]. The correction scenarios abstract the user from the structural information, providing a consistent editing experience throughout different objects.

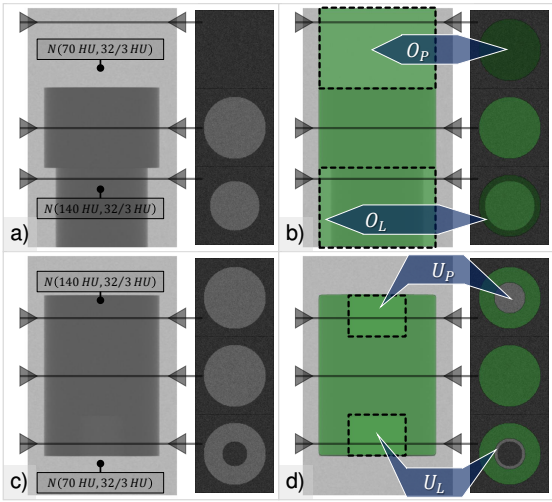

Figure 3: *Two phantom datasets with common segmentation defects: (a) object (dark gray), (b) segmentation (green) of the object from a) with O_P, O_L over-estimation defects, (c) object (dark gray), (d) segmentation (green) of the object from c) with U_P, U_L under-estimation defects.*

3.3. Data Collection

The data values are collected into histograms within the influence zones and the skeleton-distance iso-surfaces (the *collection regions*), which respect the structure of the editing region. We make no assumption regarding the underlying distribution of data values and directly compare the histograms to find dissimilarities among them, since they reveal segmen-

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

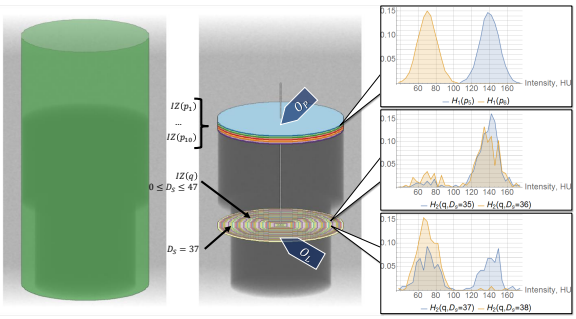

Figure 4: *Structure-aligned data collection using the skeleton and the influence zones. The histogram $H_1(p_i)$ collects data values in a whole influence zone $IZ(p_i)$ (to identify O_P defects). The histogram $H_2(q,j)$ collects data values at a skeleton distance $D_S = j$ in an influence zone $IZ(q)$ (to identify O_L defects). The histograms are normalized for illustration purposes. The bin size is 4 HU. Colors differentiate the influence zones and the skeleton-distance iso-surfaces.*

tation defects. For all histograms, we choose the same fixed bin size b. As illustrated in Figure 4, O defects result in additional structures along the skeleton (O_P) and "orthogonal" to it (O_L). To detect O_P defects, we collect the data values of the influence zone $IZ(p)$ of each skeleton voxel p into a corresponding histogram $H_1(p)$. Additionally, we generate histograms $H_2(p,d)$ for each iso-surface with skeleton distance d in $IZ(p)$. As shown in Figure 4, we expect the H_1 histograms to vary considerably along the skeleton in the presence of O_P defects, while O_L defects would cause an abrupt change of the H_2 histograms at the skeleton distance of the defect. At this point, the histograms are not normalized, as later we may apply the summation kernel on them.

3.4. Dissimilarity Measure

Using the collected histograms, we now compute two dissimilarity measures to detect segmentation defects: a dissimilarity value $\delta_1(p)$ (resp. $\delta_2(p)$), computed from H_1 (resp. H_2) histograms to detect O_P (resp. O_L) defects is assigned to each skeleton voxel p. Each value is given by a *dissimilarity measure* $\delta(S_H)$, evaluated on a subset S_H of the corresponding histograms. *Dissimilarity detection kernels*, specific to O_P and O_L defects, generate the subsets S_H.

We use the histogram intersection metric, a common bin-by-bin histogram comparison metric, equivalent to the L_1 distance. As it requires the histograms to be normalized, we perform this operation on the fly. We generalize the metric by intersecting all histograms in S_H (i.e., $h_\cap(l) = \min_{h_k \in S_H} h_k(l)$ for each bin l) and computing the intersection area (i.e., $d_\cap(S_H) = \sum_l h_\cap(l)$). The *dissimilarity measure* is now defined as $\delta(S_H) = 1 - d_\cap(S_H)$. It is normalized, i.e., it yields 0 for a set of identical histograms and 1 for no match between any two compared histograms. Other histogram metrics, such as the Kullback-Leibler divergence, the earth-mover's distance or the Kolmogorov-Smirnov statistic, would provide statistical information on dissimilarities. A generalization of these metrics to our scenario, which would respect mathematical statistics, requires significant theoretical work and is an interesting direction for future research.

Sometimes, a single histogram may contain too few values to be representative; hence, larger collection regions are required. For this purpose, we employ an iterative growing process during which we sum up nearby histograms bin-wise, i.e., we apply the summation kernel. This leads to the following input parameters for the dissimilarity detection algorithm: the detection kernels *sizes* and the *number* of the growth iterations. Different input data quality, objects, scanning resolutions and modalities complicate manual specification of these parameters. Hence, we propose an automatic tuning procedure. For this, we look at the average $\mu(\delta)$ over all dissimilarity values δ over the whole skeleton, where δ is either δ_1 or δ_2. Serving as a global characterization of the dissimilarities, a value of $\mu(\delta)$ close to 0 indicates a low dissimilarity. Hence, the editing region either has only subtle inhomogeneities and, probably, is free from defects (case 1) or the parameter selection suppresses the defect detection (case 2). If $\mu(\delta)$ is close to 1, most of the skeleton voxels have a high dissimilarity. This means that either the editing region is very inhomogeneous, inevitably causing false positives (case 3), or the parameters cause too much noise to be detected as defects (case 4). The goal of the parameter tuning is to achieve $\mu(\delta) \simeq 0.5$, which provides sufficient discrimination between homogeneous (low dissimilarity) and inhomogeneous (high dissimilarity) regions with respect to the actual data. We found that a two-phase approach is sufficient for our purposes: in the first phase, the detection kernel size is increased, which increases the number of histograms that are intersected when computing the dissimilarity measures $\delta(S_H)$. In the second phase, we sum neighboring histograms, enlarging the collection regions. Note that $\mu(\delta)$ monotonically increases in the first phase, whereas it monotonically decreases in the second phase.

For the dissimilarity $\delta_1(p)$ (for O_P defect detection), the set S_H is chosen from nearby histograms H_1 with a maximum distance of R_1 along the skeleton, i.e., $\delta_1(p) = \delta(\{H_1(q) | d(p,q) \le R_1\})$. In the first phase, we grow R_1 from its starting value 1 in steps of 1 until we hit the limit (10 in our case) or the kernel covers more than 3% of the skeleton voxels or $\mu(\delta_1) \simeq 0.75$, i.e., the midpoint between the target value 0.5 and the maximum. In the second phase, we grow the collection regions using histogram summation between the influence zones, i.e., $H_1(p) \leftarrow \sum_{d(p,q) \le 1} H_1(q)$ until $\mu(\delta_1) \simeq 0.5$, or the number of growth iterations equals R_1. This limit is due to the fact that the feature detection kernel should be larger than the grown regions.

The parameter tuning for the dissimilarity $\delta_2(p)$ is slightly different, because it is based on H_2 histograms of the skeleton-distance iso-surfaces of the influence zone

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

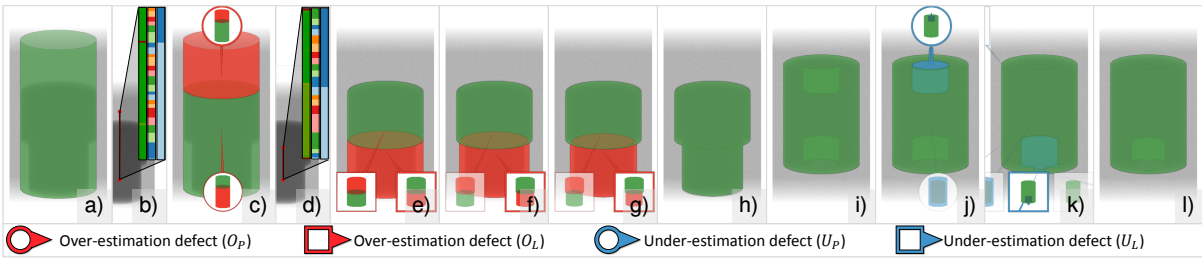

Figure 5: *Smart volume editing in the synthetic datasets of Figure 3: (a) the initial segmentation of object #1 with two defects, (b) skeleton view: the dissimilarity values δ_1 (green to red transition for $[0;1]$), the basins of the watershed transformation, and a level of the basins hierarchy, used in c), (c) the suggestion for the O_P defect, (d) skeleton view: the dissimilarity values δ_2 (green to red transition for $[0;1]$), the basins of the watershed transformation, and a level of the basins hierarchy, used in e), (e-g) the suggestions for the O_L defect, (h) the correct segmentation of object #1, (i) the initial segmentation of object #2 with two defects, (j) the suggestion for the U_P defect, (k) the suggestion for the U_L defect, (l) the correct segmentation of object #2.*

of p. First, the histograms $H_2(\cdot,0)$, $H_2(\cdot,1)$, $H_2(\cdot,2)$ are removed due to their insufficient information content (too few collected data values). The first phase is a single step in which the maximal kernel size is chosen by computing $\delta_2(p)$ over *all* skeleton distances, i.e., $\delta_2(p) = \delta(\{H_2(p,i)|2 < i < I_{\max}\})$. By summing histograms from neighboring iso-surfaces, i.e., $H_2(p,j) \leftarrow \sum_{|j-i|\leq 1} H_2(p,i)$, the second phase is continued until $\mu(\delta_2) \simeq 0.5$ or the iteration limit (again 10) is reached. Finally, Laplacian sharpening of δ_2 is performed along the skeleton to locate the borders of O_L defects between the influence zones.

Choosing a limit of 10 iterations, i.e., a maximal kernel size of 11 for δ_1 and up to 10 growth iterations proved sufficient for all real-world datasets presented in our results, although the actual tuning process typically uses fewer iterations. Note that these limits guarantee termination of the tuning procedure for cases 1 and 3, whereas the correct behavior in cases 2 and 4 is ensured by the monotonicity of the two phases with respect to the increase and decrease of $\mu(\delta)$.

3.5. Correction Scenarios

Having calculated the dissimilarity values $\delta_1(p)$ and $\delta_2(p)$ for each skeleton voxel p, we finally generate the *correction regions*. Each such region constitutes a set of voxels on which a correction operation will be applied. One could be tempted to just use thresholding on the dissimilarity values, but this would lead to a binary decision. We employ a hierarchical watershed transformation, following the work of Hahn and Peitgen [HP03], to support users with a variety of suggestions and smooth transitions between the levels of detail, thus offering more possibilities to fine-tune the results. Our approach has similarities to the work of Schultz et al. [STS07], which uses the watershed transformation over an anisotropy measure to delineate object features.

The watershed transformation splits the spatial data into basins with watersheds, placed at the ridges of data values. Applied on the skeleton voxels with δ_1 dissimilarities, it places the *watersheds* at the ridges of δ_1 values. The correction regions are created by propagating the *basins* on the skeleton towards the editing region by means of the influence zones. These regions are the O_P and U_P defects and the actual parts of the editing region, as shown in Figures 5b), 6b,c). The same logic applied on the skeleton voxels with δ_2 values provides the correction regions for O_L and U_L defects, as shown in Figure 5d). The most detailed level of the correction regions consists of all the initial basins.

Having generated a usually large number of basins for various defects, we proceed with the construction of a hierarchy. The main idea is that adjacent correction regions, which exhibit the same dissimilarities and correct the same type of defects, can be grouped together. This is the case in the presence of severe defects, which span over several influence zones and lead to a multitude of correction scenarios.

To simplify the adjacency check of correction regions, we add each watershed to every basin it delineates. Successively two basins are merged in each iteration, so that a new basin becomes a new correction region, which replaces two parental regions from the previous level of detail. To find the merged basins we use the distance metric $D_W(B_1,B_2) = W(B_1,B_2) - min(L(B_1),L(B_2))$, where B_1,B_2 are two basins, $W(B_1,B_2) = min_{r\in B_1\cap B_2}\delta(r)$ is the watershed level between them and $L(B_i) = min_{p\in B_i}\delta(p)$ is the bottom levels of B_i. Non-adjacent basins ($B_1 \cap B_2 = \emptyset$) cannot be merged. We always merge the pair of basins with minimal distance D_W. The merging stops if there are no more adjacent basins. The metric D_W guarantees that we merge basins with closer dissimilarities first. The hierarchy of the correction scenarios is illustrated in Figure 5b,d).

For each correction region we select a suitable operation to correct the potential defect at this region. For O_P defects, it is sufficient to simply remove the correction region from the editing region, as shown in Figure 5c). For O_L defects, the surface layer of the correction region is peeled away from the editing region, as illustrated in Figure 5e). The thickness of the peeled layer is fixed at two voxels.

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

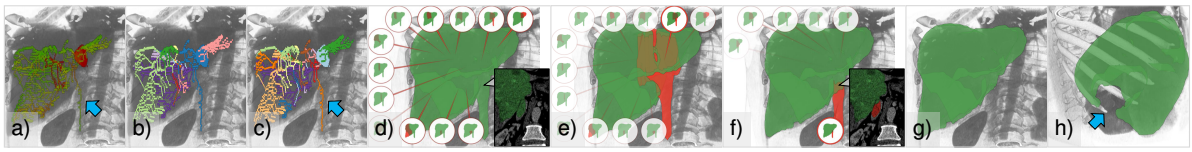

Figure 6: *(a) The dissimilarity values δ_1 (green to red transition for $[0;1]$), (b) A coarse level of the basins hierarchy, (c) A finer level of the basins hierarchy with the basin, which corresponds to the defect. The user interaction with our proposed technique: (d) the initial coarse level of detail, (e) the user selects correction scenario in the coarse level, (f) the user explores the finer level of detail and selects the finer correction scenario, which fixes the defect, (g) after applying the selected correction scenario, (h) example of defect, which is not properly captured by the skeleton. Each correction scenario is depicted by a single glyph.*

Each level of the hierarchy represents the editing region at a certain level of detail, determined as the ratio of the number of the correction scenarios at this level to the number of the initially created correction scenarios. This level of detail approach provides the user with a quality control of the correction scenarios. Low levels provide coarse corrections, affecting a large number of voxels, but leaving certain defects. High levels offer detailed correction scenarios, which affect only small local defects, but provide the required fine-tuning to reach an acceptable quality. Multiple different levels of detail of correction scenarios are illustrated in Figure 6e,f).

3.6. Visual Mapping and Interaction

We guide the user through the correction process by visually conveying the correction scenarios with glyphs. The object is displayed as a surface in 3D space, together with silhouettes to enhance shape perception. The surface is generated by the Marching Tetrahedrons algorithm.The context around the object is visualized by direct volume rendering. We desaturate the colors of the context in order to draw more attention to the object and the correction scenarios.

We only display glyphs for the correction scenarios at the current level of detail. Each glyph connects the projection of the correction region's barycenter (the anchor point) with a miniature that shows a maximum intensity projection rendering of the suggested changes. The background, the object, and the changes are mapped to data values of 0, 0.5, and 1 respectively; a color map is then applied. Similar to Ropinski et al. [RVB*09], we use a boundary layout to arrange the miniatures around the viewing area, so that the object and the suggested correction region remain unoccluded. We modulate the opacities of the glyphs with the relative depth of their anchor points. We use color-coding to differentiate between editing the object (red) and its complement (blue). To quickly visually convey our two types of defects (P,L), we encode them into the shape of the glyph (see Figure 5).

We propose the following correction workflow, illustrated in Figure 6d-g). First, the user operates on the 3D rendering, finding the appropriate correction scenario. Initially, a coarse level of detail is set. If the user hovers over the editing region, only glyphs representing the corresponding correction scenarios show up. If only a single correction scenario is suggested, it can be selected with a single mouse click. If the user hovers over a glyph, the boundary surface of the respective correction region is highlighted in order to provide a visual feedback. The correction scenario can also be selected by clicking on its glyph. Whenever a correction scenario is selected, a preview is shown to the user. Additionally, an integrated oblique slice view is provided at the anchor point of the selected scenario. The user can quickly analyze the results and, in case of doubt, refine the selection by scrolling with the mouse wheel in order to adjust the level of detail. The glyphs of the requested level of detail are displayed instantly. If the user selects a correction scenario, we display only descendants or ancestors of the selected scenario at a different level of detail. This offers the user the possibility to find the desired correction at a suitable level of detail. At any time, the user only gets a small amount of suggestions (up to 15) to avoid cluttering. To continue, the user applies the selected correction scenario, and the pipeline is repeated until the required quality is reached.

While the correction scenarios enable the efficient resolution of defects at multiple levels of detail, some segmentations may still require fine-grained control at the voxel level. This is the case for defects that partially occupy influence zones. Hence, our system provides an integrated oblique slice view as the highest level of detail, which allows users to inspect and edit minor local deficiencies of the selected scenario (Figure 7g). Even if a suggested correction is not perfect, the user still has the ability to fine-tune the resulting segmentation while benefiting from the guidance provided by our approach. To aid the user during the interaction, we allow panning, rotating, and scaling in the 3D view. Miniature 3D renderings are instantly updated in case of rotations. Panning and scaling have no effect on the miniatures, as they are automatically centered and scaled.

4. Implementation

Our system is implemented in C#, using Intel TBB (C++, CPU) and DirectX 11 Compute Shaders (HLSL, GPU) for parallelization. When calculating the complement of the object we apply the QuickHull algorithm by Barber et al. [BDH96] to construct the convex hull. The implementation is based on the work of Sehnal [Seh12] and runs on the CPU in a single thread. For the skeletonization we

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

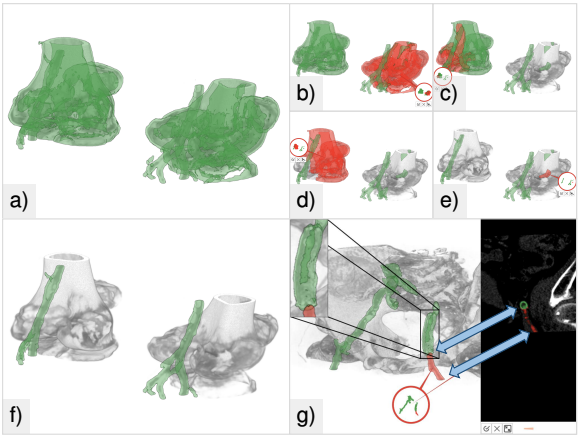

Figure 7: *Guided volume editing in CT-A data: (a) the automatic vessel segmentation exhibiting defects (vessels and bones are touching), (b-e) the first four correction operations, (f) the corrected vessel segmentation after only twelve operations, (g) correctly detected vascular occlusion.*

employ the algorithm by Lee et al. [LKC94]. The skeleton distance field and the influence zones are calculated using the algorithm from our previous work [KMS*13]. Between individual correction operations all data is recomputed, as all steps of the pipeline depend on the skeleton, which is a global characteristic of the corresponding editing region. The performance was estimated on twenty Computed-Tomography Angiography (CT-A) datasets with a slice resolution of 512×512 pixels and the number of slices ranged from 64 to 512. We used an Intel Core i7-2600K 3.4 GHz CPU with 16 GB of RAM and an NVidia GeForce 680 GPU. The pre-calculation time ranged from 1.8s to 9.1s with an average of 4.8s when correcting the object and from 2.8s to 10.4s with an average of 5.5s when correcting its complement. The dissimilarity analysis took most of the time. Even though we parallelize it on the CPU, a GPU implementation would further improve the performance of our system.

5. Results and Discussion

The only free parameter of our pipeline is the **bin size** b of the histograms. We tested different bin sizes 1, 2, 4, 8, 16 HU (Hounsfield Units) on all 4 types of defects, each defect on a specific CT-A dataset. The objects had data values from -50 HU to 250 HU. Our technique detected the defects properly for every bin size and provided appropriate correction scenarios. We suggest $b = 4$ HU to balance computation time, memory usage and a small number of level of detail adjustments (see supp. material #1 for details).

We demonstrate the **robustness** of our technique with respect to the **low contrast** between object and defects in supp. material #2. Robustness against **noise** is investigated in supp. material #3. Even at the peak signal-to-noise ratio of 19 dB, our technique can detect and correct segmentation defects.

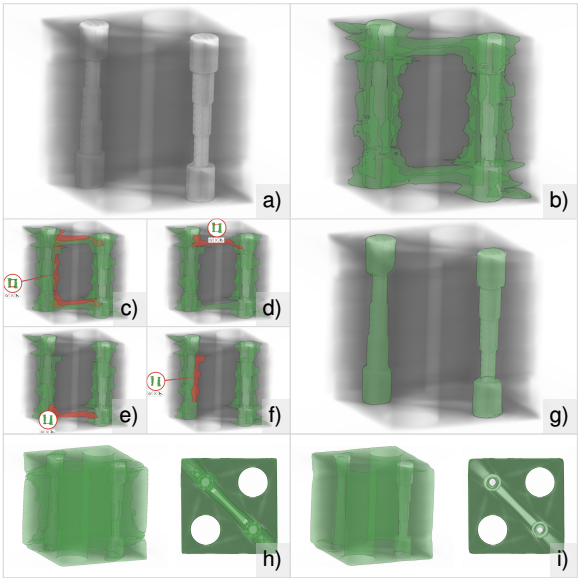

Figure 8: *Guided volume editing in industrial XCT data: (a) the dataset, (b) the initial segmentation of the two metal rods, (c-f) the first four of seven correction operations, (g) the correct segmentation of the metal rods, (h) the initial segmentation of the plastic box, (i) the correct segmentation of the box, after only nine operations.*

The study in supp. material #4 shows that our technique offers a consistent editing process even in case of **different initial segmentations**.

5.1. Application Scenarios

We demonstrate the **generality** of our technique by editing objects in medical, biological, and industrial data. In the **medical** scenario, we applied our technique to liver segmentation (see Figure 1) and vessel segmentation (see Figure 7). Both cases, acquired with CT-A, are clinically relevant and benefit from our tool. One common task in the latter case is the separation of vessels from bone when they are touching each other and automatic segmentations regularly fail. The usual procedure involves manually specifying separation objects – a tedious and time-consuming process. With our technique, this scenario was corrected with only twelve simple editing operations (see Figure 7f) in about 2m. The second example shows a vascular occlusion and, surprisingly, our technique provides the insight that the occlusion is even prolonged inside the stent, which hints to its potential usefulness even in segmentation-unrelated applications (see Figure 7g). The **industrial** scenario is represented by X-ray Computed Tomography (XCT) data (Figure 8), featuring significant metal artifacts, rendering the results of common segmentation techniques impractical. Our method allows to easily correct the segmentations of the metal rods (about 100s) and the plastic box (about 4m) – the latter significantly

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

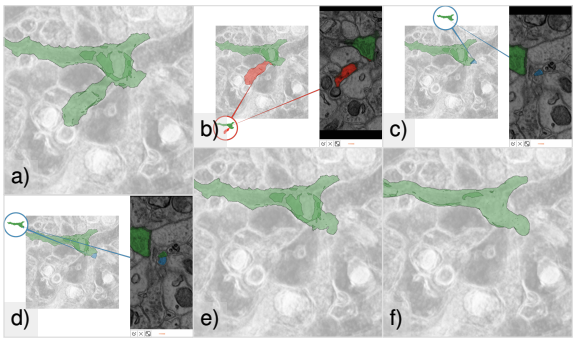

Figure 9: *Guided volume editing in electron microscopy data: (a) the neuron, segmented automatically, (b-d) three correction steps, (e) the segmentation of the neuron, corrected without slice editing, (f) the segmentation of the neuron, created manually by the domain expert.*

more challenging. As for the **biological** scenario, we apply our technique to Electron Microscopy (EM) data of a mouse cortex, provided by the ISBI 2013 challenge "SNEMI3D: 3D Segmentation of Neurites in EM images" (Figure 9). The automatic technique described by Kaynig et al. [KVRKB*13] provided the segmentation of the neurons, which then requires proofreading (i.e., manual inspection and correction), as explained by Haehn et al. [HKBR*14]. The data suffers from high noise levels and slicing nature of the acquisition technique, leading to difficulties in identifying cell boundaries. When correcting the segmentation of a particular neuron, our technique allows the user to isolate over-estimated regions as well as to recover under-estimated ones in only three simple operations in about 84s – an otherwise cumbersome and time-consuming manual task. We show the result (Figure 9e) before the supported slice-based refinement.

5.2. Evaluation

We evaluate our method with respect to usability, guidance, interaction, and quality of the results. We conducted a user study with three radiologists, who have experience in segmenting various organs and correcting their segmentation masks, since the results of their currently employed methods do not meet the required quality standards. Two experts analyze data for surgical intervention planning, whereas the third assesses treatment results comparing data before and after treatment. The task of the study was to correct the segmentation of the liver in abdominal CT-A data. We used 26 datasets, including the material of the SLiver 2007 contest [HvGS*09] and datasets from the routine practice of our domain experts. For the initial segmentation, we employed a technique based on the work of Šrámek and Dimitrov [vD03]. This technique merges watershed segments using data values and gradients. Before the evaluation, we introduced the experts to our system with 8 datasets. A skilled user presented all types of correction scenarios to provide a basic understanding and overview. Then, the experts were introduced to the workflow of the system, as well as to the method for suggestion generation. Interaction examples were presented. Finally, the experts tried the system themselves on the training datasets.

Afterwards, each expert corrected datasets with our technique (T1) and compared it to our previous technique (T2) [KMS*13], and base-line segmentation techniques of their choice: interactive editing with *Philips Brilliance* software (T3) and *Toshiba Vitrea* software (T4), manual editing with *General Electric LightSpeed* software (T5). As T3, T4, and T5 do not allow importing custom segmentation masks, the experts had to re-create masks beforehand. We excluded timings and efforts of such operations to conduct a fair comparison. Each domain expert was assigned 18 unique pairs, consisting of one dataset (each assigned only once) and one technique (each used 6 times). After the experts finished their corrections, they answered our questionnaire. Even after only a short accommodation period of a few hours, the domain experts rated our system favorably against well-established clinical tools of their workflow.

5.2.1. Technique Evaluation

We evaluated the usability and the guidance provided by our technique, as well as the following features: the levels of detail, the suggestions, and the miniatures (see Figure 10a). Our system's average score on the System Usability Scale by Brooke [Bro96] was 78 out of 100, which corresponds to the 82nd percentile of the perceived **usability**. Considering the relatively short accommodation period, this poses an overall good result and demonstrates the potential of our approach. The domain experts considered the overall user **guidance** through the correction process as useful. The experts specifically asked whether such guidance could be implemented in other editing methods as well. Since our system describes a fairly general concept of structure-aware editing, we are confident that it can be adapted to other applications. The outcome shows that there is room for improving the **overview** provided by our technique. The domain experts mentioned to display the suggestions for an object together with the complement at the same time. A hierarchy visualization of the correction scenario space would be a useful extension, which may help the user in finding the required correction scenarios faster. The domain experts rated the possibility to adjust the **level of detail** for the correction as very useful. With this feature and its immediate visual feedback within the system's interface, the experts were able to quickly select the correction scenario at the desired level of detail. The experts specifically mentioned that the **suggestions** of our system allow them to focus more on the fine-tuning, analysis and verification of the corrections. The **miniatures** were well accepted by the participants, as they provide an overview and simplify the exploration of the correction scenarios. The experts confirmed that our technique, combined with the automatically shown slice views, provides sufficient information to make **decisions** in segmentation correction.

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

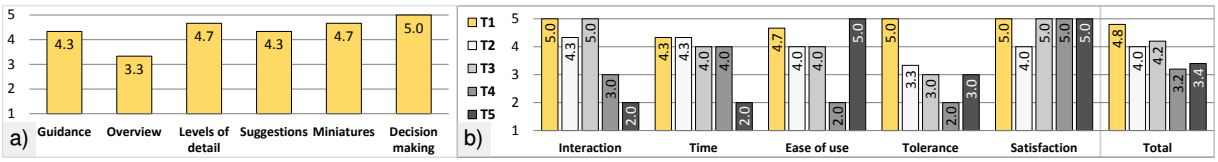

Figure 10: *(a) Evaluation of our technique. (b) Comparative evaluation. Grades for a) and b) range from 1 (worst) to 5 (best).*

5.2.2. Comparative Evaluation

We compared the volume editing techniques in terms of interaction, time spent, ease of use, tolerance to imprecise user input, and user satisfaction (see Figure 10b). Our technique T1 requires the least **interaction**, as it suggests suitable correction scenarios, resulting in a reduction of data exploration time and a reduced need for manual correction. The necessary interaction **time** is shortest with our method T1, since the user explores the correction scenarios, fine-tunes and verifies the corrections instead of manually specifying the correction regions. The experts noted, that slice-based editing, like in T5, is always **easier to use**, than 3D-space-based editing. Nevertheless, our technique is rated second, being just slightly more complex than T5. As we present corrections scenarios to the user, voxel-wise interactions are avoided, leading to increased **tolerance** to imprecise user input in comparison to the other evaluated techniques. Since the domain experts still favor the possibility to edit on the voxel level (even if only in rare cases), they have rated our technique T1 and their clinical methods T3, T4, and T5 as **completely satisfying** regarding the achievable quality of the results. With no voxel-level editing available, the experts reached less satisfying results with technique T2.

5.2.3. Quantitative Evaluation

We analyzed the defect **detection and correction rates** to quantify the accuracy of our technique. The first domain expert had no problems with correcting O defects (12 O_P and 8 O_L), but left 1 out of 5 U_P and 1 out of 2 U_L defects uncorrected. The second expert successfully corrected O defects (1 O_P and 9 O_L), but did not correct 1 out of 11 U_P and 3 out of 5 U_L defects. While correcting two of these U_L defects, the expert had an issue with the lowest level of detail. The third domain expert corrected all but one U_P defect out of 8 O_P, 6 O_L, 5 U_P and 3 U_L defects. As each expert left one U_P defect uncorrected, we investigated the issue. Since these three U_P defects are not surrounded by the object, the complements capture the defects only partially, and correction requires multiple steps, which were not straightforward to our experts. An example of such defect is the large tumor in the liver in Figure 6h). This issue is not specific to our method, since highly competent domain experts had difficulties in correcting these defects with other techniques too. Modulation of growth of the complement by the dissimilarities between parts of the background and the object could resolve the issue. As we support manual local editing, all remaining defects could be fixed to complete satisfaction. Nevertheless, the results, presented in this paper, were obtained without such low-level editing. To evaluate the **efficiency** of the techniques, we recorded the interaction time. As for the **accuracy**, we evaluated the quality with the Jaccard coefficient of the resulting segmentation with a ground truth, created manually by domain experts. A perfect match yields 1. The coefficient decreases, as differences appear. Before the correction, the average quality was at 0.83. With our technique T1, the experts achieved an average quality of 0.92 in 40s on average. The same average quality was reached with T2, but it took 1m on average. The quality and the interaction time is comparable to results of the interactive method by Beichel et al. [BBB*07] (before MBR stage). With T3 and T4 the experts achieved an average quality of 0.91 in 100s and of 0.88 in 150s respectively. The technique T5 achieved the best average quality (0.94), but it took much longer (13m on average). We can conclude that the **quality** of the suggestions is comparable to the quality of manual corrections, finely tuned by the domain experts with T2. Figure 1 depicts the correction of each of the four defect types.

6. Conclusion

We proposed a novel approach for guided volume editing of segmentation results based on the analysis of histogram dissimilarities. Considering the shape of the object by means of the skeleton, we allow users to correct parts of the object that contain segmentation defects. Our system makes suggestions, which guide users through the entire correction process. The presented results show that our system provides an efficient segmentation correction, while offering the possibility to reach the desired quality with a small number of simple interactions in a short amount of time.

Acknowledgments

We acknowledge M. Reiter and the research group XCT of the University of Applied Sciences, Upper Austria (TK08 dataset, industrial case). The medical datasets are courtesy of the SLiver 2007 contest (acknowledging T. Heimann, B. van Ginneken, and M. Styner), and the KFJ Hospital of Vienna, Austria. The EM biological data is courtesy of the SNEMI3D contest (acknowledging I. Arganda-Carreras, H. S. Seung, A. Vishwanathan, and D. Berger), and its segmentation is courtesy of the Harvard Connectome group of J. Lichtman and H. Pfister. We acknowledge M. Waldner, M. Bernhard, P. Mindek, J. Schmidt. T. Auzinger was funded by an Austrian Science Fund grant (FWF P24600-N23).

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.

References

[APB07] ARMSTRONG C. J., PRICE B. L., BARRETT W. A.: Interactive segmentation of image volumes with Live Surface. *Computer & Graphics 31*, 2 (2007), 212–229. 2

[BBB*07] BEICHEL R., BAUER C., BORNIK A., SORANTIN E., BISCHOF H.: Liver segmentation in CT data: A segmentation refinement approach. In *Proc. of MICCAI Workshop on 3D Segmentation in The Clinic* (2007), Heimann T., Styner M., van Ginneken B., (Eds.), pp. 235–245. 9

[BDH96] BARBER C. B., DOBKIN D. P., HUHDANPAA H.: The quickhull algorithm for convex hulls. *ACM Trans. Math. Software 22*, 4 (1996), 469–483. 6

[BFL06] BOYKOV Y., FUNKA-LEA G.: Graph cuts and efficient N-D image segmentation. *Int. J. Comput. Vision 70*, 2 (2006), 109–131. 2

[Bro96] BROOKE J.: SUS: A "quick and dirty" usability scale. In *Usability Evaluation in Industry*, Jordan P. W., Thomas B., Weerdmeester B. A., McClelland A. L., (Eds.). London: Taylor and Francis, 1996, ch. 21, pp. 189–194. 8

[EZJS13] EL-ZEHIRY N., JOLLY M., SOFKA M.: A splice-guided data driven interactive editing. In *Proc. IEEE Int. Symp. Biomed. Imaging* (2013), pp. 6–9. 2

[GFL06] GRADY L., FUNKA-LEA G.: An Energy Minimization Approach to the Data Driven Editing of Presegmented Images / Volumes. *Lecture Notes in Comput. Sci. 4191* (2006), 888–95. 2

[HB14] HECKEL F., BRAUNEWELL S.: A Concept for the Application of a Hierarchical Image Subdivision to the Segmentation Editing Problem. In *Proc. of MICCAI Workshop on Interactive Medical Image Computing* (2014). 2

[HE98] HASTREITER P., ERTL T.: Fast and Interactive 3D-Segmentation of Medical Volume Data. In *Proc. Image and Multidimensional Digital Signal Processing* (1998), pp. 78–85. 2

[HKBR*14] HAEHN D., KNOWLES-BARLEY S., ROBERTS M., BEYER J., KASTHURI N., LICHTMAN J. W., PFISTER H.: Design and Evaluation of Interactive Proofreading Tools for Connectomics. *IEEE Trans. Vis. Comput. Graphics 20*, 12 (2014), 2466–2475. 8

[HMM*14] HECKEL F., MOLTZ J. H., MEINE H., GEISLER B., KIEßLING A., D'ANASTASI M., DOS SANTOS D. P., THERUVATH A. J., HAHN H. K.: On the evaluation of segmentation editing tools. *Journal of Medical Imaging 1*, 3 (2014), 034005–01–034005–16. 2

[HMTH13] HECKEL F., MOLTZ J., TIETJEN C., HAHN H. K.: Sketch-Based Editing Tools for Tumour Segmentation in 3D Medical Images. *Comput. Graph. Forum 32*, 8 (2013), 144–157. 2

[HP03] HAHN H. K., PEITGEN H.-O.: IWT – Interactive Watershed Transform: A hierarchical method for efficient interactive and automated segmentation of multidimensional grayscale images. In *Proc. SPIE Med. Imaging* (2003), vol. 5032, pp. 643–653. 2, 5

[HPB*10] HAIDACHER M., PATEL D., BRUCKNER S., KANITSAR A., GRÖLLER M. E.: Volume Visualization based on Statistical Transfer-Function Spaces. In *Proc. PacificVis* (2010), pp. 17–24. 2

[HvGS*09] HEIMANN T., VAN GINNEKEN B., STYNER M., ARZHAEVA Y., AURICH V., BAUER C., BECK A., BECKER C., BEICHEL R., BEKES G., BELLO F., BINNIG G. K., BISCHOF H., BORNIK A., CASHMAN P., CHI Y., CORDOVA A., DAWANT B. M., FIDRICH M., FURST J. D., FURUKAWA D., GRENACHER L., HORNEGGER J., KAINMÜLLER D., KITNEY R., KOBATAKE H., LAMECKER H., LANGE T., LEE J., LENNON B., LI R., LI S., MEINZER H.-P., NÉMETH G., RAICU D. S., RAU A.-M., VAN RIKXOORT E. M., ROUSSON M., RUSKÓ L., SADDI K. A., SCHMIDT G., SEGHERS D., SHIMIZU A., SLAGMOLEN P., SORANTIN E., SOZA G., SUSOMBOON R., WAITE J. M., WIMMER A., WOLF I.: Comparison and evaluation of methods for liver segmentation from CT datasets. *IEEE Trans. Med. Imag. 28*, 8 (2009), 1251–1265. 8

[KMS*13] KARIMOV A., MISTELBAUER G., SCHMIDT J., MINDEK P., SCHMIDT E., SHARIPOV T., BRUCKNER S., GRÖLLER E.: ViviSection: Skeleton-based Volume Editing. *Comput. Graph. Forum 32* (2013), 461–470. 2, 3, 7, 8

[KVRKB*13] KAYNIG V., VAZQUEZ-REINA A., KNOWLES-BARLEY S., ROBERTS M., JONES T. R., KASTHURI N., MILLER E., LICHTMAN J., PFISTER H.: Large-Scale Automatic Reconstruction of Neuronal Processes from Electron Microscopy Images. `arXiv:arXiv:1303.7186v1`. 8

[LKC94] LEE T. C., KASHYAP R. L., CHU C. N.: Building skeleton models via 3D medial surface/axis thinning algorithms. *Lecture Notes in Comput. Sci. 56* (1994), 462–478. 7

[LRN08] LIU L., RABER D., NOPACHAI D.: Interactive separation of segmented bones in CT volumes using graph cut. *Lecture Notes in Comput. Sci. 5241* (2008), 296–304. 2

[MB95] MORTENSEN E. N., BARRETT W. A.: Intelligent scissors for image composition. In *Proc. SIGGRAPH* (1995), pp. 191–198. 2

[MMV*13] MISTELBAUER G., MORAR A., VARCHOLA A., SCHERNTHANER R., BACLIJA I., KÖCHL A., KANITSAR A., BRUCKNER S., GRÖLLER E.: Vessel Visualization using Curvicircular Feature Aggregation. *Comput. Graph. Forum 32* (2013), 231–240. 2

[PDP10] PROKSCH D., DORNHEIM J., PREIM B.: Interaktionstechniken zur Korrektur medizinischer 3D-Segmentierungen. In *Bildverarbeitung für die Medizin* (2010), vol. 574 of *CEUR Workshop Proceedings*, pp. 420–424. 2

[PRH10] PRAßNI J.-S., ROPINSKI T., HINRICHS K.: Uncertainty-Aware Guided Volume Segmentation. *IEEE Trans. Vis. Comput. Graphics 16*, 6 (2010), 1358–65. 2

[Ren09] RENIERS D.: *Skeletonization and Segmentation of Binary Voxel Shapes*. PhD thesis, Technische Universiteit Eindhoven, 2009. 3

[RVB*09] ROPINSKI T., VIOLA I., BIERMANN M., HAUSER H., HINRICHS K.: Multimodal Visualization with Interactive Closeups. In *Proc. EGUK (Theory and Practice of Computer Graphics)* (2009), pp. 17–24. 6

[Seh12] SEHNAL D.: MIConvexHull, 2012. Published: 21.11.2012. Accessed: 16.03.2014. URL: `http://miconvexhull.codeplex.com/`. 6

[SRW*12] SALZ P., RESKE A., WRIGGE H., SCHEUERMANN G., HAGEN H.: User-guided Segmentation of Thoracic Computed Tomography Data for Electrical Impedance Tomography Image Reconstruction. Poster at the 2nd IEEE Symposium on Biological Data Visualization, 2012. 2

[STS07] SCHULTZ T., THEISEL H., SEIDEL H.-P.: Segmentation of DT-MRI Anisotropy Isosurfaces. In *Proc. EuroVis* (2007), pp. 187–194. 5

[THA11] TOP A., HAMARNEH G., ABUGHARBIEH R.: Spotlight: Automated confidence-based user guidance for increasing efficiency in interactive 3D image segmentation. *Lecture Notes in Comput. Sci. 6533* (2011), 204–213. 2

[vD03] ŠRÁMEK M., DIMITROV L.: Segmentation of Tomographic Data by Hierarchical Watershed Transform. *Journal of Medical Informatics and Technologies 3* (2003), 161–169. 8

© 2015 The Author(s)
Computer Graphics Forum © 2015 The Eurographics Association and John Wiley & Sons Ltd.