Toward Automated Validation of Sketch-Based 3D Segmentation Editing Tools

Frank Heckel[1], Momchil I. Ivanov[1,2], Jan H. Moltz[1], and Horst K. Hahn[1,2]

[1] Fraunhofer MEVIS, Universitaetsallee 29, 28359 Bremen, Germany
frank.heckel@mevis.fraunhofer.de
http://www.mevis.fraunhofer.de
[2] Jacobs University, Campus Ring 1, 28759 Bremen, Germany

Abstract. Segmentation is one of the main tasks in medical image analysis. Measuring the quality of 3D segmentation algorithms is an essential requirement during development and for evaluation. Various methods exist to measure the quality of a segmentation with respect to a reference segmentation. Validating interactive 3D segmentation approaches or methods for 3D segmentation editing is more complex, however. Using interactive tools, the user plays a central role during the segmentation process as he or she needs to react on intermediate results, making established static validation approaches insufficient. In this paper we present a method to automatically generate plausible user inputs for 3D sketch-based segmentation editing algorithms, to allow an objective and reproducible validation and comparison of such tools. The user inputs are generated iteratively based on the intermediate and the reference segmentation, while static quality measurements are tracked over time. We present first results where we have compared two segmentation editing algorithms using our framework.

Keywords: Validation, Evaluation, Interactive Segmentation, Segmentation Editing, Simulation, Automation.

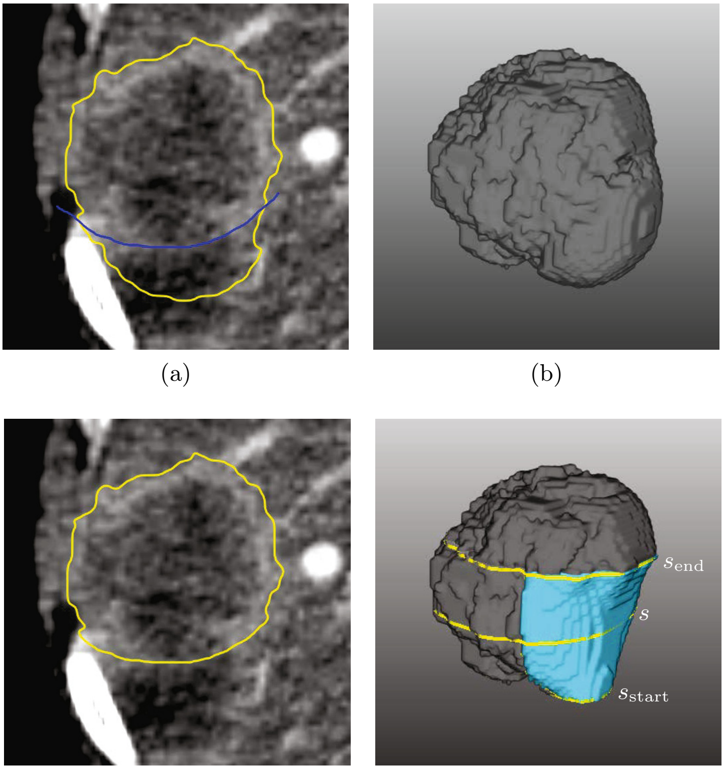
1 Introduction

The delineation of objects in images is one of the main tasks in image analysis. This process is called segmentation. Medical images are often acquired by CT[1] or MRI[2], resulting in 3D images given by a stack of parallel 2D *slices*. For the segmentation of objects in 3D medical images, many algorithms have been developed to solve this problem during the past decades [11,13]. In interactive 3D segmentation methods, a 3D segmentation is typically generated by a set of 2D user inputs on the slices of the image. Consequently, each input modifies the segmentation result in 3D. Some methods even allow the user to modify the result in any slice of any multi-planar reformatting (MPR), which we refer to as *view*. Segmentation editing can be seen as a special case of interactive segmentation.

[1] Computed Tomography.

[2] Magnetic Resonance Imaging.

J.-K. Kämäräinen and M. Koskela (Eds.): SCIA 2013, LNCS 7944, pp. 256–265, 2013.



(a) (b)

(c) (d)

Fig. 1. Sketch-based editing example for a liver metastasis in CT where the segmentation leaked into an adjacent metastasis, which should be cut away: (a) initial segmentation (yellow) and sketch-based user input (blue) in a 2D slice s, (b) segmentation before correction in 3D, (c) editing result in s and (d) 3D result after applying our image-independent manual correction algorithm [3], where s_{start} and s_{end} indicate the 3D influence of the editing step.

In contrast to general interactive segmentation, segmentation editing typically starts with an *initial* segmentation that the user locally corrects until it matches his or her needs. The initial segmentation is given by a dedicated automatic or semi-automatic algorithm that can be independent of the editing tool.

We have previously shown that sketching provides an intuitive 2D interface for segmentation editing in the contour-domain. Based on this 2D editing, we have developed an image-based [5] as well as an image-independent method [3] for intuitive an efficient segmentation editing in 3D in the context of tumor segmentation in CT. The image-based method iteratively simulates the sketch-based user input on the neighboring slices using a block matching followed by a shortest path approach on gradients within the image. The image-independent method reconstructs a new surface based on the user input and the initial segmentation using an object reconstruction approach that we have discussed earlier

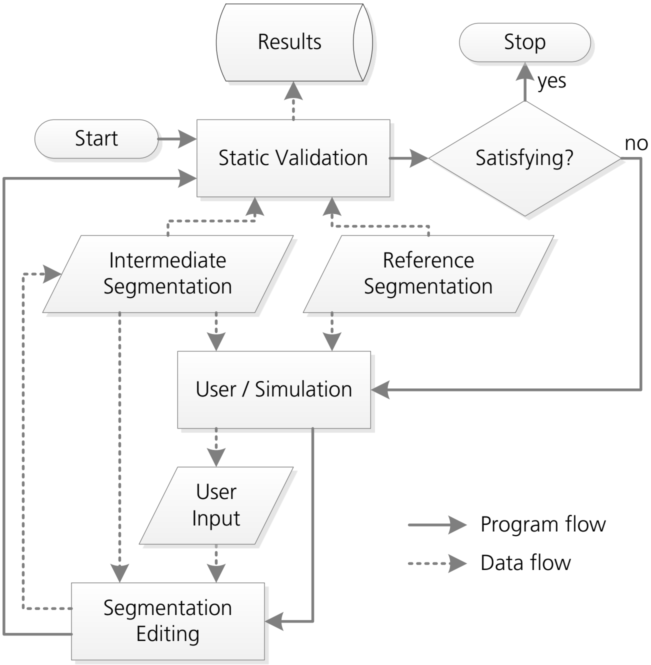


Fig. 2. Segmentation editing and simulation process: An initial segmentation is iteratively edited by the user or the simulation using a dedicated editing tool until its quality is sufficient. During this process the quality of the intermediate results with respect to the given reference segmentation is tracked.

[4]. Figure 1 shows an example of a sketch-based editing step using our image-independent algorithm.

Validation refers to the process of evaluating whether a software satisfies specific requirements [1], for instance the quality of the segmentation result with respect to the given segmentation problem. Validation is not only an important tool to decide which algorithm solves a given problem best. It is also essential during development for regression testing or parameter optimization. Various methods exist that measure the similarity of a segmentation result to a *reference* segmentation, i.e., its *quality*. We refer to this as *static* validation in this paper. Common static quality measures include volume-based metrics, like the volume overlap (Jaccard coefficient), as well as surface-based metrics, like the mean and maximum surface distance (Hausdorff distance) [6]. Reference segmentations are often given by manual delineations, which are used as a surrogate for the unknown ground truth.

Using interactive tools, the final segmentation result is given by a *process*, however, in which the user plays a central role. On the one hand, the segmentation result strongly depends on the user's input. On the other hand, the input itself depends on all previous *intermediate* results. As a consequence, changes in the underlying segmentation algorithm not only change the intermediate results but also the user inputs to the algorithm that are necessary to converge to the user's intended result. This makes a user mandatory for testing and comparing interactive algorithms or differing versions of the same algorithm. When evaluating interactive tools, it is insufficient to validate the final result only, because the segmentation task is a user-driven, dynamic process. The quality of such tools is influenced by additional factors, like the number of interactions, and their acceptance suffers from bad intermediate results. Comparing results from various user studies has a limited reliability, though, because of the bad reproducibility of manually or interactively generated segmentation results. Even if the same object is segmented twice by the same user using the same interactive tool, these results differ from each other due to the intrinsic intra-observer variability. All these facts make established (static) validation approaches unsuitable for measuring the quality of interactive segmentation as well as segmentation editing methods. However, almost no research has been done in this field so far.

In order to allow an objective, reproducible validation and comparison of 3D segmentation editing tools, without the necessity of the user, we propose a *dynamic* validation approach that simulates the user in the context of sketch-based editing (see Fig. 2). Inspired by the work of Moschidis and Graham as well as McGuinness and O'Connor [10,8], plausible user inputs are generated iteratively based on the intermediate and the reference segmentation as shown in Fig. 3, while various quality measures are tracked over time.

2 Related Work

Udupa et al. have summarized challenges in the evaluation of segmentation algorithms in the context of medical imaging [12]. The authors also propose a general methodology for the evaluation of such algorithms, including requirements, its implementation and performance metrics, i.e., quality measures. However, specific challenges for interactive approaches are not discussed by Udupa et al.

Existing work on evaluation of interactive segmentation methods focuses on "scribble-based" approaches like graph-cuts or random walker, where the user draws foreground and background markers to influence the result. McGuinness and O'Connor have investigated the evaluation of such algorithms for 2D natural images [7]. Later the authors proposed a simulation-based automated evaluation for scribble-based methods in 2D [8]. For scribble-based interactive segmentation of 3D medical images, Moschidis and Graham proposed a simulation-based framework for performance evaluation [9] as well as a systematic comparison of various interactive segmentation methods [10].

We are not aware of any research on automatic validation of dedicated 3D segmentation editing tools, neither in simulating plausible sketch-based user inputs nor in measuring the quality of such tools with respect to their dynamic nature.

3 Sketch-Based Editing Simulation

Our automated validation is designed for sketch-based user interactions. Sketching provides an intuitive 2D interface to perform manual corrections, where the user modifies a binary segmentation result in the contour domain as shown in Fig. 1a. For details we refer to the description of our manual correction algorithms [5,3].

We call parts that are missing or which are unintentionally included in the segmentation *errors*. The editing simulation consists of two steps: finding the 3D error that the user would most probably correct and finding the view as well as the slice in which he or she might correct it in 2D. For simplicity, we assume that the user is allowed to correct exactly one error per correction step by adding or removing a part, although our sketching interface is not restricted to this. We further assume that correction can be done in axial, coronal or sagittal view.

3.1 Finding the Most Probably Corrected Error

In the first step, we compute all errors of the intermediate segmentation $\mathcal{S}_i$ with respect to the reference segmentation $\mathcal{R}$ by subtracting $\mathcal{S}_i$ from $\mathcal{R}$. Next, we compute all connected components in 3D using a 6-neighborhood to get all unique errors (see Fig. 3b). For each error a rating is computed that represents the probability of being corrected by the user. Based on our experience, users that are familiar with our correction tools tend to correct the most prominent errors in the current segmentation first. In order to model this we propose a volume-based and a surface-distance-based rating strategy.

The volume-based strategy selects the error $\mathcal{E}$ based on its volume V and its compactness C, i.e., by maximizing

$$r_V(\mathcal{E}) = \alpha \frac{V(\mathcal{E})}{V_{\max}} + \beta C(\mathcal{E}), \tag{1}$$

with $V_{\max}$ being the volume of the largest error. The compactness is defined as the volume-to-surface-area ratio, scaled to $[0,1]$. α and β allow adjusting the influence of the volume and the compactness. The surface area is approximated by the volume of all voxels on the surface of the segmentation result.

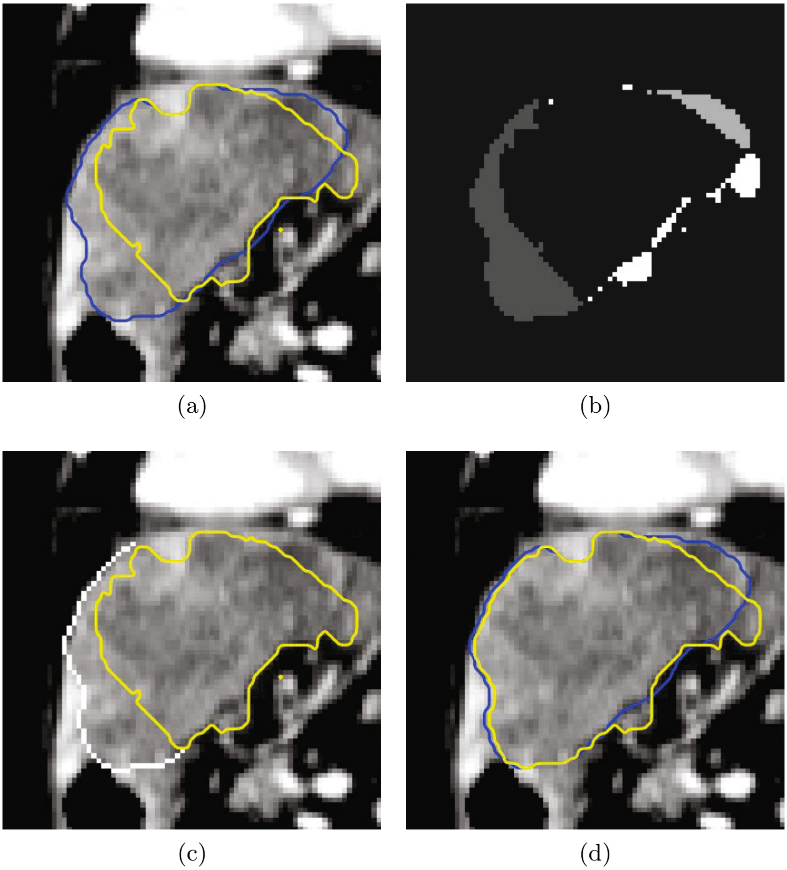
The surface-based strategy selects the error with the largest Hausdorff distance d_H with respect to $\mathcal{R}$:

$$r_D(\mathcal{E}) = d_H(\mathcal{E}, \mathcal{R}). \tag{2}$$

If exactly the same error is chosen in successive steps, it will be ignored, because in this case we have to assume that it could not be corrected by the editing algorithm.

3.2 User-Input Generation

The most probably corrected error $\mathcal{E}_j$ can be corrected by the user in any slice s in any view v. $\mathcal{E}_j$ might consist of several components in a slice. Therefore, we only


(a) (b)

(c) (d)

Fig. 3. Simulation example showing a liver metastasis in CT: (a) initial segmentation $\mathcal{S}_0$ (yellow) and reference segmentation $\mathcal{R}$ (blue), (b) errors in $\mathcal{S}_0$ (gray), (c) generated correction contour (white voxels) and (d) result after applying the image-based segmentation editing algorithm [5]

consider the largest connected component of $\mathcal{E}_j$ with respect to a 4-neighborhood in the following. Again, experienced users tend to correct the error were it is best seen, so we propose both an area- and a distance-based strategy for finding the most probable slice and view. In addition, our correction algorithms work best if the error is corrected roughly at its center in z-direction, which is the direction orthogonal to the view. We therefore prefer slices that are close to the center slice s_c of the error for our simulation.

The area-based strategy selects s and v by maximizing

$$r_A(s,v) = \alpha \frac{A(s,v)}{A_{\max}(v)} + \beta C(s,v) + \gamma \left(1 - \frac{|s - s_c(v)|}{e_z(v)}\right), \qquad (3)$$

where $A(s, v)$ and $C(s, v)$ are the area and the compactness of the error in the current slice of the current view, $A_{\max}(v)$ is the maximum area in the current view and $e_z(v)$ is the z-extent of the error in the current view. The weights α, β and γ allow adjusting the influence of each feature.

The distance-based strategy maximizes

$$r_D(s,v) = \alpha \frac{d_H(s,v)}{d_{H_{\max}}(v)} + \gamma \left(1 - \frac{|s - s_c(v)|}{e_z(v)}\right), \tag{4}$$

with $d_H(s, v)$ being the Hausdorff distance in the current slice of the current view and $d_{H_{\max}}(v)$ being the maximum Hausdorff distance of all slices of v.

Because users tend to correct in the same view as long as it is appropriate, we also apply a reward of 10% to r if the view is kept between successive steps.

Finally, a contour is generated that adds/removes the error to/from the intermediate segmentation. This contour is defined by all voxels $\widetilde{\mathcal{E}_j} \setminus \widetilde{\mathcal{S}_i}$, with $\widetilde{\mathcal{E}_j}$ and $\widetilde{\mathcal{S}_i}$ being all voxels on the surface of $\mathcal{E}_j$ and $\mathcal{S}_i$, respectively (see Fig. 3c) To generate a contour from those voxels, we assume the voxels to form graphs, where the voxels are the nodes, which are connected to all voxels in their 8-neighborhood. We then compute all longest paths in all graphs. To allow for small holes in the voxel representation of the contour, we additionally connect two adjacent paths, if the distance between their start and end points is smaller than 2 voxels. Note that this definition also covers the case where a segmentation is completely missing in a certain slice.

4 Results and Discussion

As a proof of concept, we applied our automated simulation to two segmentation editing tools on an exemplary liver metastasis shown in Fig. 3. Our goal was to show how the validation of sketch-based editing algorithms can benefit from the proposed framework. The editing approaches used during this evaluation were improved versions of the previously published image-based and image-independent algorithms [5,3]. The image-based algorithm has been extended by an optimized reference point placement based on image information, more advanced stopping criteria and a feature that allows it to consider previous user inputs. The image-independent approach has been extended by a step that resolves contradictory user inputs. We tracked the volume overlap, the Hausdorff distance as well as the size of the largest error for all intermediate results. For the results shown in Fig. 4, we set the weight of the compactness to $\beta = 1.5$, while all the other features were weighted by 1.

As expected, the results for a specific quality measure depend on the chosen simulation strategy. If the volume-based strategy is used, the size of the largest error decreases faster, while the Hausdorff distance benefits from the distance-based strategy. The overlap to the reference segmentation grows faster for the distance-based simulation, which indicates that the editing algorithms might benefit from this correction strategy. The plots show that the image-independent algorithm performs better in this example for all quality measures, so it seems

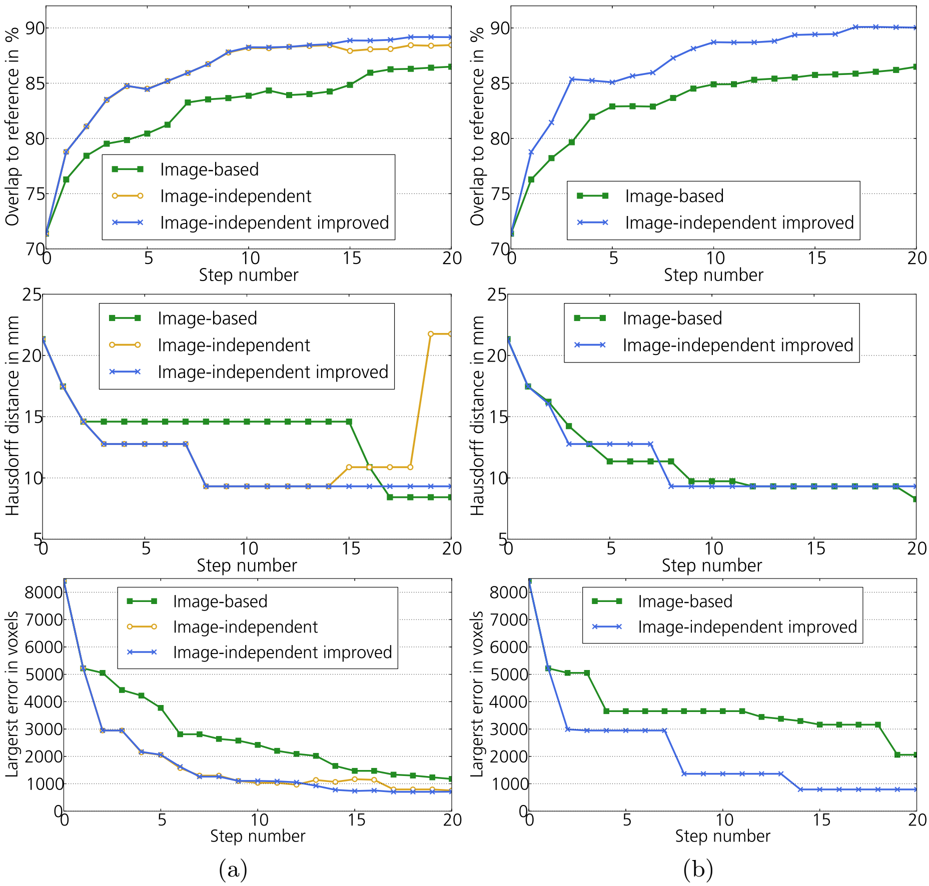


Fig. 4. Validation results for the liver metastasis example shown in Fig. 3: (a) volume-based and (b) distance-based simulation. Note the issues at step 15 and 19 of the image-independent editing algorithm before improvement.

to be the better choice for this specific segmentation problem. The plot of the Hausdorff distance revealed an issue of the image-independent algorithm in steps 15 and 19 that could be resolved as shown in the plots as well (see Fig. 4a). Note that this is hardly visible both in the overlap plot and the plot of the largest error. Assuming that the inputs to the image-independent correction algorithm have been stored during a user study, all inputs after step 15 would become invalid after modifying the algorithm so they cannot be used for evaluation anymore. Using the proposed simulation-based evaluation framework, new valid inputs can be generated after the algorithm has been modified.

In various interviews with radiologists from different clinics, we got the feedback that, due to time constraints, a maximum of 5 correction steps would be performed in clinical practice, at least in the context on oncological chemotherapy response monitoring, where the change of a tumor's size is measured.

The plots of our simulation support this, as the first 3 to 6 correction steps show the best improvement of the segmentation, depending on the quality measure. Following correction steps improve the segmentation results only slightly. Also note that for liver tumor segmentations a volume overlap of about 87% is considered to be within the variability of different users [2], which the image-independent algorithm reaches in the 8th step for this specific case.

5 Conclusion

We have discussed the validation of sketch-based segmentation editing algorithms for 3D medical images and we have presented an automated, dynamic validation approach for such tools that simulates the user. For generation of plausible user inputs based on the intermediate and a reference segmentation, we have proposed two strategies that utilize the volume and the Hausdorff distance. Our framework allows an objective and reproducible validation and comparison of sketch-based segmentation editing algorithms, without the necessity of the user, which we have shown in a first example for two different manual correction tools. It supports the development of 3D manual correction algorithms by allowing dynamic regression tests with multiple correction steps. Moreover, it yields the potential to better assess the quality of manual correction algorithms with respect to their dynamic nature.

6 Future Work

Future work could focus on solving current limitations and making the simulated inputs more realistic. For example, our simulation is not able to handle holes correctly and completely removing the segmentation in a slice is not supported. Users typically draw contours with a certain amount of inaccuracy. Modeling this property would allow drawing conclusions on the robustness of the segmentation editing algorithms to varying user inputs. Our simulation currently does not support that an error is corrected until it is solved before the next one is chosen, which some users do. Users that are not familiar with our editing algorithms sometimes tend to correct an error in the first slice where it appears. Simulating these user groups as well would further improve the value of our framework.

In addition, it needs to be investigated how the overall quality of segmentation editing algorithms can be measured best so it correlates with the user's subjective impression. Finally, we plan to apply our automated simulation to a larger, representative database in order to improve not only our simulation but also our editing algorithms.

References

1. Boehm, B.W.: Verifying and validating software requirements and design specifications. IEEE Software 1(1), 75–88 (1984)

2. Deng, X., Du, G.: Editorial: 3D segmentation in the clinic: A grand challenge II - liver tumor segmentation (2008), `http://www.grand-challenge2008.bigr.nl/proceedings/liver/articles.html`
3. Heckel, F., Braunewell, S., Soza, G., Tietjen, C., Hahn, H.K.: Sketch-based image-independent editing of 3D tumor segmentations using variational interpolation. In: Eurographics Workshop on Visual Computing for Biology and Medicine, pp. 73–80. Eurographics Association (2012)
4. Heckel, F., Konrad, O., Hahn, H.K., Peitgen, H.O.: Interactive 3D medical image segmentation with energy-minimizing implicit functions. Computers & Graphics: Special Issue on Visual Computing for Biology and Medicine 35(2), 275–287 (2011)
5. Heckel, F., Moltz, J.H., Bornemann, L., Dicken, V., Bauknecht, H.C., Fabel, M., Hittinger, M., Kießling, A., Meier, S., Püsken, M., Peitgen, H.O.: 3D contour based local manual correction of tumor segmentations in CT scans. In: SPIE Medical Imaging: Image Processing, vol. 7259, p. 72593L. SPIE (2009)
6. Heimann, T., van Ginneken, B., Styner, M.A., Arzhaeva, Y., Aurich, V., Bauer, C., Beck, A., Becker, C., Beichel, R., Bekes, G., Bello, F., Binnig, G., Bischof, H., Bornik, A., Cashman, P.M.M., Chi, Y., Córdova, A., Dawant, B.M., Fidrich, M., Furst, J.D., Furukawa, D., Grenacher, L., Hornegger, J., Kainmuller, D., Kitney, R.I., 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 Transactions on Medical Imaging 28(8), 1251–1265 (2009)
7. McGuinness, K., O'Connor, N.E.: A comparative evaluation of interactive segmentation algorithms. Pattern Recognition 43(2), 434–444 (2010)
8. McGuinness, K., O'Connor, N.E.: Toward automated evaluation of interactive segmentation. Computer Vision and Image Understanding 115(6), 868–884 (2011)
9. Moschidis, E., Graham, J.: Simulation of user interaction for performance evaluation of interactive image segmentation methods. In: Medical Image Understanding and Analysis, pp. 209–213 (2009)
10. Moschidis, E., Graham, J.: A systematic performance evaluation of interactive image segmentation methods based on simulated user interaction. In: IEEE International Symposium on Biomedical Imaging, pp. 928–931 (2010)
11. Pham, D.L., Xu, C., Prince, J.L.: Current methods in medical image segmentation. Annual Review of Biomedical Engineering 2(1), 315–337 (2000)
12. Udupa, J.K., LeBlanc, V.R., Zhuge, Y., Imielinska, C., Schmidt, H., Currie, L.M., Hirsch, B.E., Woodburn, J.: A framework for evaluating image segmentation algorithms. Computerized Medical Imaging and Graphics 30(2), 75–87 (2006)
13. Withey, D.J., Koles, Z.J.: Medical image segmentation: Methods and software. In: International Symposium on Noninvasive Functional Source Imaging of the Brain and Heart and the International Conference on Functional Biomedical Imaging, pp. 140–143 (2007)