DOI: 10.1111/cgf.12193

COMPUTER GRAPHICS *forum*
Volume 32 (2013), number 8 pp. 144–157

Sketch-Based Editing Tools for Tumour Segmentation in 3D Medical Images

Frank Heckel[1], Jan H. Moltz[1], Christian Tietjen[2] and Horst K. Hahn[1]

[1]Fraunhofer MEVIS, Germany
{frank.heckel, jan.moltz, horst.hahn}
@mevis.fraunhofer.de
[2]Siemens AG, Healthcare Sector, Imaging & Therapy Division, Computed Tomography, Germany
christian.tietjen@siemens.com

Abstract

In the past years sophisticated automatic segmentation algorithms for various medical image segmentation problems have been developed. However, there are always cases where automatic algorithms fail to provide an acceptable segmentation. In these cases the user needs efficient segmentation editing tools, a problem which has not received much attention in research. We give a comprehensive overview on segmentation editing for three-dimensional (3D) medical images. For segmentation editing in two-dimensional (2D) images, we discuss a sketch-based approach where the user modifies the segmentation in the contour domain. Based on this 2D interface, we present an image-based as well as an image-independent method for intuitive and efficient segmentation editing in 3D in the context of tumour segmentation in computed tomography (CT). Our editing tools have been evaluated on a database containing 1226 representative liver metastases, lung nodules and lymph nodes of different shape, size and image quality. In addition, we have performed a qualitative evaluation with radiologists and technical experts, proving the efficiency of our tools.

Keywords: segmentation, editing, correction, interaction, sketching, contours, 3D, CT

ACM CCS: Computer Graphics [I.3.5]: Computational Geometry and Object ModellingCurve; surface; solid; and object representations; Computer Graphics [I.3.6]: Methodology and TechniquesInteraction techniques; Image Processing and Computer Vision [I.4.6]: Segmentation

1. Introduction

In medical imaging, *segmentation* refers to the delineation of an object, e.g. a tumour, in a two-dimensional (2D) or three-dimensional (3D) image. Segmentation is one of the essential tasks in medical image analysis. It typically is the basis for all successive steps, such as classification, quantification or visualization and it can even have direct implications for the patient, like in radiotherapy planning. Therefore, a proper segmentation of objects in medical images is crucial in many cases. For solving the segmentation problem, many methods exist that differ in the amount of user-interaction as shown in Figure 1. In this regard, *interactive segmentation* refers to methods that are based on an iterative process in which the user plays a central role by steering and correcting the segmentation result. *Segmentation editing* can be seen as a special case of this. In contrast to general interactive segmentation, segmentation editing typically starts with an *initial segmentation* that the user locally corrects until it satisfies his or her needs.

Particularly in clinical routine, a fully manual segmentation takes too much time and lacks reproducibility. Therefore, automatic segmentation algorithms have been developed as an alternative for specific purposes. However, even sophisticated automatic algorithms fail to provide an acceptable segmentation in certain cases. For example, due to low contrast, noise and biological variability, the segmentation might leak into adjacent structures or parts of the objects are missing in the segmentation result. Nevertheless, most parts of the segmentation mask are usually correct and only small regions

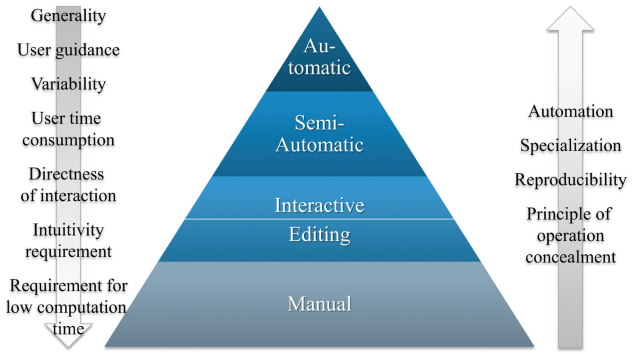


Figure 1: *Overview on properties and requirements of segmentation methods depending on the grade of automation.*

are erroneous, so locally correcting the errors is the most intuitive and efficient solution in such cases. This makes segmentation editing an indispensable step in the segmentation process, especially for methods used in clinical routine, as already stated by Heimann and Meinzer [HM09].

This paper gives a comprehensive overview on segmentation editing for 3D medical images. We discuss sketch-based segmentation editing in 2D, i.e. on the 2D slices of the 3D image. Sketching refers to a contour-based interaction, which is a common interface in computer-aided design where it is used for modelling objects in 3D [IMT99]. It has proven to be an intuitive interaction for segmentation editing and has been widely accepted by various clinicians in our evaluations. Based on this 2D editing, we present an image-based as well as an image-independent method for intuitive and efficient segmentation editing in 3D in the context of tumour segmentation in computed tomography (CT). The image-based method iteratively simulates the sketch-based user input on the neighbouring slices using image information. The image-independent method reconstructs a new surface based on the user input and the initial segmentation using an object reconstruction approach. We have proposed both methods previously [HMB*09], [HBS*12]. The contributions of this paper for the image-based algorithm are an optimized reference point placement based on image information, improved stopping criteria and an extension that allows the algorithm to consider previous inputs. The image-independent approach has been extended by a step that resolves contradictory inputs.

The proposed algorithms are targeting radiology workstations for oncological therapy response monitoring in CT, where the clinician typically interacts with 2D slices of the 3D images in order to assess the size of a tumour over time. Our goal was developing tools that allow correcting a segmentation with a few steps in a few seconds for the majority of cases. We have designed our algorithms as general segmentation tools that also enable the user to specify segmentations of objects from scratch, in cases where a dedicated (semi-)automatic algorithm has failed completely or is not available at all. Our tools can be used in any multi-planar reformatting, which we refer to as *view*. They even allow the user to arbitrarily switch between views and between both algorithms during the editing process.

2. Approaches to Segmentation Editing

Given an automatic or semi-automatic segmentation result, segmentation editing approaches in general can roughly be classified into three categories:

(1) *Parameter tuning*: One or more parameters of the (semi-) automatic algorithm are adapted. Changing parameters of the algorithm is a rather unintuitive approach. It has a global influence on the segmentation result and the user has to know at least some technical details about the underlying algorithm. Depending on the computation times of the algorithm, changing parameters might not always be possible at interactive speed.

(2) *Incorporate user information*: The segmentation algorithm computes new results using additional information interactively given by the user. This is often referred to as *manual segmentation refinement* in the literature. It is a common approach for algorithms that are based on shape models or that are automated ('minimally interactive') versions of fundamentally interactive approaches. Such tools can take advantage from the information of the automatic algorithm and they can thus generate better results than general manual correction tools. However, such editing methods only work for the specific segmentation algorithms and they often tend to change the segmentation globally. Particularly for shape models, including user information might not be possible at interactive speed.

(3) *Manual correction*: The segmentation is locally corrected using dedicated editing tools. These tools are independent of the initial segmentation method so they can be applied to various segmentation problems and algorithms.

In the context of interactive segmentation, a *direct* interaction refers to methods where the input data has direct influence on the (3D) segmentation result, usually leading to low-level interaction, as discussed by Olabarriaga and Smeulders [OS01]. In contrast, *indirect* inputs are interpreted by the segmentation method in order to provide high-level interaction and to reduce the amount of necessary interactions. Consequently, approaches in the manual correction category can further be subdivided into low- and high-level tools:

(3a). *Low-level tools*: If the user needs at least some basic knowledge about the segmentation method and its influence on the segmentation result, we refer to it as low-level. Low-level tools provide general editing solutions for a lot of segmentation problems. However, they are typically more time consuming and less intuitive if they perform modifications in 3D.

(3b). *High-level tools*: If the technical aspects of the method are hidden from the user, it is referred to as high-level. In contrast to low-level tools, high-level tools typically try to infer the user's intention on the segmentation result in 3D based on the information given during the editing process. Due to the fact that only a minimum amount of additional information is given by the user, segmentation editing is an ill-posed problem in this case. Consequently, high-level approaches are often heuristic to some extent, making them more application specific.

Table 1 summarizes these properties. As stated by Olabarriaga and Smeulders in the context of interactive segmentation, manual correction needs high-level tools for being efficient [OS01].

Table 1: *Overview on segmentation editing approaches.*

	Parameter tuning	Incorporate user information	Low-level manual correction	High-level manual correction
Interaction	Direct	Indirect	Direct	Indirect
Influence	Global	Global or local	Local	Local
Performance requirement	Moderate	Fast	Real-time response	Real-time response
Generality	Algorithm specific	Algorithm specific	General	General
Reproducibility	High	Medium	Low	Medium
Intuitivity	Low	High	Medium	High
Technical knowledge required	Yes	No	No (2D)/Yes (3D)	No

High-level manual correction in 3D is a challenging task, however, where as many segmentation problems as possible should be solved with as few interactions as possible. Hence, the tool needs to estimate the user's intention in 3D while it should modify the segmentation only locally and the algorithm needs to be fast enough to react on inputs in real time. Both the required accuracy and the amount of time spent for editing depend on the specific segmentation problem and the context in which it is done. The segmentation problems that should be solved by such a tool are typically difficult, because the preceding algorithm already failed to segment the object appropriately. Nevertheless, and despite its importance in the segmentation process, research typically focuses on automatic or semi-automatic segmentation methods, while intuitive and efficient manual correction does not receive much attention. Table 2 summarizes the state-of-the-art in segmentation editing.

2.1. Incorporate user information

One of the first works on editing of automatic segmentation results was published by Neumann and Lorenz, who proposed to take user-defined constraints into account when fitting a statistical shape model to the image [NL98]. This basic approach was later used by several other authors in different ways for segmentations given by statistical shape models or similar model-based methods, like active shape and mass-spring models [vGdBLV03], [TPvB*03], [KSH*04], [SHT*08], [MFT*08], [LLO*10], [RDDP10], [SPA12], [SSB13]. Using such algorithms, the model is typically fit to the image again after each correction step. Representatives of methods that are automated versions of interactive algorithms, which explicitly focus on editing as well, are based on the interactive watershed transform, graph-cuts and belief propagation. These methods have been discussed by Hahn *et al*., Egger *et al*. and Steger and Sakas, respectively [HWKVP06], [ECFN12], [SS12a], [SS12b].

2.2. Low-level manual correction

The first work that mentions segmentation editing in the context of medical imaging was published by Elliott *et al*., where the authors mention a tool that allows editing contours in 2D [EKS92]. The first work that focuses on general purpose 3D segmentation editing, has been published by Kang *et al*. [KEK04]. The authors propose some basic low-level correction tools: hole-filling, point-bridging and surface-dragging. Later, several other authors have proposed dedicated low-level segmentation editing tools. Heimann *et al*. suggested a contour-based 2D correction tool, which was later extended by Maleike *et al*. [HTKM04], [MNMW09]. Maleike *et al*. also propose some morphological operations that allow filling holes and erasing islands [MNMW09]. Another low-level segmentation editing approach has been suggested by Yushkevich *et al*., who have implemented a cut-plane tool that allows removing parts of the segmentation that leaked into adjacent structures [YPH*06]. A similar tool called 'virtual knife' was proposed by Beck and Aurich [BA07]. A low-level approach by Silva *et al*. allows adding and removing voxels in 3D by a spherical brush [SSMS10]. Low-level manual correction is also often performed using deformable models that are modified by dragging or bulging [BMS*01], [BBR*03], [JSG03], [KEK04], [BRB*04], [JG05], [BBS06], [RBBS06], [BBB*07], [VGNN08], [SSMS10]. Such approaches can lead to unintended topology changes and degenerated surfaces, though, as shown by Silva *et al*. [SSMS10].

2.3. High-level manual correction

For general to use high-level manual correction, only a few solutions have been proposed so far. Kunert *et al*. as well as Heimann *et al*. have developed approaches that prevent region-growing-based algorithms from leaking into structures that the user explicitly marks by clicking into them [KSH*04], [HTKM04]. An image-based approach for general manual correction using the random walker algorithm has been suggested by Grady and Funka-Lea as well as El-Zehiry *et al*. [GFL06], [EZJS13]. A similar approach based on the image foresting transform has been developed by Miranda *et al*. [MFR10], [MaRC11]. For segmentation editing using deformable models, Proksch *et al*. have proposed two high-level tools that use sketching [PDP10]. In contrast to the dragging and bulging tools, this requires less knowledge about the underlying model. Another high-level method in the context of deformable models has been suggested by Ijiri and Yokota [IY10]. The tool described by the authors is based on an intuitive sketching interface and it is able to use all user-drawn contours.

2.4. Human-computer interaction

Some of the above-mentioned authors have mainly focused on human-computer interaction for interactive segmentation. The system developed by Bornik *et al*., Reitinger *et al*. as well as Beichel *et al*. uses an augmented reality approach where the user wears a shutter glass and interacts with a 3D scene using a dedicated input device that is tracked in 3D space [BBR*03], [BRB*04], [BBS06],

Table 2: *Overview on the state-of-the-art in segmentation editing in medical imaging (IUI, incorporate user information; LMC, low-level manual correction; HMC, high-level manual correction). HCI indicates that the paper focuses on human--computer interfaces/interaction. Publications referring to the same methods are assembled in one row.*

	IUI	LMC	HMC	HCI	Dim.	Method	Application	Evaluation
Elliott *et al.*, 1992 [EKS92]		•			2D	Contour correction	Brain, eye, optic nerve, tumours (CT, MRI)	-
Neumann and Lorenz, [NL98]	•				2D	Statistical shape model	Vertebra (CT)	-
Beichel *et al.*, 2001 [BMS*01]		•			3D	Statistical shape model	Diaphragm (CT)	-
Bornik *et al.*, 2003, 2004, 2006 [BBR*03], [BRB*04], [BBS06], Reitinger *et al.*, 2006 [RBBS06], Beichel *et al.*, 2007 [BBB*07]		•		•	3D	Deformable model	Liver, tumours, vessels, spleen, kidney, lung (CT)	Comparison to manual segmentations, correction time; up to 10 datasets (liver), 1 dataset (other objects); up to 18 participants
Ginneken *et al.*, 2003 [vGdBLV03]	•				2D	Statistical shape model	Lung, hand (Radiography); thrombus (CTA)	Comparison to manual segmentations; 115 (lung), 50 (hand), 11 patients/560 slices (thrombus)
Jackowski *et al.*, 2003, 2005 [JSG03], [JG05]		•			3D	Deformable model	Brain tumour (MRI), left ventricle (MRI), liver (CT)	1 (tumour), 1 (left ventricle), 1 (liver)
Timinger *et al.*, 2003 [TPvB*03]	•				3D	Statistical shape model	Vertebra, femur-head (CT)	Comparison to manual segmentations; 1 dataset per object
Kang *et al.*, 2004 [KEK04]		•			3D	Morphological, deformable model	Femur (CT)	1 dataset; 2 participants
Kunert *et al.*, 2004 [KSH*04] Heimann *et al.*, 2004 [HTKM04] Maleike *et al.*, 2009 [MNMW09]	•	•	•		2D, 3D	Active shape model, region growing, contour correction, morphological	Liver, tumours, vessels (CT)	up to 5 datasets; up to 12 participants
Grady and Funka-Lea, 2006 [GFL06]			•		3D	Random walker	Aortic aneurysm, bone, lung tumour, liver tumour, left ventricle (CT), brain tumour (MRI), left ventricle (US)	Correction time; 1 dataset per object
Yushkevich *et al.*, 2006 [YPH*06]		•			3D	Cut-plane	Caudate nucleus (MRI)	Comparison to manual segmentations; 15 datasets; 3 participants
Beck and Aurich, [BA07]		•			3D	Cut-plane	Liver (CT)	Comparison to manual segmentations, correction time; 20 datasets; 2 participants

(Continued)

Table 2: *Continued.*

	IUI	LMC	HMC	HCI	Dim.	Method	Application	Evaluation
Schwarz *et al.*, 2008 [SHT*08], Maleike *et al.*, 2008 [MFT*08]	•				3D	Statistical shape model	Left ventricle (MRI), liver, lymph nodes (CT)	Comparison to manual segmentations, correction time; 1 (left ventricle, liver), 29 (lymph nodes); 2 participants
Vidholm *et al.*, 2008 [VGNN08]		•		•	3D	Deformable model	Liver (CT)	Comparison to manual segmentations, correction time; 23 datasets; 2 participants
Heckel *et al.*, 2009 [HMB*09], [HMD*09]			•		3D	Contour correction, block matching, shortest path	Tumours, liver (CT)	Qualitative rating, correction time; 108 (tumours), 51 (liver); up to 6 participants
Ijiri and Yokota, [IY10]			•		3D	Contour correction, deformable model	Bone, kidney, muscles, bowel (CT)	Correction time; 1 (bone), 1 (bowel); 1 participant
Miranda *et al.*, 2010, 2011 [MFR10], [MaRC11]			•		3D	Image foresting transform	Brain, cerebellum, cerebral hemispheres (MRI)	Comparison to manual segmentations, number of correction steps; 10 datasets
Proksch *et al.*, 2010 [PDP10]		•	•	•	3D	Deformable model	Thyroid cartilage (CT)	-
Rahner *et al.*, 2010 [RDDP10]	•				3D	Mass-spring model	Thyroid cartilage, lymph nodes (CT)	Correction time; 3 (thyroid cartilage), 1 (lymph node); 4 participants
Silva *et al.*, 2010 [SSMS10]		•			3D	Paint brush, deformable model	Left ventricle (CT)	Correction time; 2 datasets; 3 participants
Egger *et al.*, 2012 [ECFN12]	•				3D	Graph-cut	Brain tumours (MRI)	Comparison to manual segmentations, correction time; 12 (brain tumours)
Shepherd *et al.*, 2012 [SPA12]	•				3D	Statistical shape model	Liver tumours (CT), MS lesions (MRI)	Comparison to manual/reference segmentations, number of correction steps, inter-observer variability; 3 synthetic liver tumours, 3 synthetic + 3 real MS lesions; 10 participants[a]
Steger and Sakas, [SS12a], [SS12b]	•				3D	Belief propagation	Tumours (CT, MRI), lymph nodes (CT)	Comparison to manual segmentations, correction time; 24 (tumours), 49 (lymph nodes)
Heckel *et al.*, 2012 [HBS*12]			•		2D, 3D	Contour correction, variational interpolation	Tumours (CT)	Qualitative rating, correction time; 89 (tumours); 2 participants
El-Zehiry *et al.*, 2013 [EZJS13]			•		3D	Random walker	Liver (CT, MRI), prostate (CT)	Comparison to reference segmentations; 30 (liver, MRI)
Sun *et al.*, 2013 [SSB13]	•			•	3D	Optimal surface fitting	Lung (CT)	Comparison to reference segmentations, correction time; 30 test cases

[a]Evaluation does not focus on editing.

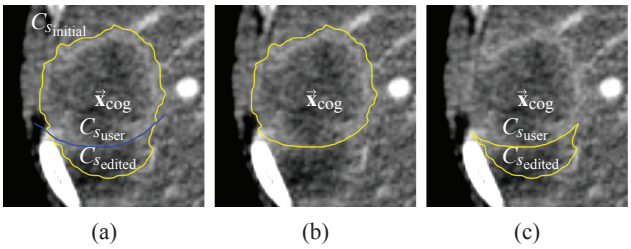


Figure 2: *Sketch-based 2D editing (liver metastasis in CT): (a) initial segmentation $C_{s_{\text{initial}}}$ (yellow), user input $C_{s_{\text{user}}}$ (blue) and the edited contour $C_{s_{\text{edited}}}$, (b) intended correction result containing the centre of gravity $\vec{x}_{\text{cog}}$ and (c) result not containing $\vec{x}_{\text{cog}}$, defining the edited region. [modified from HBS*12]*

[RBBS06], [BBB*07], [SSB13]. The system by Vidholm *et al*. is based on a haptic interface, where interaction is performed using a pencil-like 3D input device [VGNN08]. Such 3D interfaces are useful for educational purposes or exhibitions, for example, but they are not appropriate solutions for clinical routine. Finally, Proksch *et al*. have discussed various interaction techniques for tools based on deformable models [PDP10].

2.5. Related work and conclusion

Commonly used segmentation algorithms in the contour domain are live wire and active contours [BM97], [CV01]. Several authors have proposed 3D extensions of the live wire algorithm [FU00], [HYML05], [LH06], [PHA08]. Image-independent manual correction can be inspired by approaches developed in the field of 3D polygonal modelling, where 3D objects are generated from user-drawn 2D strokes. Such tools have been proposed by Igarashi *et al*., Karpenko *et al*. and Nealen *et al*., for example [IMT99], [KHR02], [NSACO05]. An example for such an approach for editing of medical image segmentations is the deformable surface editing proposed by Ijiri and Yokota [IY10]. A well-known method for surface reconstruction from unorganized point clouds is called variational interpolation. It has first been described by Turk and O'Brien [TO99]. We have discussed its application to contour-based interactive segmentation previously [HKHP11].

In conclusion, only a few authors have focused on dedicated, intuitive, general to use high-level tools for editing of 3D medical image segmentations. In addition, most of the proposed segmentation editing solutions are not well evaluated making it hard to assess their value for clinical practice.

3. Sketch-Based Editing in 2D

In 2D, sketch-based interaction provides a rather simple yet precise and intuitive manual correction in the contour domain. An insufficiently segmented slice s is corrected by drawing a contour along the correct object border (see Figure 2a). Based on this *user-drawn correction contour* $C_{s_{\text{user}}}$, a specific part of the contour of the initial segmentation $C_{s_{\text{initial}}}$ is replaced (see Figure 2b). This can be interpreted as adding some part, cutting away some part, combinations of both or replacing the segmentation as shown in Figure 3. Note

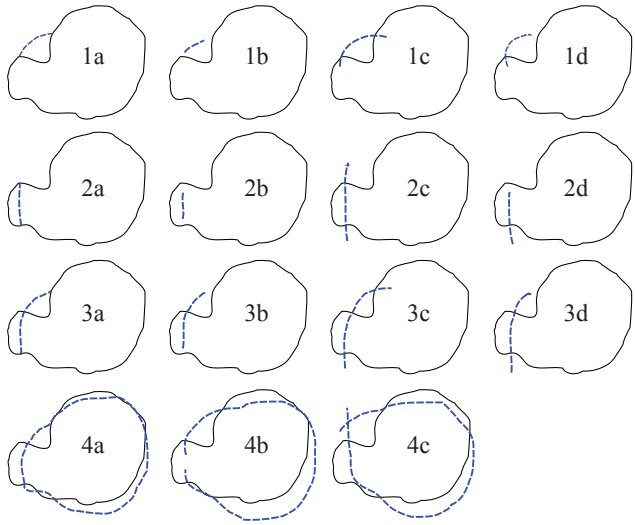


Figure 3: *Common user inputs (blue dashed lines) for 2D sketch-based corrections in the contour domain: (1a–d) add, (2a–d) remove, (3a–d) add + remove and (4a–c) replace. [from HBS*12]*

that this does not have to be differentiated in the contour domain. The contours $C_{s_{\text{initial}}}$ can be efficiently derived from the segmentation mask using a marching squares algorithm.

The user input will be subject to imperfection, such that $C_{s_{\text{user}}}$ might not be connected to the initial segmentation or it might intersect it in an arbitrary number of points (see Figure 3). For robustly handling those cases we clip $C_{s_{\text{user}}}$ to the first and the last intersection point. If $C_{s_{\text{user}}}$ does not intersect the initial segmentation near the first and the last point of $C_{s_{\text{user}}}$, we project those points on $C_{s_{\text{initial}}}$, i.e. the points of $C_{s_{\text{initial}}}$ that are closest to the start- and endpoints are used to extend $C_{s_{\text{user}}}$. If more than one initial contour exists in slice s, we use the one that is closest to the first point of $C_{s_{\text{user}}}$. Finally, the user has the possibility to replace the initial segmentation by drawing a self-intersecting or closed contour. A contour drawn in a slice that has not yet been segmented is implicitly closed by connecting its first and its last point.

The segmentation given by an open contour is ambiguous. There are two possible results for an initial contour $C_{s_{\text{initial}}}$ and the correction contour $C_{s_{\text{user}}}$ (see Figure 3). We use the following heuristics that decides what is most likely to be the user's intended result, where $\vec{x}_{\text{cog}}$ is the centre of gravity of $C_{s_{\text{initial}}}$:

(1) Keep the contour containing $\vec{x}_{\text{cog}}$.
(2) If both or none of the candidates contain $\vec{x}_{\text{cog}}$ keep the contour with the largest area.

If a specific point is known to be inside of the object that should be segmented, it can be used instead of $\vec{x}_{\text{cog}}$. The result that is not kept exactly encloses the region where the segmentation has been edited, i.e. the added or removed part in s (see Figure 2c). We call the *edited part* of the initial segmentation $C_{s_{\text{edited}}}$, while the whole *edited region* is given by $C_{s_{\text{user}}} \cup C_{s_{\text{edited}}}$.

The result might contain new holes. Holes can be found by checking for each contour if it is contained in another contour using a point-in-polygon test for each contour point. This needs to be done recursively defining a *level of embedding*. A hole is given by a contour whose level of embedding is odd. New holes are defined by contours whose level of embedding has changed. These contours are removed.

4. Sketch-Based Editing in 3D

To transform the 2D correction given by the user into 3D, we first estimate the extent of the edited region in z-direction based on its geometrical properties in the edited slice s. We call this the *correction depth* $d(C_{s_{\text{user}}})$. It represents a trade-off between the necessary number of correction steps and the risk of replacing too much of the initial segmentation. Our evaluations have shown that, for tumour segmentations, the z-extent of the edited region often corresponds to its thickness in the current slice. Therefore, we compute $d(C_{s_{\text{user}}})$ by

$$d(C_{s_{\text{user}}}) = \left\lceil \frac{\max_i \left\{ \min_j \left\{ \left| C_{s_{\text{edited}}}[i] - C_{s_{\text{user}}}[j] \right| \right\} \right\}}{d_s} \right\rceil, \tag{1}$$

where $C_{s_{\text{edited}}}[i]$ and $C_{s_{\text{user}}}[j]$ refer to the ith point of $C_{s_{\text{edited}}}$ and the jth point of $C_{s_{\text{user}}}$, respectively. d_s is the distance between adjacent slices in millimetres. If the initial segmentation is replaced, we cannot use it for computing the correction depth. Instead, we assume the shape of the object to be roughly spherical and the correction depth $\widehat{d}(C_{s_{\text{user}}})$ is computed as the radius of a circle that has the same area $A(C_{s_{\text{user}}})$ as $C_{s_{\text{user}}}$, i.e.

$$\widehat{d}(C_{s_{\text{user}}}) = \left\lceil \sqrt{\frac{A(C_{s_{\text{user}}})}{\pi}} \right\rceil. \tag{2}$$

The correction depth defines the first and the last slice s_{start} and s_{end} in which the segmentation is edited. In order not to interpolate beyond the initial segmentation, s_{start} and s_{end} are clamped to the minimum and maximum slice $s_{\min}$ and $s_{\max}$ of the initial segmentation, i.e. they are given by

$$s_{\text{start}} = \max\left(s - d(C_{s_{\text{user}}}), s_{\min}\right), \tag{3}$$

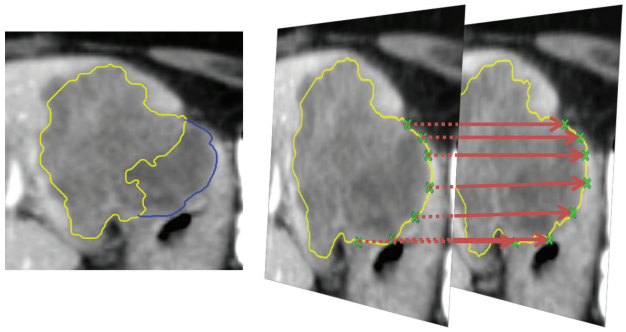
$$s_{\text{end}} = \min\left(s + d(C_{s_{\text{user}}}), s_{\max}\right). \tag{4}$$

4.1. Image-based extrapolation

In order to simulate the user input on the next (previous) slice $s \pm 1$, $C_{s_{\text{user}}}$ is sampled into a set of reference points $\vec{p}_{s_i}$. The reference points are placed on voxels that have high structural information, so they can be found more accurately on $s \pm 1$. In addition, we enforce a minimum distance d_P between two reference points. We use the discrete 2D structure tensor,

$$T(\vec{p}) = \sum_r w_r \begin{bmatrix} \mathcal{I}_x(\vec{p}-\vec{r})^2 & \mathcal{I}_x(\vec{p}-\vec{r})\mathcal{I}_y(\vec{p}-\vec{r}) \\ \mathcal{I}_x(\vec{p}-\vec{r})\mathcal{I}_y(\vec{p}-\vec{r}) & \mathcal{I}_y(\vec{p}-\vec{r})^2 \end{bmatrix}, \tag{5}$$

to measure the image information in a range $\vec{r}$ around $\vec{p}_{s_i}$. $\mathcal{I}_x$ and $\mathcal{I}_y$ are the partial derivatives of the image $\mathcal{I}$. The larger the tensor's smallest eigenvalue is, the higher is the structural information of the image at $\vec{p}_{s_i}$. The reference points are moved to $s \pm 1$ using a block-matching algorithm. It locates the best matching position for a reference block $\mathcal{R}_{s_i}$ in a search area $\mathcal{S}_{s\pm 1_i}$ around the perpendicular projection of $\vec{p}_{s_i}$ on $s \pm 1$. The median of squared differences is used as similarity measure. If several points have the same result, the one closest to $\vec{p}_{s_i}$ is used.



(a) (b)

Figure 4: *Image-based extrapolation (liver metastasis in CT): (a) initial segmentation (yellow), user input $C_{s_{\text{user}}}$ (blue) and (b) extrapolation of $C_{s_{\text{user}}}$ to the next slice with the reference points (green crosses) and the editing result (yellow).*

In the next step, adjacent reference points are connected with a shortest path method similar to the live wire algorithm [BM97] (see Figure 4b). Our method also includes a variation of the on-the-fly training by utilizing the information given by $C_{s_{\text{user}}}$. The user has drawn the correction contour along the actual border of the lesion, which is characterized by a specific gradient magnitude. We determine the average gradient magnitude $\widehat{G}$ of the gradient magnitudes G of all voxels $\vec{p}$ along $C_{s_{\text{user}}}$, from which we compute what we call the *preferred gradient magnitude feature* $\widehat{f}_G(\vec{p})$:

$$\widehat{f}_G(\vec{p}) = \frac{\left| G(\vec{p}) - \widehat{G} \right|}{\max(\widehat{G}, G_{\max} - \widehat{G})}. \tag{6}$$

$\widehat{f}_G(\vec{p})$ assigns costs of 0 to the known optimal gradient $\widehat{G}$ and costs of 1 to the gradient with the maximum difference to $\widehat{G}$. Note that $G(\vec{p}) \geq 0$ holds $\forall \vec{p}$. If the user has corrected errors at several different structures in the image at once, the average gradient magnitude does not represent the image information for any of the structures, though, and the basic gradient magnitude feature $f_G(\vec{p})$ proposed by Barrett and Mortensen gives better results. In order to robustly detect such cases, we compute the quartile variation of all gradient magnitudes of all voxels along $C_{s_{\text{user}}}$:

$$v_r = \frac{Q_{0.75} - Q_{0.25}}{Q_{0.5}}, \tag{7}$$

where $Q_{0.25}$, $Q_{0.5}$ and $Q_{0.75}$ are the 25%, 50% and 75% quantiles. If v_r is below 1, $\widehat{G}$ is considered to be a good representation of the objects border and the costs for two neighbouring voxels $\vec{p}$ and $\vec{q}$ are computed by

$$c(\vec{p},\vec{q}) = w_G \widehat{f}_G(\vec{q}) + w_Z f_Z(\vec{q}) + w_D f_D(\vec{p},\vec{q}) + w_L l(\vec{p},\vec{q}), \tag{8}$$

with $f_Z(\vec{p})$ being the Laplacian zero-crossing feature, $f_D(\vec{p})$ being the gradient direction feature and $l(\vec{p},\vec{q})$ being the length of the

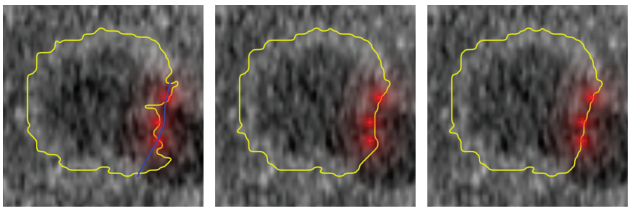


Figure 5: *Influence of the user consistency feature f_U for the example shown in Figure 2: (a) segmentation result (yellow) after three corrections (red) in another view and fourth correction (blue) in s, (b) result in $s+1$ without as well as (c) with f_U. The intensity of the red overlay encodes f_U.*

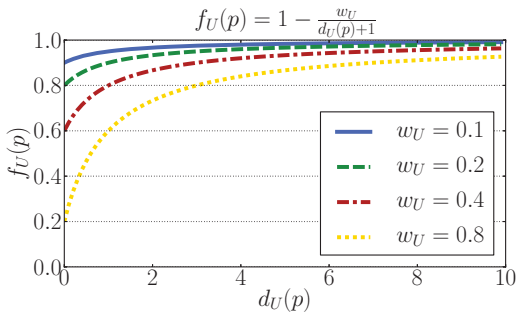


Figure 6: *Influence of w_U on the user consistency feature $f_U(\vec{p})$ (cp. Equation 8).*

path between $\vec{p}$ and $\vec{q}$, which prevents too long paths. Each feature is weighted by w_G, w_Z, w_D and w_L, respectively. If $v_r \geq 1$, $\widehat{f_G}(\vec{p})$ is replaced by $f_G(\vec{p})$. For details about the computation of $f_G(\vec{p})$, $f_Z(\vec{p})$ and $f_D(\vec{p})$, the interested reader is referred to the live wire algorithm [BM97].

This *extrapolated contour* is used for correcting the segmentation on $s \pm 1$ as described in Section 3. A manual correction might separate an initially coherent segmentation into several parts. Our algorithm aims at segmenting individual compact objects, though. Therefore, we perform a 3D connected component analysis and only keep the component that contains $\vec{x}_{\text{cog}}$ (*cp.* Section 3). If $\vec{x}_{\text{cog}}$ is outside of the resulting segmentation, we keep the component that is closest to it. The final step is a smoothing of the segmentation result using a $3 \times 3 \times 5$ Gaussian kernel followed by a thresholding.

4.1.1. *Stopping criteria and consistency checks*

The maximum extrapolation depth is given by s_{start} and s_{end}. The extrapolation might stop earlier if specific stopping criteria are fulfilled, though. These can be subdivided into two categories: Stopping due to information given by the user in previous steps, and heuristics that detect errors and inconsistencies of the extrapolation step.

The extrapolation stops at slices in which a closed contour has been drawn, assuming that they define the ground truth in the specific slice, which should not be changed by following editing steps. The heuristics test if the extrapolated contour is too small or if it has moved too far away from the initial segmentation. The extrapolation also stops if the relative size of the segmentation changes too much between two slices. Finally, a consistency check has been implemented that computes for each reference point whether it is inside or outside of the initial segmentation. A part of a correction contour that adds (removes) something to (from) the initial segmentation should not remove (add) something after extrapolation. If such a situation is detected, the reference point is removed. If adjacent reference points are both outside (inside), the path computed between them should as well be outside (inside). If this is not fulfilled, the extrapolation stops.

4.1.2. *Considering previous inputs*

Typically, the user performs several correction steps in varying views. The user-drawn contours from previous steps can be used to steer the extrapolation (see Figure 5). Ideally, each user-drawn contour should be part of the final segmentation result. In order to consider previous contours, the similarity measure of the block matching and the cost function of the shortest path are adapted by a term that depends on the shortest distance to any user-drawn contour. We call this the *user consistency feature* $f_U(\vec{p}) \in]0, 1]$, which is computed by

$$f_U(\vec{p}) = \begin{cases} 1 - \frac{w_U}{d_U(\vec{p})+1}, & \text{if } d_U(\vec{p}) \leq d_{U_{\max}} \\ 1, & \text{else} \end{cases}, \tag{9}$$

with $d_U(\vec{p})$ being the minimum distance of a voxel $\vec{p}$ to any user-drawn contour and $d_{U_{\max}}$ being the maximum influence range. $w_U \in [0, 1[$ controls the strength of the user consistency feature as shown in Figure 6. $f_U(\vec{p})$ is multiplied to the similarity measure result as well as the costs for the path during the shortest path search. It can be interpreted as a reward for paths passing near or going through previous contours making them 'soft constraints'. We have defined $f_U(\vec{p})$ such that it gives a much higher reward if a path passes exactly through a voxel on which another user-contour exists (i.e. if $d_U(\vec{p}) = 0$). Considering previous correction contours is particularly useful if the contrast or the signal-to-noise ratio are low, so the image itself does not provide adequate information for the extrapolation as shown in Figure 5.

4.2. Image-independent extrapolation

For cases with a very low contrast or signal-to-noise ratio, the image might not provide enough information for segmentation, so we have developed a purely geometrical approach as alternative for the image-based method. In order to generate a new segmentation in 3D that fits into the initial segmentation, we use the contours from the initial segmentation in s_{start} and s_{end}. Based on these contours as well as the editing result in s, a new segmentation is reconstructed using a variational-interpolation-based segmentation algorithm.

In variational interpolation methods, an object is represented by an implicit function $f(\vec{x})$ that evaluates to zero for each point on the surface of the object. It is computed based on a set of constraints given by the contour points (surface constraints) as well as the

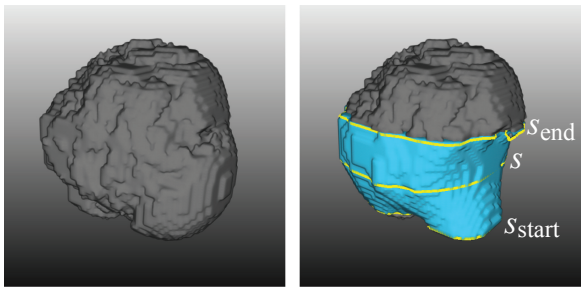


Figure 7: *Image-independent extrapolation of the correction shown in Figure 2: (a) the initial segmentation in 3D (grey) and (b) result after replacing the segmentation between s_{start} and s_{end} with the variational interpolation result (blue). [modified from HBS* 12]*

local normals in each contour point (normal constraints). Using an appropriate radial basis function (RBF) $\phi(\vec{x})$, $f(\vec{x})$ can be written as

$$f(\vec{x}) = P(\vec{x}) + \sum_{j=1}^{k} w_j \phi(\vec{x} - \vec{c}_j), \tag{10}$$

where $\vec{c}_j$ denote the constraints weighted by w_j and $P(\vec{x})$ is a degree-one polynomial that accounts for the linear and constant portions of $f(\vec{x})$. With $\phi(\vec{x}) = \|\vec{x}\|^3$, Equation 9 generates a smooth C^2-continuous surface. A quality-preserving constraint reduction can be used to speed up computations so interactive rates are achieved. This reduction is controlled by a quality factor q, which defines the portion of contour points that are used for the reconstruction. For details about the algorithm the interested reader is referred to our previous paper [HKHP11]. The mask in a slice can contain holes, however, which is not covered by the previously proposed algorithm. Holes can be handled by inverting all normal constraints for $f(\vec{x})$ given by all contours with an odd level of embedding (*cp.* Section 3).

The segmentation result given by the variational interpolation replaces the initial segmentation between s_{start} and s_{end}. Because s_{start} and s_{end} represent the initial segmentation and because of the properties of the variational interpolation, the new segmentation continuously fits into the initial segmentation (see Figure 7b). The final step of our algorithm is again the connected component analysis that removes separated parts of the segmentation (*cp.* Section 4.1).

4.2.1. *Considering previous and contradictory inputs*

By including all user-drawn contours into the reconstruction, it is guaranteed by the variational interpolation algorithm, that all contours are part of the segmentation result's surface. This allows the user to arbitrarily switch between different views during the correction process without replacing already performed corrections.

However, sometimes the user gives contradictory information, particularly if the correction is done in different views (see Figure 8). In order to resolve such contradictions, we search for all constraints from previous inputs in a specific range around each

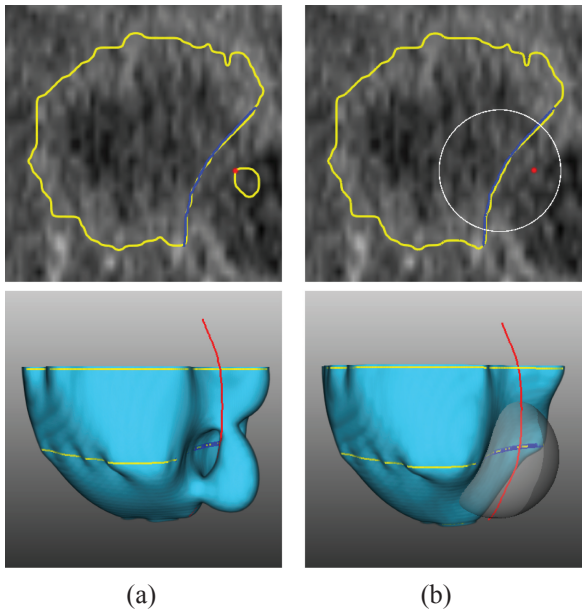


Figure 8: *Contradictory corrections example: (a) result (yellow) after the second correction (blue) without and (b) with resolving of contradictions. The first correction is indicated in red. The bottom row shows the variational interpolation result. The circle/sphere indicates the search range $s(C_{s_{\text{user}}})$.*

surface constraint given by a correction contour. This is done iteratively from the newest to the oldest user input. We use a specific portion of the correction depth as search range $s(C_{s_{\text{user}}})$, which is computed by

$$s(C_{s_{\text{user}}}) = \sigma d(C_{s_{\text{user}}}). \tag{11}$$

As corrections in other views are the main reason for contradictions, we ignore all constraints within $s(C_{s_{\text{user}}})$ during the reconstruction that have been given by correction contours from views that differ from the view of the currently inspected contour. $\sigma = 0.75$ has shown to be a good trade-off between ignoring too much information from previous contours and resolving unintended contradictions.

4.3. Appending, replacing, segmentation from scratch

Sometimes parts above or below the object are missing in the segmentation so the user needs to add something by drawing a closed contour in a slice that has not yet been segmented. In this case, we do not extrapolate beyond the edited slice so the extrapolation is done between s and s_{min} or s_{max}. If a segmentation in a slice is completely erroneous the user typically wants to replace it by drawing a closed contour. The correction depth is computed by Equation (2) as described in Section 4. The initial segmentation is replaced by $C_{s_{\text{user}}}$ in s and in all slices to which $C_{s_{\text{user}}}$ has been extrapolated. This replacement approach can also be used for segmentation from scratch, i.e. if no initial segmentation is available.

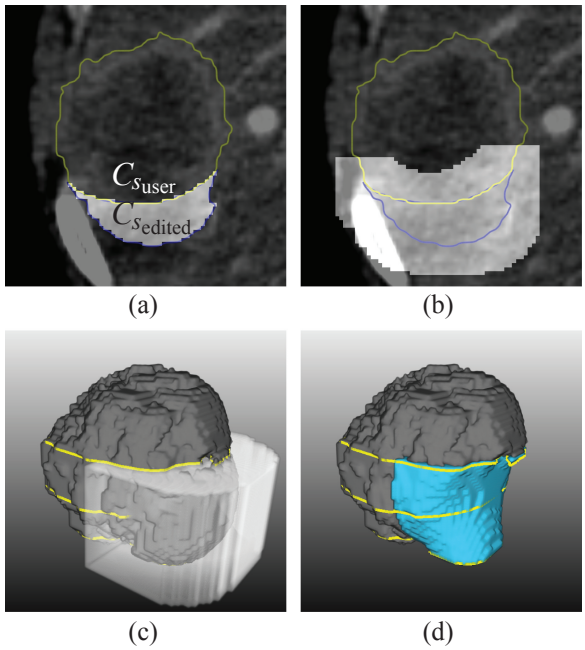


Figure 9: *Masking step on the example of the image-independent extrapolation: (a) voxelization of the edited region $C_{s_{\text{edited}}} \cup C_{s_{\text{user}}}$, (b) dilation of the edited region with a 15×15 kernel (cp. Equation 11), (c) duplication to all slices between s_{start} and s_{end} and (d) final result after masking. [modified from HBS*12]*

4.4. Making the correction local

Replacing the whole segmentation between s_{start} and s_{end} also changes parts of a segmentation that are actually correct. We need to restrict the correction to a region, where the user intends a correction. Based on the edited region $C_{s_{\text{user}}} \cup C_{s_{\text{edited}}}$, we define a region $\mathcal{I}_e \subset \mathcal{I}$ where the extrapolated result is used. For all voxels in $\mathcal{I} \setminus \mathcal{I}_e$ the original segmentation is kept (see Figure 9). We call this step *masking*. It is only applied if $C_{s_{\text{user}}}$ is an open contour. $\mathcal{I}_e$ is derived from the edited region as follows:

(1) Voxelization of $C_{s_{\text{user}}} \cup C_{s_{\text{edited}}}$.
(2) Dilation of resulting mask using an appropriate kernel.
(3) Duplication of dilated mask to slices in $[s_{\text{start}}, s_{\text{end}}]$.

The size k of the kernel for the dilation operation needs to be adapted to the size of the object, because the masking must cover a larger region for large objects and a smaller region for small objects. We made k proportional to the radius of a sphere, whose volume equals the volume of the initial segmentation given by the number of voxels n:

$$k = 2 \left\lceil \frac{1}{4} \sqrt[3]{\frac{3n}{4\pi}} \right\rceil + 1. \tag{12}$$

On our training data, this has provided a good coverage of the edited region while it efficiently prevented the correction from changing too much in unintended regions.

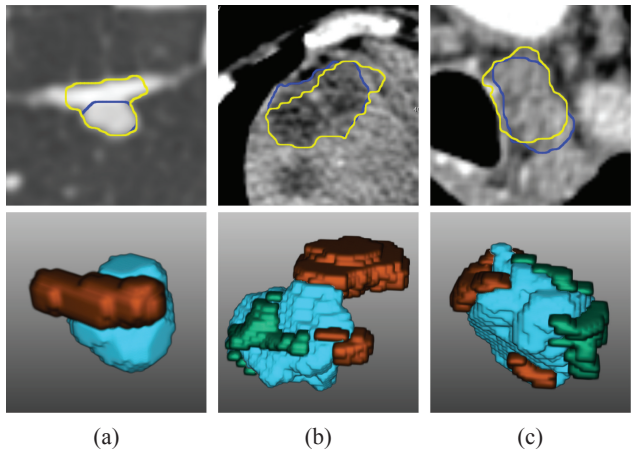


Figure 10: *Examples from our test database in 2D (top row) and 3D (bottom row): (a) a lung nodule, (b) a liver metastasis and (c) a lymph node. In 2D, the yellow and blue contour show the initial and reference segmentation, respectively. In 3D, the main parts that have to be removed from the initial segmentation are shown in orange, while the most relevant parts that need to be added are visualized in green.*

Table 3: *Average similarity to reference segmentations after the first editing step ($n = 1226$). Image-based (old) refers to the previously published algorithm [HMB*09].*

	Jaccard	Hausdorff
Initial	72.33%	9.31 mm
Image-based (old)	75.6%	8.19 mm
Image-based	78.1%	7.72 mm
Image-independent	75.72%	7.74 mm

5. Results

We have evaluated our algorithms on a database containing 1226 representative tumours in CT (liver metastases, lung nodules and lymph nodes) of different shape, size and image quality (see Figure 10). The data was collected from several studies of different clinics. It contained cases where the segmentation results of the algorithms by Moltz *et al.* [MBK*09] were edited using the previously published version of our image-based editing tool. The parameters of the image-based algorithm were set to $|\mathcal{R}_{s_i}| = 5 \times 5 \times 3$, $|\mathcal{S}_{s\pm 1_i}| = 11 \times 11 \times 3$, $d_P = 11\text{mm}$, $w_G = 0.5$, $w_Z = 0.1$, $w_D = 0.4$, $w_L = 0.003$ and $w_U = 0.1$. These values were optimized using our database of tumours. For estimating the quality of our algorithms, we applied the first correction and measured the similarity of the result to the reference segmentation generated by the clinicians. The image-based and the image-independent algorithm increased the average Jaccard index (volume overlap) by 8% and 4.7%, respectively (see Table 3). The Hausdorff distance (maximum surface distance) was decreased by 17.1% and 16.9% on average. Note that the reference segmentations are biased towards the image-based method, as a previous version of it was used for generating them. Subjectively, the results of the

Table 4: *Average similarity to reference segmentations without and with the user consistency feature f_U (image-based) and contradiction resolving (image-independent)* ($n = 66$).

	Jaccard	Hausdorff
Initial	52.85%	20.49 mm
Image-based (w/o f_U)	72.2%	14.49 mm
Image-based (w/ f_U)	72.38%	13.88 mm
Image-independent (w/o resolving)	66.57%	14.05 mm
Image-independent (w/ resolving)	67.4%	13.2 mm

image-independent method are comparable to the image-based algorithm. The extensions of the image-based algorithm proposed in this paper significantly improve the segmentation result of the first correction step with respect to the reference segmentations.

In order to assess the influence of the user consistency feature f_U on the image-based and of the contradiction handling on the image-independent algorithm, we have taken 66 cases from our database were the users have performed at least 10 corrections. We have applied all corrections both with and without f_U and the contradiction resolving step, respectively. In order to consider the size of the object, $d_{U_{\max}}$ was set to 0.2 times of the segmentation's bounding box extent. As shown in Table 4, the user consistency feature slightly improves the final segmentation result for the image-based algorithm. Likewise, the results of the image-independent method are improved by the resolving of contradictions. In this setting, only the first correction step is valid, however. If the results of the editing algorithm change, the following correction contours become suboptimal or even invalid, as they were drawn on the intermediate result at the time of the study. Hence, these results do not reflect the practical benefit of the user consistency and the contradiction handling.

Finally, our algorithms have been evaluated by six experienced radiologists (image-based) and two technical experts with at least 6 years of experience in tumour segmentation and assessment (image-independent). Both editing algorithms did not include the extensions described in this paper, though. The evaluations were performed on 89 tumours with an initially insufficient segmentation calculated by a dedicated automatic tool [BDK*07]. The data contained a representative set of tumours in CT (lung nodules, liver metastases, lymph nodes) of different shape, size and image quality. Correction was possible in axial, sagittal and coronal view and the participants were told to generate acceptable segmentations within a time that would be acceptable from their point of view. All participants got a short introduction to the tool and processed training examples. The quality of the editing tools in terms of their suitability for the specific segmentation task was rated on a 5-point scale (see Table 5). The participants needed a median number of four steps and a median time of 43 s (image-based) and 53 s (image-independent), including the time for assessment and rating. Note that an undo was counted as one step as well. The average computation time of one correction step is 0.6 s (image-based) and 0.3 s (image-independent, $q = 0.2$) on an Intel Xeon X5550. Overall, the image-based algorithm was well suited (rating of acceptable or better) in 91% of the segmentation tasks and the image-independent one for 92.1%, while about 9%/7.9% could not be corrected adequately within an acceptable time.

Table 5: *Quality ratings of our editing algorithms* ($n = 89$).

Rating	Image-based	Image-independent
Perfect (++)	19.1%	15.7%
Good (+)	47.2%	48.3%
Acceptable (0)	24.7%	28.1%
Bad (−)	5.6%	6.7%
Unacceptable (–)	3.4%	1.1%

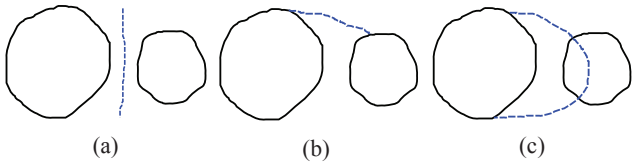


Figure 11: *Unsupported user inputs for multiple objects in a slice: (a) split, (b) merge and (c) 'partial replace'.*

6. Discussion

In our studies, most segmentation errors could be solved efficiently with only a few editing steps in a time that is much lower than a manual segmentation on each slice. However, for the image-based method, the surface of intermediate segmentation results sometimes suffers from 'unnatural' discontinuities caused by the stopping criteria and the fact that the geometry of the surface is not considered during extrapolation, which was criticized by the participants of our studies. In contrast, the image-independent method sometimes fails to recover sharp object features and high curvature due to its smooth nature. In addition, the users expected the image-independent tool to consider at least high contrast image features. Using local interpolation schemes for the image-independent reconstruction, algorithm could reduce the computational complexity but it would not help to solve the problem of changing the segmentation in unintended regions, as only a few contours from the initial segmentation in s, s_{start} and s_{end} are used for reconstruction.

If the interpretation of the user-drawn contour described in Section 3 fails, this editing step gives a result that does not match the user's intention. In the worst case, an editing step is virtually ignored. Other cases that cannot be corrected adequately with our methods are segmentations where the larger part of the contour does not belong to the object, because the wrong part is kept (*cp.* Section 3). This can be handled by drawing closed contours, though. As already mentioned in Section 3 an object might consist of more than one component in a slice. For such cases we have observed user interactions as shown in Figure 11 that are not supported by our algorithms. Filling holes is also not directly supported by sketching and must be done by replacing the segmentation in the affected slice or by editing in another view. Moreover, it is not possible to delete the segmentation in a slice completely. A solution could be to provide a specific button or key for this purpose on the application

level. In addition, this can most often be solved by editing in another view.

Contradictory inputs have shown to be a rather frequent issue. Our database contained several cases where something has been added to the segmentation, which has later been removed again, for example. Thus, previous user inputs cannot always be considered as 'hard constraints'. We know from discussions with our clinical partners that segmentation tools need to be as easy as possible and they need to integrate into the established clinical workflow. Otherwise, such tools are hardly accepted and will not be used in routine. Thus, we decided to not provide 3D views or the possibility for interactions in 3D. Although 3D renderings could have helped the user to better understand the segmentation problem and reduce the probability for contradictory inputs, it would have made the tools too complicated. Deformable 3D models have the advantage that they provide a 'template' shape, which is modified by the user, so contradictions are not a problem. However, if topology changes are necessary as shown in Figures 10(a) and (b), for example, this is more complicated and less intuitive with deformable models.

Having an efficient and intuitive editing tool that produces sufficient results with only a few steps in about 90% of the cases is very helpful in practice. Current state-of-the-art tumour segmentation algorithms provide sufficient results in 87% of the cases [MBK*09]. Combining this with our tools means that most lesions can be segmented with little work (about four correction steps). For the remaining cases a slice-wise correction can still be used if the segmentation is mandatory. This smoothly integrates into the contour-based workflow, e.g. by a modifier key, which prevents our tools from performing the extrapolation step. This was not supported during our evaluations, though. For the image-based method, we have already shown that it can be applied to liver segmentation as well [HMD*09]. Our tools are not suitable for objects that are not compact, like vessel trees, though, as they are not necessarily continuous over neighbouring slices. Comparing our methods to other segmentation editing tools is hard, because the results strongly depend on the specific user and on the segmentation task. To overcome this, we propose a segmentation editing challenge where different tools are applied to the same problems by the same users.

7. Conclusion

Segmentation editing is an indispensable step in the segmentation process. Efficient high-level manual correction in 3D is a challenging problem. Sketching has proven to provide an intuitive interface for segmentation editing in 2D. We have discussed both an image-based and an image-independent method for transforming those 2D user inputs into 3D. If the contrast or the signal-to-noise ratio are very low, the image often does not provide enough information for segmentation, so image-independent editing is better suited in such cases. Our editing tools can even be used in a slice-wise manner and if no initial segmentation is available. Because they only need the image, the segmentation mask and the user input for computation, they can be combined with any automatic segmentation algorithm and they can be used in any 3D modality, such as CT or magnetic resonance imaging (MRI). Moreover, our tools are applicable for a wide range of other objects as long as they are compact. In conclusion, the presented algorithms provide universal and intuitive editing tools for segmentation of compact objects in 3D medical images. An evaluation on segmentations of solid tumours in CT has proven that our algorithms allow an efficient editing of such segmentations.

8. Future Work

Future work will focus on further improving our algorithms in terms of computation time and segmentation quality so they can be applied to larger and more complex objects, like organs, while still providing their results at interactive rates. In particular, the image-independent method should benefit from a GPU implementation, allowing its application to large structures without the necessity of the constraint reduction. We are also thinking of ways to combine our algorithms in a hybrid approach. Alternatively, image information could be included into the variational interpolation method using image-adaptive RBFs proposed by Mory *et al*. [MAYT09].

For validating editing tools the user is mandatory. The results depend on the user's input and user inputs depend on previous results. If the editing algorithm is changed, the intermediate results and consequently following inputs change as well. Furthermore, it is insufficient to validate the final result only, as the acceptance of editing tools suffers from bad intermediate results. In order to allow a more convincing validation without the need for a user, we are currently investigating methods for plausibly simulating the user.

References

[BA07] Beck A., Aurich V.: A semiautomatic liver segmentation system. In *Proceedings of the 3D Segmentation in The Clinic: A Grand Challenge*, MICCAI Workshop (2007), pp. 225–234.

[BBB*07] Beichel R., Bauer C., Bornik A., Sorantin E., Bischof H.: Liver segmentation in CT data: A segmentation refinement approach. In *Proceedings of the 3D Segmentation in The Clinic: A Grand Challenge*, MICCAI Workshop (2007), pp. 235–245.

[BBR*03] Bornik A., Beichel R., Reitinger B., Gotschuli G., Sorantin E., Leberl F. W., Sonka M.: Computer-aided liver surgery planning: An augmented reality approach. In *Proceedings of the SPIE Medical Imaging: Visualization, Image-Guided Procedures, and Display* (2003), pp. 395–406.

[BBS06] Bornik A., Beichel R., Schmalstieg D., : Interactive editing of segmented volumetric datasets in a hybrid 2D/3D virtual environment. In *Proceedings of the ACM Symposium on Virtual Reality Software and Technology* (2006), pp. 197–206.

[BDK*07] Bornemann L., Dicken V., Kuhnigk J.-M., Wormanns D., Shin H.-O., Bauknecht H.-C., Diehl V., Fabel M., Meier S., Kress O., Krass S., Peitgen H.-O.: OncoTREAT: A software assistant for cancer therapy monitoring. *International Journal of Computer Assisted Radiology and Surgery 1*, *5* (2007), 231–242.

[BM97] Barrett W. A., Mortensen E. N.: Interactive live-wire boundary extraction. *Medical Image Analysis 1* (1997), 331–341.

[BMS*01] Beichel R., Mitchell S. C., Sorantin E., Leberl F., Goshtasby A. A., Sonka M.: Shape- and appearance-based segmentation of volumetric medical images. In *Proceedings of the*

International Conference on Image Processing 2 (2001), pp. 589–592.

[BRB*04] Bornik A., Reitinger B., Beichel R., Sorantin E., Werkgartner G.: Augmented-reality-based segmentation refinement. In *Proceedings of the SPIE Medical Imaging: Visualization, Image-Guided Procedures, and Display* (2004), pp. 77–87.

[CV01] Chan T. F., Vese L. A.: Active contours without edges. *IEEE Transactions on Image Processing 10*, *2* (Feb. 2001), 266–277.

[ECFN12] Egger J., Colen R. R., Freisleben B., Nimsky C.: Manual refinement system for graph-based segmentation results in the medical domain. *Journal of Medical Systems 36* (Aug. 2012), 2829–2839. Online First (2011).

[EKS92] Elliott P. J., Knapman J. M., Schlegel W.: Interactive image segmentation for radiation treatment planning. *IBM Systems Journal 31*, *4* (1992), 620 –634.

[EZJS13] El- Zehiry N., Jolly M.-P., Sofka M.: A splice-guided data driven interactive editing. In *Proceedings of the International Symposium on Biomedical Imaging: From Nano to Macro* (2013), pp. 1086–1089.

[FU00] Falc ao A. X., Udupa J. K.: A 3D generalization of user-steered live-wire segmentation. *Medical Image Analysis 4*, *4* (2000), 389–402.

[GFL06] Grady L., Funka-Lea G.: An energy minimization approach to the data driven editing of presegmented images/volumes. In *Proceedings of the International Conference on Medical Image Computing and Computer Assisted Intervention 2* (2006), pp. 888–895.

[HBS*12] Heckel F., Braunewell S., Soza G., Tietjen C., Hahn H. K.: Sketch-based image-independent editing of 3D tumor segmentations using variational interpolation. In *Proceedings of the Eurographics Workshop on Visual Computing for Biology and Medicine* (2012), pp. 73–80.

[HKHP11] 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* (2011), 275–287.

[HM09] Heimann T., Meinzer H.-P.: Statistical shape models for 3D medical image segmentation: A review. *Medical Image Analysis 13*, *4* (2009), 543–563.

[HMB*09] Heckel F., Moltz J. H., Bornemann L., Dicken V., Bauknecht H.-C., Fabel M., Hittinger M., Kiessling A., Meier S., Püsken M., Peitgen H.-H.: 3D contour based local manual correction of tumor segmentations in CT scans. In *Proceedings of the SPIE Medical Imaging: Image Processing 7259* (2009), pp. 72593L-1–9.

[HMD*09] Heckel F., Moltz J. H., Dicken V., Geisler B., Bauknecht H.-C., Fabel M., Meier S., Peitgen H.-O.: 3D contour based local manual correction of liver segmentations in CT scans. *Computer Assisted Radiology and Surgery 4* (2009), 45–46.

[HTKM04] Heimann T., Thorn M., Kunert T., Meinzer H.-P.: New methods for leak detection and contour correction in seeded region growing segmentation. *International Archives of Photogrammetry and Remote Sensing 35* (2004), 317–322.

[HWKVP06] Hahn H. K., Wenzel M. T., Konrad-Verse O., Peitgen H.-O.: A minimally-interactive watershed algorithm designed for efficient CTA bone removal. *Computer Vision Approaches to Medical Image Analysis 4241* (2006), 178–189.

[HYML05] Hamarneh G., Yang J., McIntosh C., Langille M.: 3D live-wire-based semi-automatic segmentation of medical images. *SPIE Medical Imaging: Image Processing 5747* (2005), 1597–1603.

[IMT99] Igarashi T., Matsuoka S., Tanaka H.: Teddy: A sketching interface for 3D freeform design. In *Proceedings of the ACM SIGGRAPH* (Los Angeles, CA, USA, 1999), ACM, pp. 409–416.

[IY10] Ijiri T., Yokota H.: Contour-based interface for refining volume segmentation. *Pacific Graphics 29* (2010), 2153–2160.

[JG05] Jackowski M., Goshtasby A.: A computer-aided design system for revision of segmentation errors. In *Proceedings of the International Conference on Medical Image Computing and Computer-assisted Intervention* (2005), pp. 717–724.

[JSG03] Jackowski M., Satter M., Goshtasby A.: Approximating digital 3D shapes by rational gaussian surfaces. *IEEE Transactions on Visualization and Computer Graphics 9*, *1* (2003), 56–69.

[KEK04] Kang Y., Engelke K., Kalender W. A.: Interactive 3D editing tools for image segmentation. *Medical Image Analysis 8*, *1* (2004), 35–46.

[KHR02] Karpenko O., Hughes J. F., Raskar R.: Free-form sketching with variational implicit surfaces. *Computer Graphics Forum 21*, *3* (2002), 585–594.

[KSH*04] Kunert T., Schroter A., Heimann T., Schobinger M., Bottger T., Thorn M., Wolf I., Engelmann U., Meinzer H.-H.: An interactive system for volume segmentation in computer-assisted surgery. *SPIE Medical Imaging: Visualization, Image-Guided Procedures, and Display 5367* (2004), 799–809.

[LH06] Lu K., Higgins W. E.: Improved 3D live-wire method with application to 3D CT chest image analysis. *SPIE Medical Imaging: Image Processing 6144* (2006), pp. 61440L-1–15.

[LLO*10] Li B., Liu Y., Occleshaw C. J., Cowan B. R., Young A. A.: In-line automated tracking for ventricular function with magnetic resonance imaging. *JACC: Cardiovascular Imaging 3*, *8* (2010), 860–866.

[MaRC11] Miranda P. A. V., Falcão A. X., Ruppert G. C. S., Cappabianco F. A. M.: How to fix 3D segmentation interactively via image foresting transform and its use in MRI brain segmentation. In *Proceedings of the IEEE International Symposium on Biomedical Imaging: From Nano to Macro* (2011), pp. 2031–2035.

[MAYT09] Mory B., Ardon R., Yezzi A., Thiran J.-P.: Non-Euclidean image-adaptive radial basis functions for 3D interactive segmentation. In *Proceedings of the IEEE International Conference on Computer Vision* (2009), pp. 787–794.

[MBK*09] Moltz J. H., Bornemann L., Kuhnigk J.-J., Dicken V., Peitgen E., Meier S., Bolte H., Fabel M., Bauknecht H.-H., Hittinger M., Kiessling A., Püsken M., Peitgen H.-H.: Advanced segmentation techniques for lung nodules, liver metastases, and enlarged lymph nodes in CT scans. *IEEE Journal of Selected Topics in Signal Processing 3*, *1* (2009), 122–134.

[MFR10] Miranda P. A. V., Falcao A. X., Ruppert G. C. S.: How to complete any segmentation process interactively via image foresting transform. In *Proceedings of the SIBGRAPI Conference on Graphics, Patterns and Images* (2010), pp. 309–316.

[MFT*08] Maleike D., Fabel M., Tetzlaff R., von Tengg-Kobligk H., Heimann T., Meinzer H.-H., Wolf I.: Lymph node segmentation on CT images by a shape model guided deformable surface method. In *Proceedings of the SPIE Medical Imaging: Image Processing 6914* (2008), pp. 69141S-1–8.

[MNMW09] Maleike D., Nolden M., Meinzer H.-H., Wolf I.: Interactive segmentation framework of the medical imaging interaction toolkit. *Computer Methods and Programs in Biomedicine 96*, *1* (2009), 72–83.

[NL98] Neumann A., Lorenz C.: Statistical shape model based segmentation of medical images. *Computerized Medical Imaging and Graphics 22*, *2* (1998), 133–143.

[NSACO05] Nealen A., Sorkine O., Alexa M., Cohen-Or D.: A sketch-based interface for detail-preserving mesh editing. In *Proceedings of the ACM SIGGRAPH Papers* (2005), pp. 1142–1147.

[OS01] Olabarriaga S. D., Smeulders A. W. M.: Interaction in the segmentation of medical images: A survey. *Medical Image Analysis 5* (2001), 127–142.

[PDP10] Proksch D., Dornheim J., Preim B.: Interaktionstechniken zur Korrektur medizinischer 3D-Segmentierungen. In *Bildverarbeitung für die Medizin* (2010), 420–424.

[PHA08] Poon M., Hamarneh G., Abugharbieh R.: Efficient interactive 3D livewire segmentation of objects with arbitrarily topologies. *Computerized Medical Imaging and Graphics 32*, *8* (2008), 639–650.

[RBBS06] Reitinger B., Bornik A., Beichel R., Schmalstieg D.: Liver surgery planning using virtual reality. *IEEE Computer Graphics and Applications 26*, *6* (2006), 36–47.

[RDDP10] Rahner S., Dornheim J., Dornheim L., Preim B.: Interaktive techniken zur korrektur medizinischer segmentierungen auf basis stabiler feder-masse-modelle. In *Jahrestagung der Deutschen Gesellschaft für Computer- und Roboterassistierte Chirurgie* (2010), pp. 159–164.

[SHT*08] Schwarz T., Heimann T., Tetzlaff R., Rau A.-A., Wolf I., Meinzer H.-H.: Interactive surface correction for 3D shape based segmentation. In *Proceedings of the SPIE Medical Imaging: Image Processing 6914* (2008), pp. 69143O-1–8.

[SPA12] Shepherd T., Prince S. J. D., Alexander D. C.: Interactive lesion segmentation with shape priors from offline and online learning. *IEEE Transactions on Medical Imaging 31*, *9* (2012), 1698–1712.

[SS12a] Steger S., Sakas G.: FIST: Fast interactive segmentation of tumors. In *Abdominal Imaging. Computational and Clinical Applications* (vol. 7029). H. Yoshida, G. Sakas, M. G. Linguraru (Eds.). Springer, Berlin, Heidelberg (2012), pp. 125–132.

[SS12b] Steger S., Sakas G.: Image gradient based shape prior for the segmentation of not that spherical structures. In *Proceedings of the IEEE International Symposium on Biomedical Imaging* (2012), pp. 1252–1255.

[SSB13] Sun S., Sonka M., Beichel R. R.: Lung segmentation refinement based on optimal surface finding utilizing a hybrid desktop/virtual reality user interface. *Computerized Medical Imaging and Graphics 37*, *1* (2013), 15–27. Online first 12 February, 2013.

[SSMS10] Silva S., Santos B. S., Madeira J., Silva A.: A 3D tool for left ventricle segmentation editing. In *Proceedings of the International Conference on Image Analysis and Recognition 6112* (2010), pp. 79–88.

[TO99] Turk G., O'Brien J. F.: Shape transformation using variational implicit functions. In *Proceedings of the ACM SIGGRAPH* (1999), pp. 335–342.

[TPvB*03] Timinger H., Pekar V., von Berg J., Dietmayer K., Kaus M.: Integration of interactive corrections to model-based segmentation algorithms. In *Bildverarbeitung für die Medizin 80* (2003), pp. 171–175.

[vGdBLV03] van Ginneken B., de Bruijne M., Loog M., Viergever M. A., : Interactive shape models. In *Proceedings of the SPIE Medical Imaging: Image Processing* (2003), pp. 1206–1216.

[VGNN08] Vidholm E., Golubovic M., Nilsson S., Nyström I.: Accurate and reproducible semi-automatic liver segmentation using haptic interaction. In *Proceedings of the SPIE Medical Imaging: Visualization, Image-guided Procedures, and Modeling* (2008), pp. 69182Q-1–8.

[YPH*06] Yushkevich P. A., Piven J., Hazlett H. C., Smith R. G., Ho S., Gee J. C., Gerig G.: User-guided 3D active contour segmentation of anatomical structures: Significantly improved efficiency and reliability. *NeuroImage 31*, *3* (2006), 1116–1128.