

3D contour based local manual correction of tumor segmentations in CT scans

Frank Heckel^a, Jan Hendrik Moltz^a, Lars Bornemann^a, Volker Dicken^a,
Hans-Christian Bauknecht^b, Michael Fabel^c, Markus Hittinger^d, Andreas Kießling^e,
Stephan Meier^f, Michael Püsken^g and Heinz-Otto Peitgen^a

^aFraunhofer MEVIS, Institute for Medical Image Computing, Bremen, Germany*;

^bCharité, Institute for Radiology, Berlin, Germany;

^cChristian-Albrechts-University, Department of Diagnostic Radiology, Kiel, Germany;

^dLudwig-Maximilians-University, Department for Clinical Radiology, Munich, Germany;

^ePhilipps-University, Department of Diagnostic Radiology, Marburg, Germany;

^fJohannes Gutenberg University, Clinic and Out-patients' Clinic for Diagnostic and
Interventional Radiology, Mainz, Germany;

^gUniversity of Münster, Institute for Clinical Radiology, Münster, Germany

ABSTRACT

Segmentation is an essential task in medical image analysis. For example measuring tumor growth in consecutive CT scans based on the volume of the tumor requires a good segmentation. Since manual segmentation takes too much time in clinical routine automatic segmentation algorithms are typically used. However there are always cases where an automatic segmentation fails to provide an acceptable segmentation for example due to low contrast, noise or structures of the same density lying close to the lesion. These erroneous segmentation masks need to be manually corrected.

We present a novel method for fast three-dimensional local manual correction of segmentation masks. The user needs to draw only one partial contour which describes the lesion's actual border. This two-dimensional interaction is then transferred into 3D using a live-wire based extrapolation of the contour that is given by the user in one slice. Seed points calculated from this contour are moved to adjacent slices by a block matching algorithm. The seed points are then connected by a live-wire algorithm which ensures a segmentation that passes along the border of the lesion. After this extrapolation a morphological postprocessing is performed to generate a coherent and smooth surface corresponding to the user drawn contour as well as to the initial segmentation. An evaluation on 108 lesions by six radiologists has shown that our method is both intuitive and fast. Using our method the radiologists were able to correct 96.3% of lesion segmentations rated as insufficient to acceptable ones in a median time of 44s.

Keywords: Segmentation, Interaction, Contour, Correction, Editing, Live-Wire, 3D, CT

1. INTRODUCTION

Segmentation is an essential task in medical image analysis. Since a manual segmentation takes too much time in clinical routine and lacks reproducibility automatic segmentation algorithms are used. However there are always cases where an automatic segmentation fails to provide an acceptable segmentation for example due to low contrast, noise or structures of the same density lying close to the object that should be segmented. Though most parts of the segmentation mask are usually correct and only small regions are erroneous. So the clinician needs the possibility to locally correct the masks by hand.

Further author information: (Send correspondence to F. Heckel.)

F. Heckel: E-mail: frank.heckel@mevis.fraunhofer.de, Telephone: +49 (0)421 218 9068

*formerly: MeVis Research GmbH

Typically the correction is done on each slice which is a time consuming task. For clinical acceptance the manual correction has to be both intuitive and fast which implies that modifications would ideally be three-dimensional. But literature on 3D manual correction is sparse which was already stated by Kang *et al.*¹ and which still holds true. Some authors address editing of segmentations by deformable 3D meshes² or surface models^{3,4} which neither use any image data nor are they intuitive to use. A promising approach is based on the random walker algorithm⁵ but the required user interaction is still not intuitive enough in our opinion.

In this paper we present a novel method for fast three-dimensional local manual correction of automatic segmentation results in the context of oncological therapy monitoring, where a good segmentation is required to reliably measure tumor growth in consecutive CT scans based on the volume of the tumor.⁶ In our approach the user needs to draw only one partial two-dimensional contour which describes the lesion's actual border. A part of the initial contour of the lesion is then replaced by the user defined one. One challenge is to extrapolate the 2D user interaction to the third dimension. Also the algorithm must not take more than a few seconds to allow an interactive correction. For addressing those requirements we use a *live-wire based extrapolation* followed by a *morphological postprocessing*. Our method is independent of the specific lesion type and the algorithm that is used for the initial segmentation.

The organization of this paper is as follows. Section 2 explains the algorithm in detail. In Sec. 3 the results of an evaluation of our algorithm are presented. Finally the algorithm and the results are discussed in Sec. 4 and our work is summarized in Sec. 5.

2. METHOD

Our manual correction algorithm is working on a region of interest (ROI) that completely contains the lesion. We assume that the lesion is located at the center of the ROI. The algorithm mainly consists of two steps. First a *live-wire based extrapolation* of the contour given by the user in one slice is performed to simulate the user interaction in adjacent slices. The second step is a *morphological postprocessing* which ensures that the resulting mask is coherent and its surface is smooth. User interaction is done in an intuitive manner. The user simply draws a partial two-dimensional contour C^u that defines the actual border of the lesion (see Fig. 4(a)). This can be done in any slice in axial, coronal or sagittal view. In detail the processing pipeline after user interaction is as follows:

1. Reformat the initial mask so it matches the view the user interacted with
2. Generate the contours C_i from slice i of the reformatted mask
3. Apply the *live-wire based extrapolation* to the user drawn contour (see Sec. 2.1)
4. Generate a mask from the modified contours C'_i
5. Perform the *morphological postprocessing* (see Sec. 2.2)
6. Reformat the corrected mask to the coordinate space of the initial mask

The contour generation is done per slice from the mask given by the initial segmentation using a *marching squares algorithm*. The contours are guaranteed to be oriented clockwise which is necessary in the live-wire extrapolation step. A mask can be generated from the contour by first voxelizing the contour itself using a derivate of the *marching cubes* scheme and then filling the area surrounded by these voxels using some *marching cubes* approach. In the following we will describe the main steps of our algorithm in detail.

2.1 Live-Wire Extrapolation

The *live-wire based extrapolation* is working on the contours generated by the marching squares algorithm. Extrapolation of C^u starts on the slice u the user interacted with. Because a slice is defined in the x - y -plane when looking at voxel coordinates, the adjacent slices are processed in order of their z -coordinate both in positive and negative z -direction.

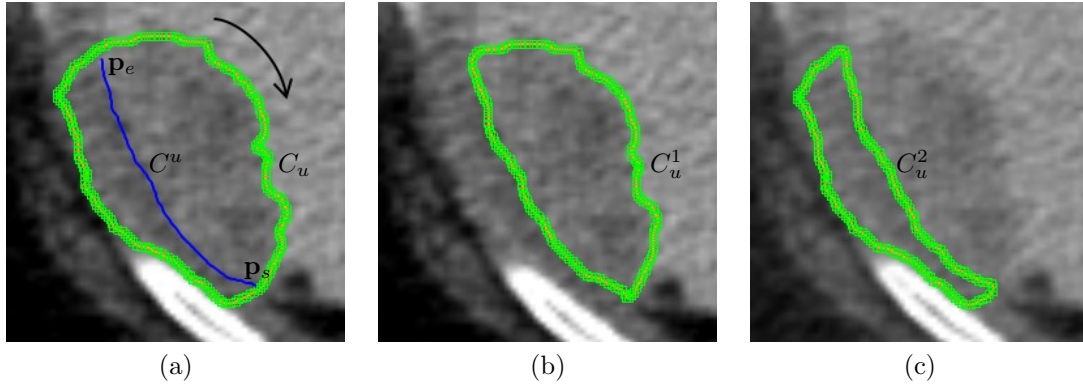


Figure 1. Two possible results for replacing a part of the initial, clockwise oriented contour: (a) the initial contour C_u (thick green/gray line) in slice u , the user drawn contour C^u (blue/dark gray line) and the start and end points \mathbf{p}_s and \mathbf{p}_e ; (b) the first and expected result C_u^1 and (c) the second possible result C_u^2

In the currently processed slice i some part of the initial 2D contour C_i is replaced by the user contour in slice i which we call C_i^u in the following. C_i^u is generated from the contour C_{i-1}^u in the previous slice when processing in positive z -direction. In negative z -direction C_{i+1}^u and in slice u C^u are used as “input” contours respectively. For readability reasons we will only refer to C_{i-1}^u in the following.

C_i^u is derived from C_{i-1}^u by calculating equidistant points on C_{i-1}^u . We refer to these points as *seed points*. In our case 15mm has shown to be a suitable distance between those points. The seed points are then moved to the current slice by a 2D block matching algorithm which we will describe in Sec. 2.1.1 (the block matching is not performed when processing slice u). After moving the points to the current slice they are connected by a *live-wire algorithm* that finds the best path between the seed points based on the intensity values of the image (see Fig. 4(b)). This algorithm is discussed in Sec. 2.1.2. Processing stops if the size of the 2D axis-aligned bounding box of all seed points is below a given threshold, if there is no contour in the current slice or if the user already corrected that slice directly by drawing a partial contour on it. The threshold for minimum size of the axis-aligned bounding box is calculated based on the extent \mathbf{e} of the initial segmentation in voxel coordinates where a threshold of $\frac{\min(\mathbf{e}_x, \mathbf{e}_y, \mathbf{e}_z)}{10}$ worked well in most cases.

After C_i^u has been calculated we need to determine which part of the initial contour should be replaced by the new one. If C^u is closed we simply can replace C_i by C_i^u to get the new contour C_i' . If C^u is not closed we calculate the points on C_i that are closest to the first (start) and the last (end) point on C_i^u (called \mathbf{p}_s and \mathbf{p}_e) and replace the part of C_i between those points by C_i^u . If C^u intersects C_i at exactly two points these intersections points are used instead of the first and last point as \mathbf{p}_s and \mathbf{p}_e . This allows cutting away some part of the segmentation in an intuitive manner. There are two possible results for replacing the part of C_i between \mathbf{p}_s and \mathbf{p}_e (see Fig. 1). The decision which part of the contour is kept is based on the following rules. We call the two possible results C_i^1 (where the part of C_u between \mathbf{p}_s and \mathbf{p}_e is replaced) and C_i^2 (where the part between \mathbf{p}_e and \mathbf{p}_s is replaced).

1. Keep the contour that contains the center \mathbf{c}_s of the current slice.
2. If both C_i^1 and C_i^2 or neither C_i^1 nor C_i^2 contain \mathbf{c}_s :
 - (a) Keep the contour that contains the center of gravity \mathbf{c}_g of the initial mask in the current slice.
 - (b) If both C_i^1 and C_i^2 or neither C_i^1 nor C_i^2 contain \mathbf{c}_g keep the longer contour.

These rules are evaluated on slice u and the decision made on that slice (i.e. whether to remove the part between \mathbf{p}_s and \mathbf{p}_e or vice versa) is applied to the other slices as well. This results in keeping the correct part of the initial contour on each slice, because all contours are oriented in the same direction which is guaranteed by the marching squares algorithm.

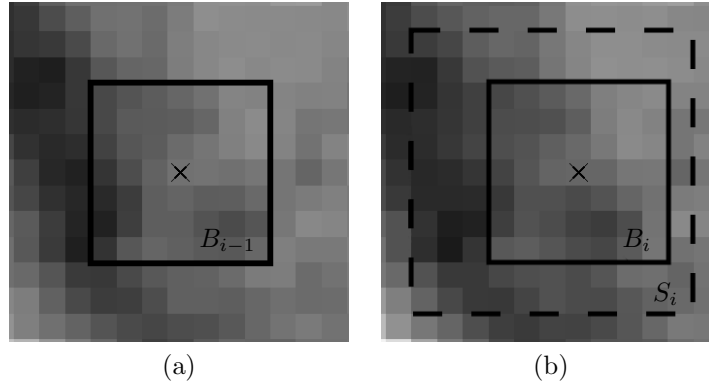


Figure 2. An example for finding a seed point (marked with \times) in an adjacent slice using the block matching algorithm: (a) the reference block B_{i-1} around the seed point in slice $i-1$ (black) that is used for matching and (b) the search area S_i in slice i (dashed black) defined around the orthogonal projection of the seed point and the most similar block B_i based on the median of squared differences measure (solid black)

2.1.1 Block Matching

The *block matching algorithm* is a method for locating similar (“matching”) blocks in two images. Given a quadratic sub image block B_{i-1} in slice $i-1$ containing the intensity values of the associated voxels the goal of this algorithm is to find the sub image block B_i in slice i in a given search area S_i that is most similar to B_{i-1} based on some similarity measure. In our case the block matching is used to find the position of a seed point from slice $i-1$ in slice i . For finding the best match a 7×7 block around the seed point in slice $i-1$ and an 11×11 search area around the orthogonal projection of this point in slice i are used. The center of the most similar block B_i is used as the new seed point in slice i (see Fig. 2). The *median of squared differences* similarity measure has produced the best results in our test cases.

2.1.2 Live-Wire

The *live-wire algorithm*⁷ is an interactive segmentation method formulated as a graph searching problem where the goal is finding the optimal path between user defined nodes on a graph. On this graph V are the *nodes* (*vertices*) representing pixels in the image. A pixel is connected to its eight adjacent pixels by *weighted edges* E . The weight or *costs* $c(\mathbf{p}, \mathbf{q})$ between nodes is the weighted sum of various features based on the intensity values in the image with $c(\mathbf{p}, \mathbf{q}) \in [0, 1]$. The optimal path (i.e. the path with minimum costs) between two nodes is calculated using the *Dijkstra algorithm*.

Barrett and Mortensen suggest the following features (see Ref. 7):

- *Gradient magnitude feature*

$$f_G(\mathbf{p}) = 1 - \frac{G(\mathbf{p})}{\max(G)} \quad (1)$$

where $G(\mathbf{p})$ is the gradient magnitude of a node \mathbf{p} and $\max(G)$ is the maximum gradient in the image.

- *Laplacian zero-crossing feature*

$$f_Z(\mathbf{p}) = \begin{cases} 0, & \text{if } I_L(\mathbf{p}) = 0 \text{ or } \exists \mathbf{q} \in V : I_L(\mathbf{p}) \cdot I_L(\mathbf{q}) < 0 \wedge (\mathbf{p}, \mathbf{q}) \in E \\ 1, & \text{otherwise} \end{cases} \quad (2)$$

where I_L is the convolution of the image using a Laplacian kernel.

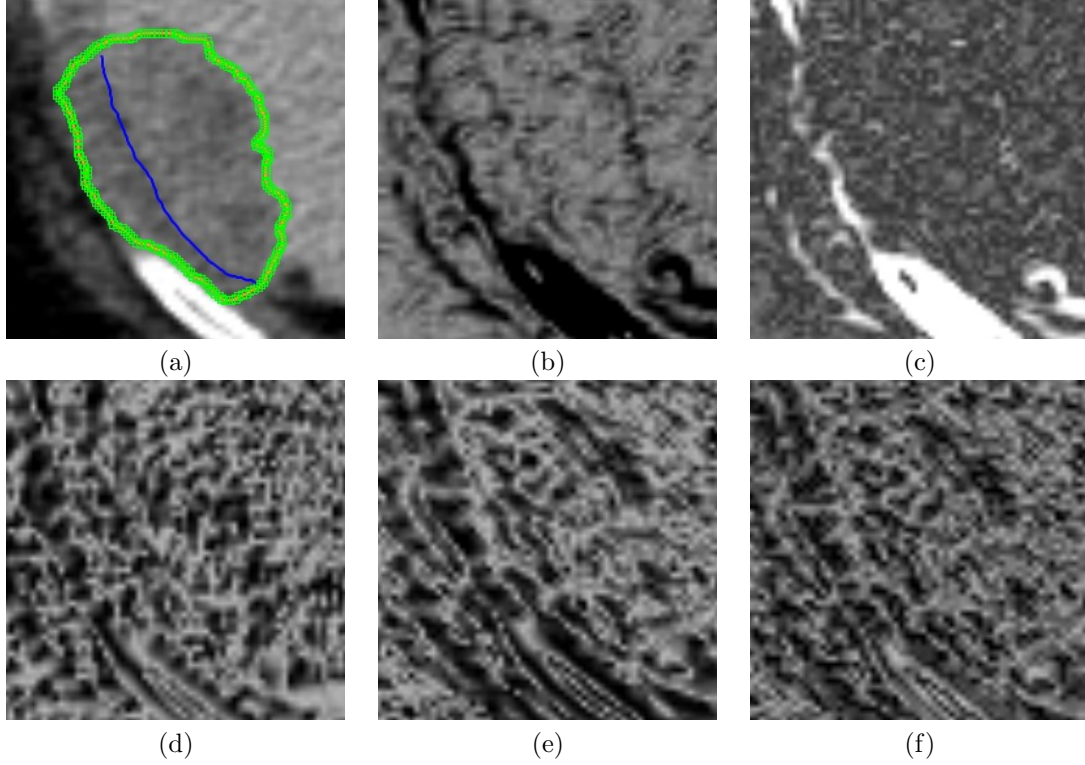


Figure 3. Examples for costs used in the live-wire algorithm (low costs are black, high costs are white) for a given input image and a given user contour (blue/dark gray line) (a): (b) the gradient magnitude feature image (see Eq. 1) and (c) the image based on the preferred gradient magnitude feature (see Eq. 9) (note the high costs assigned the “strong” edges in the input image because the user contour is drawn on “weak” edges); (d) the directional feature costs for northern and (e) for north-western direction; (f) the combination of all costs calculated by Eq. 10 for north-western direction

- *Gradient direction feature*

$$f_D(\mathbf{p}, \mathbf{q}) = \frac{2}{3\pi} (\cos^{-1}(d_p(\mathbf{p}, \mathbf{q})) + \cos^{-1}(d_q(\mathbf{p}, \mathbf{q}))) \quad (3)$$

with

$$d_p(\mathbf{p}, \mathbf{q}) = \mathbf{d}(\mathbf{p}) \cdot \mathbf{l}(\mathbf{p}, \mathbf{q}) \quad (4)$$

$$d_q(\mathbf{p}, \mathbf{q}) = \mathbf{d}(\mathbf{q}) \cdot \mathbf{l}(\mathbf{p}, \mathbf{q}) \quad (5)$$

$$\mathbf{l}(\mathbf{p}, \mathbf{q}) = \begin{cases} \mathbf{q} - \mathbf{p}, & \text{if } \mathbf{d}(\mathbf{p}) \cdot (\mathbf{q} - \mathbf{p}) \geq 0 \\ \mathbf{p} - \mathbf{q}, & \text{if } \mathbf{d}(\mathbf{p}) \cdot (\mathbf{q} - \mathbf{p}) < 0 \end{cases} \quad (6)$$

$$\mathbf{d}(\mathbf{p}) = \begin{pmatrix} \mathbf{g}_y(\mathbf{p}) \\ -\mathbf{g}_x(\mathbf{p}) \end{pmatrix} \quad (7)$$

where $\mathbf{g}(\mathbf{p}) = (\mathbf{g}_x(\mathbf{p}), \mathbf{g}_y(\mathbf{p}))$ is the gradient of the pixel at position \mathbf{p} .

Using these features the costs are given by

$$c(\mathbf{p}, \mathbf{q}) = w_G \cdot f_G(\mathbf{q}) + w_Z \cdot f_Z(\mathbf{q}) + w_D \cdot f_D(\mathbf{p}, \mathbf{q}) \quad (8)$$

with w_G , w_Z and w_D being the weights of the features and $w_G + w_Z + w_D = 1$. Figure 3 shows examples of various costs for a single slice.

In our *live-wire extrapolation* we use the live-wire algorithm to find the best path between two seed points that have been generated from the user contour C^u . Because the costs are calculated based on the gradients of the image the optimal path moves along the border of the object that should be segmented. The calculation for the costs of a path between two seed points has been extended by the *length of the path* which is weighted by 0.0001. This way straight paths are preferred over more complex ones. Moreover the gradient feature has been replaced by a new one which we call *preferred gradient magnitude feature*. Because the user has drawn a contour along the border of the object in slice u we already know the optimal gradient magnitude \widehat{G} along this border. We use the average of all gradients along the path as \widehat{G} . Now we are able to normalize the gradients of all voxels. Costs of 0 are assigned to the known optimal gradient and costs of 1 to the gradient with the maximum difference to \widehat{G} (see also Fig. 3(c)):

$$\widehat{f}_G(\mathbf{p}) = \frac{\left| \frac{G(\mathbf{p})}{\max(G)} - \widehat{G} \right|}{\max(\widehat{G}, |1 - \widehat{G}|)} \quad (9)$$

This is somehow similar to the *on-the-fly training* described by Barrett and Mortensen. Using \widehat{f}_G and the length $l(\mathbf{p}, \mathbf{q})$ between two nodes the costs are given by

$$\widehat{c}(\mathbf{p}, \mathbf{q}) = w_G \cdot \widehat{f}_G(\mathbf{q}) + w_Z \cdot f_Z(\mathbf{q}) + w_D \cdot f_D(\mathbf{p}, \mathbf{q}) + 0.0001 \cdot l(\mathbf{p}, \mathbf{q}) \quad (10)$$

For being less sensitive to noisy images we apply a *Gaussian convolution* to the image before calculating the costs. The following weights for the features have worked well in most cases: $w_G = 0.4$, $w_Z = 0.1$ and $w_D = 0.5$.

2.2 Morphological Postprocessing

As the final step we perform a *morphological postprocessing* on the mask that has been generated from the contours C_i^u . First we perform a *three-dimensional morphological opening* using a $3 \times 3 \times 3$ kernel to remove small artifacts that might have been generated by outliers in the extrapolation step. Then we compute the *connected components* of the mask. Because we assume that the lesion is located at the center of the ROI and because a lesion is a compact and coherent structure we only keep the connected component that is closest to the center. The final step is a *three-dimensional morphological closing* using a $3 \times 3 \times 3$ kernel to fill small gaps and to get a smoother surface (see Fig. 4(d)). Although the postprocessing is a global operation (i.e. it is performed on the whole mask) the changes to parts of the mask that have not been replaced by C_i^u are negligible as long as similar morphological operations have been performed in the initial segmentation process.

3. RESULTS

Our algorithm was evaluated by six radiologists on 108 lesions for which the automatic segmentation was rated as insufficient by the clinicians. The dataset contained lung nodules, liver metastases and lymph nodes and was collected from several clinics and CT scanners. Ratings could be either $--$, $-$, 0 , $+$ or $++$ where we refer to $--$ and $-$ as insufficient segmentations. 0 is an acceptable, $+$ and $++$ are good or excellent segmentations. By using our method 67.6% of the segmentations could be improved to good or excellent ones and 96.3% were

Table 1. Ratings of initially insufficient ($--$, $-$) segmentations after correction with our algorithm.

Rating	Number of lesions	Percent
$--$	1	0.9%
$-$	3	2.8%
0	31	28.7%
$+$	50	46.3%
$++$	23	21.3%

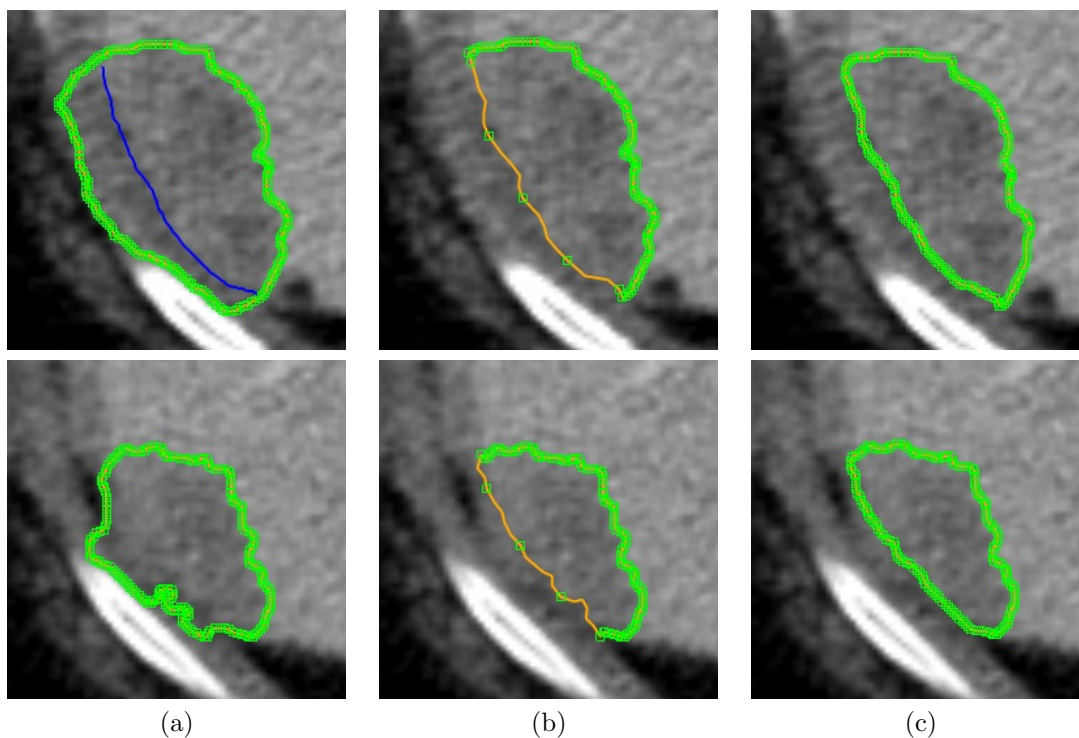


Figure 4. An example for the correction of a liver lesion where segmentation leaked into the intercostal muscles. Top row is the axial slice that the user interacted with, bottom row is another slice of the lesion: (a) the initial segmentation (thick green/gray line) and the user drawn partial contour (blue/dark gray line) that describes the correct border of the lesion; (b) some part of the initial contour has been replaced by the new contour that has been generated from the user contour by moving seed points (green/gray squares) to the current slice using a block matching algorithm and connecting them using live-wire; (c) the final segmentation mask after applying the morphological postprocessing

acceptable or better. A detailed overview on the ratings of the initially insufficient segmentations after manual correction is given in Table 1.

One correction step including both the live-wire extrapolation and the postprocessing takes about 1 to 5 seconds on a 2.66GHz Intel Core2 CPU which is fast enough to allow an interactive correction. The whole correction process of a segmentation including several correction steps and visually validating the results took a median time of about 44 seconds. Our algorithm was rated as both intuitive to use and fast enough for clinical routine by each radiologist.

For comparison: Assuming that the initial segmentation of an insufficiently segmented lesion has about 40 slices (which is a typical value in our database), that half of the slices need to be corrected and that the correction of one slice using only partial contour replacement (i.e. no live-wire extrapolation and no postprocessing) takes 5 seconds, the correction of the whole segmentation would take about 100 seconds. This is 2.3 times slower than our method. A complete pure manual segmentation would take even longer.

4. DISCUSSION

Not much work has yet been done on 3D manual correction of segmentation masks. Existing methods do not use image data or they are not intuitive to use.²⁻⁵ Our novel approach is based on an intuitive two-dimensional user interaction and uses gradients and intensity values to perform a three-dimensional correction. It operates locally and computation times are low so our algorithm allows an interactive correction of the initial segmentation. Our algorithm is also applicable for other segmentation correction tasks as long as the segmented objects are compact and coherent. Furthermore the objects need to be small because the computation time grows linear in the number of voxel in the ROI and thus the performance might not be fast enough for an interactive correction.

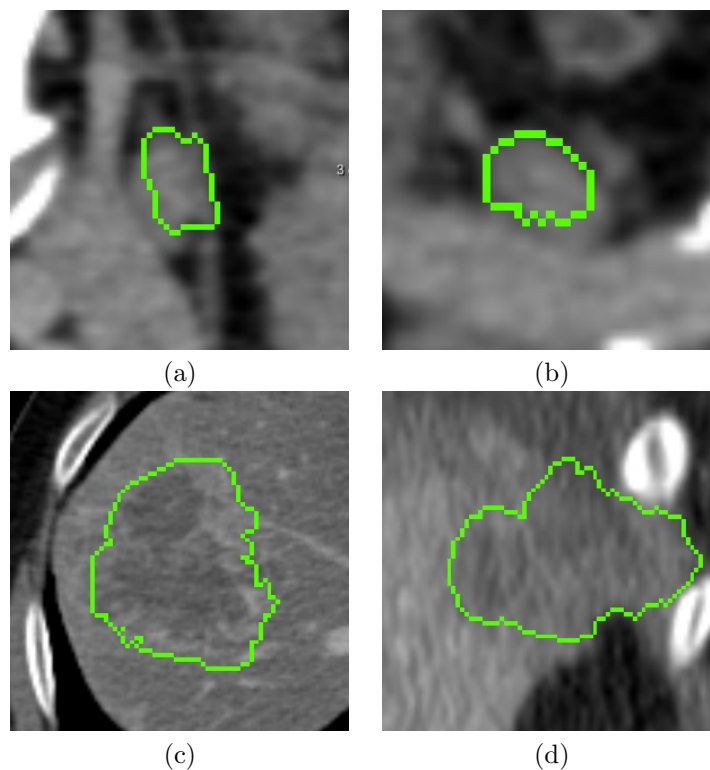


Figure 5. The lesions for which the segmentations have been rated as insufficient even after manual correction. The segmentations shown in the images are the *initial* segmentations. (a) and (b) are lymph nodes with low contrast to adjacent structures (e.g. vessels); (c) is a liver lesion with no visible border to the liver tissue; (d) is a liver lesion with no contrast to the intercostal muscles

The data used for evaluation contained lesions for which the automatic segmentation results were insufficient. Thus the data represents challenging segmentation tasks. The lesions for which the segmentations have been rated as insufficient even after manual correction are shown in Fig. 5. In these cases the borders between the lesion and the surrounding structures are not clear (even for the clinicians) or the contrast was too low and therefore there are no edges as needed for the live-wire algorithm.

5. CONCLUSION AND FUTURE WORK

We have presented an intuitive and fast method for local three-dimensional manual correction of segmentations based on a two-dimensional user interaction on an arbitrary slice. Our algorithm is fast enough for use in clinical routine which has been shown in an evaluation by six radiologists. It is independent of the specific lesion type and the algorithm that is used for the initial segmentation. Moreover it is a general approach that can be used for other segmentation correction tasks as well after minor modification.

Future work could especially aim at improving the live-wire extrapolation. The similarity measure of the block matching should be improved to be less sensitive to structures in the image that are different from the lesion that should be segmented (e.g. vessels). Moreover the cost function of the live-wire algorithm could be further improved to be more robust to noise and to generate smoother paths between the seed points. These optimizations would both improve the quality of the segmentation and reduce overall correction times because less correction steps would be necessary.

ACKNOWLEDGMENT

This work was supported by a research grant from Siemens Healthcare, Computed Tomography, Forchheim, Germany.

REFERENCES

- [1] Kang, Y., Engelke, K., and Kalender, W. A., “Interactive 3D editing tools for image segmentation,” *Medical Image Analysis* **8**(1), 35–46 (2004).
- [2] Timinger, H., Pekar, V., von Berg, J., Dietmayer, K., and Kaus, M., “Integration of interactive corrections to model-based segmentation algorithms,” in [*Bildverarbeitung für die Medizin*], 171–175 (2003).
- [3] Bornik, A., Beichel, R., and 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*], 197–206 (2006).
- [4] Schwarz, T., Heimann, T., Tetzlaff, R., Rau, A.-M., Wolf, I., and Meinzer, H.-P., “Interactive surface correction for 3D shape based segmentation,” in [*Proc. SPIE Medical Imaging 2008: Image Processing*], **6914**(1), 69143O (2008).
- [5] Grady, L. and Funka-Lea, G., “An energy minimization approach to the data driven editing of presegmented images/volumes,” in [*Proceedings of MICCAI 2006*], **2**, 888–895 (2006).
- [6] 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., and Peitgen, H.-O., “OncoTREAT: a software assistant for cancer therapy monitoring,” *Int. J. CARS* **1**(5), 231–242 (2007).
- [7] Barrett, W. A. and Mortensen, E. N., “Interactive live-wire boundary extraction,” *Medical Image Analysis* **1**, 331–341 (1997).