An Energy Minimization Approach to the Data Driven Editing of Presegmented Images/Volumes*

Leo Grady[1,2] and Gareth Funka-Lea[1,3]

[1] Department of Imaging and Visualization
Siemens Corporate Research
Princeton, NJ, USA
[2] leo.grady@siemens.com,
[3] gareth.funka-lea@siemens.com

Abstract. Fully automatic, completely reliable segmentation in medical images is an unrealistic expectation with today's technology. However, many automatic segmentation algorithms may achieve a near-correct solution, incorrect only in a small region. For these situations, an interactive editing tool is required, ideally in 3D, that is usually left to a manual correction. We formulate the editing task as an energy minimization problem that may be solved with a modified version of either graph cuts or the random walker 3D segmentation algorithms. Both algorithms employ a seeded user interface, that may be used in this scenario for a user to seed erroneous voxels as belonging to the foreground or the background. In our formulation, it is unnecessary for the user to specify both foreground and background seeds.

1 Introduction

Automatic segmentations of targets in images/volumes often require some degree of editing to meet the needs of a particular user (e.g., a physician). The question that we are concerned with is: Given a pre-existing segmentation (obtained through other means, e.g., an automatic algorithm), how can one edit the segmentation to correct problems? We formulate the editing task as an energy minimization problem and show how it may be optimized with a modified graph cuts [1] or random walker algorithm [2]. Another view of this work is that we detail how a prior segmentation may be seamlessly combined with graph cuts or the random walker algorithm to allow for editing, while maintaining the important property of both algorithms that an arbitrary segmentation may be achieved with enough interaction. We will use the term **presegmentation** to refer to the prior, incorrect, pre-existing segmentation that was obtained through other means and presented for editing.

In our experience with clinicians, an editing tool is expected to have the following characteristics:

1. Operate locally to the interaction site.

* Published in: Proceedings of Medical Image Computing and Computer-Assisted Intervention — MICCAI 2006, pp. 888–895, 2006, ed. Rasmus Larsen and Mads Nielsen and Jon Sporring, Vol. II, LNCS 4191, Oct., Springer.

2. Operate quickly.
3. Produce modifications in 3D, not just on the viewing slice.
4. Produce intuitive solutions.

The graph cuts and random walker interactive segmentation algorithms appear to be good candidates for editing. Both algorithms require the user to place marks with a mouse (hereafter referred to as **seeds**), to indicate a few pixels belonging to the foreground/background of the target object. These seed locations are then used to produce a full 3D segmentation (labeling). A characteristic of the algorithms is that an undesired segmentation may be easily modified by adding new seeds, which typically results in faster processing of the updated segmentation. In our problem of editing, we would like to preserve the quality of fast modification of the segmentation but, instead of being given a previous set of seeds, we are given a previous complete segmentation produced by another algorithm. Use of these algorithms for editing therefore requires that we preserve the character of these algorithms as fast, 3D, and intuitive, while enforcing a locality of operation.

Implicit in the requirements for local operation and intuitive results is a requirement for stability. In particular, when editing is invoked the presegmented result should not change if the user chooses not to interact. Although obvious, this requirement is not met by many potential approaches. If the editing will be done at a pixel level and the presegmentation is at a subpixel resolution then the first step of the editing will be a pixelation of the presegmentation which will change the results due to sampling even if the user does not change the label of a single pixel. The editing we describe will be at a pixel level and we assume that the presegmentation is at the same level of resolution. Another effect can also lead to instability when initiating editing — if the data driven editing is formulated as an optimization, and the current presegmentation is not a local optimum, then the segmentation result will change with the optimization even without user input. Since the source of our presegmentation is unknown, we will formulate our optimization in such a way that the presegmentation is a global optimum without user input.

As stated by Kang *et al.* [3], "...the literature on editing tools in general and on 3D tools in particular is sparse." In fact, many publications on medical image segmentation explicitly assume the availability of a manual editing tool to correct undesirable results. Although some user interaction is clearly necessary, our goal is to provide a tool that requires minimal user interaction to achieve the desired, edited, result. Kang *et al.* [3] introduce three tools for interactive correction of a presegmented structure. The first tool allows the user to select a VOI, within which holes in the segmentation are filled. The second tool allows a user to bridge points in the segmentation to indicate to the hole-filling system that a volume in the segmentation should be filled. The final tool introduces control points on the presegmented surface that the user is allowed to drag/modify. Modification of a control point on the boundary introduces a displacement field on nearby control points that results in the displacement of a boundary region in the neighborhood of the modified control point. Each of these tools has the drawback that the image content is ignored in modifying the presegmentation. In the approach we present here, the user seeds, presegmentation and image content all impact the edited segmentation.

There are a number of tools for interactive segmentation in 2D [4,5,6] and 3D [7,8,9]. However, none of these are formulated to make use of a presegmentation. In addition there is a large body of literature in the computer aided design and computer graphics communities that looks at graphical model editing, but the editing is done without the influence of image data. Our work is related to the large body of work on image segmentation using shape priors (see, for instance, [10,11,12,13,14,15]). The presegmentation in our work has some of the aspects of a shape prior. However, shape priors are built from a sampled distribution of shapes while in the case of a presegmentation there is only one instance of a prior shape. So, for our problem, there is no learned uncertainty in the prior information. Instead, the goal is to deviate from the presegmentation locally to the interaction site and relative to the image data.

This paper is organized as follows. Section 2 formulates the presegmentation editing problem as a graph cuts or random walker segmentation problem. Section 3 offers several 2D and 3D editing results. Finally, Section 4 presents concluding remarks.

2 Method

In order to meet the goal stated above that an unedited presegmentation returns the presegmentation as the optimum, we avoid a continuum formulation out of concern that the discretization step might alter the presegmentation. Therefore, our formulation will be on a discrete space or, in general, a graph. We begin by defining a precise notion for a graph. A **graph** [16] consists of a pair $G = (V, E)$ with **vertices (nodes)** $v \in V$ and **edges** $e \in E \subseteq V \times V$. An edge, e, spanning two vertices, v_i and v_j, is denoted by e_{ij}. A **weighted graph** assigns a value to each edge called a **weight**. The weight of an edge, e_{ij}, is denoted by $w(e_{ij})$ or w_{ij} and is assumed to be positive. The **degree** of a vertex is $d_i = \sum w(e_{ij})$ for all edges e_{ij} incident on v_i. We associate each pixel (voxel) with a node and we will assume that each pixel (voxel) is connected by an edge to its four (six) cardinal neighbors.

Define an affinity weighting between pixels as given by the typical [7,2] Gaussian weighting function

$$w_{ij} = \exp\left(-\beta(g_i - g_j)^2\right), \tag{1}$$

where g_i represents the grayscale intensity at node (pixel) v_i.

Define a presegmentation, p, determined by another process (e.g., a separate, automatic segmentation algorithm), as

$$p_i = \begin{cases} 1 & \text{if } v_i \text{ was presegmented as foreground,} \\ 0 & \text{if } v_j \text{ was presegmented as background.} \end{cases} \tag{2}$$

Given a segmentation p, define the editing problem as the minimization of the energy functional

$$Q(x) = \sum_{e_{ij}} w_{ij}(x_i - x_j)^2 + \gamma \left(\sum_i (1 - p_i)\, x_i + \sum_i p_i\, (1 - x_i) \right), \tag{3}$$

with respect to the foreground indicator function x, defined on the nodes, where γ is a parameter indicating the strength of the presegmentation. This functional encourages

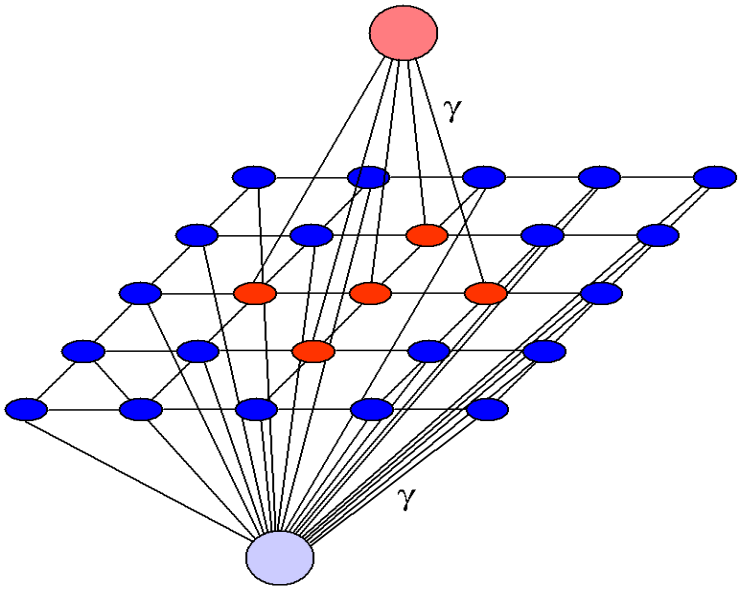

Fig. 1. Graphical interpretation of the editing formulation given a presegmentation. The dark red and blue pixels correspond to the presegmented foreground/background. The light red/blue pixels represent "supernodes" that are attached to the presegmented foreground/background with strength γ. The energy functional of (3) may be minimized by applying either the graph cuts (binary minimization) or the random walker (real-valued minimization) algorithm to this graph construction. User placed editing seeds may now be employed in the manner of the hard constraints used in the standard formulation of both algorithms.

the presegmentation as well as encouraging a data-driven smoothness in the form of the first term. Note that, with a sufficiently large γ, the presegmentation will always be returned. Given a user-defined editing set of nodes (possibly empty) marked as foreground seeds $F \subset V$ and a user-defined editing set of nodes (possibly empty) marked as background seeds, $B \subset V$, such that $F \cap B = \emptyset$, the seeds are incorporated into the minimization of (3) by performing a constrained minimization of $Q(x)$ with respect to the constraints

$$x_i = \begin{cases} 1 & \text{if } v_i \subset F, \\ 0 & \text{if } v_j \subset B. \end{cases} \tag{4}$$

If the minimization of $Q(x)$ in (3) is forced to give a binary-valued optimum, x, for all unseeded nodes, then the minimization of (3) is given by the graph cuts algorithm of [7] with the construction of Figure 1. In the language of [7], seeds are given by F, B, N-links have weight w_{ij} and each node, v_i, is connected via a T-link to a foreground (if $p_i = 1$) or background (if $p_i = 0$) "supernode" with weight γ. If the optimization of (3) is performed with respect to a real-valued x (afterward thresholded at 0.5 to produce a "hard" segmentation) the random walker algorithm may be performed with

the same weighted graph construction given above for graph cuts [9]. Since the random walker has a provable robustness to noise [9] not offered by the graph cuts algorithm and because the "soft" confidence values for each node that are returned by the random walker algorithm are generally beneficial for visualization and smoothing purposes, we will focus here on the random walker solutions for the editing problem given (3).

One view of the editing formulation of (3) is as a statistical *prior* that is fed to the segmentation algorithm. Statistical priors may be incorporated into either the graph cuts or the random walker algorithm in the same manner — by connecting each node to a floating foreground/background (i.e., source/terminal) node with strength proportional to the prior [7,9]. Figure 1 gives an example of this construction.

The formulation given above satisfies three of the four design criteria for an interactive editing algorithm. Modifications behave intuitively, are performed in 3D and the updates may be computed quickly (we refer the reader to [9] for the reasons why this computation is more efficient in the case of the random walker). However, the criterion of local operation is not incorporated into the above formulation, i.e., the segmentation could change at any location in the image. Therefore, we propose to make the presegmentation strength, γ, a function of distance from the seed locations [4], i.e.,

$$\gamma_i = \kappa \exp\left(-\frac{\mathrm{d}\left(v_i, v_j\right)}{\sigma}\right), \tag{5}$$

where $d(v_i, v_j)$ is the minimum distance from v_i to all $v_j \in F, B$. Therefore, the κ parameter indicates the overall strength of consideration given to the presegmentation and the parameter σ controls the locality of the modification given by the seeds.

3 Results

Validation of an editing algorithm such as this is difficult, since the ultimate metric of utility is the match between the intuition of the user and the result of the editing. However, it is possible to demonstrate that the editing tool satisfies the remaining design criteria of speed, locality and 3D operation.

We will begin with 2D examples in order to characterize the algorithm behavior and report calculation speeds. Figure 2 shows six examples of images taken from different imaging modalities. For each image, an incorrect presegmentation was given from a separate system (e.g., an automatic segmentation algorithm) that a user would want to correct. The incorrect presegmentations are given in the second column, outlined in blue/gray. A user may place foreground seeds to include regions excluded in the presegmentation, indicated by green/dark gray marks or background seeds to exclude regions included by the presegmentation, indicated by yellow/light gray marks. Although none of the examples in the second column include both foreground and background seeds, there is no limitation to using both seed types. These experiments were conducted on an Intel Xeon with a 2.40GHz processor using a conjugate gradients solver for the random walker algorithm. From top to bottom — the aortic aneurysm CT image had 512×512 pixels and the editing required 18.28s, the bone CT image had 512×512 pixels (cropped

[4] We would like to thank Reto Merges for this modification

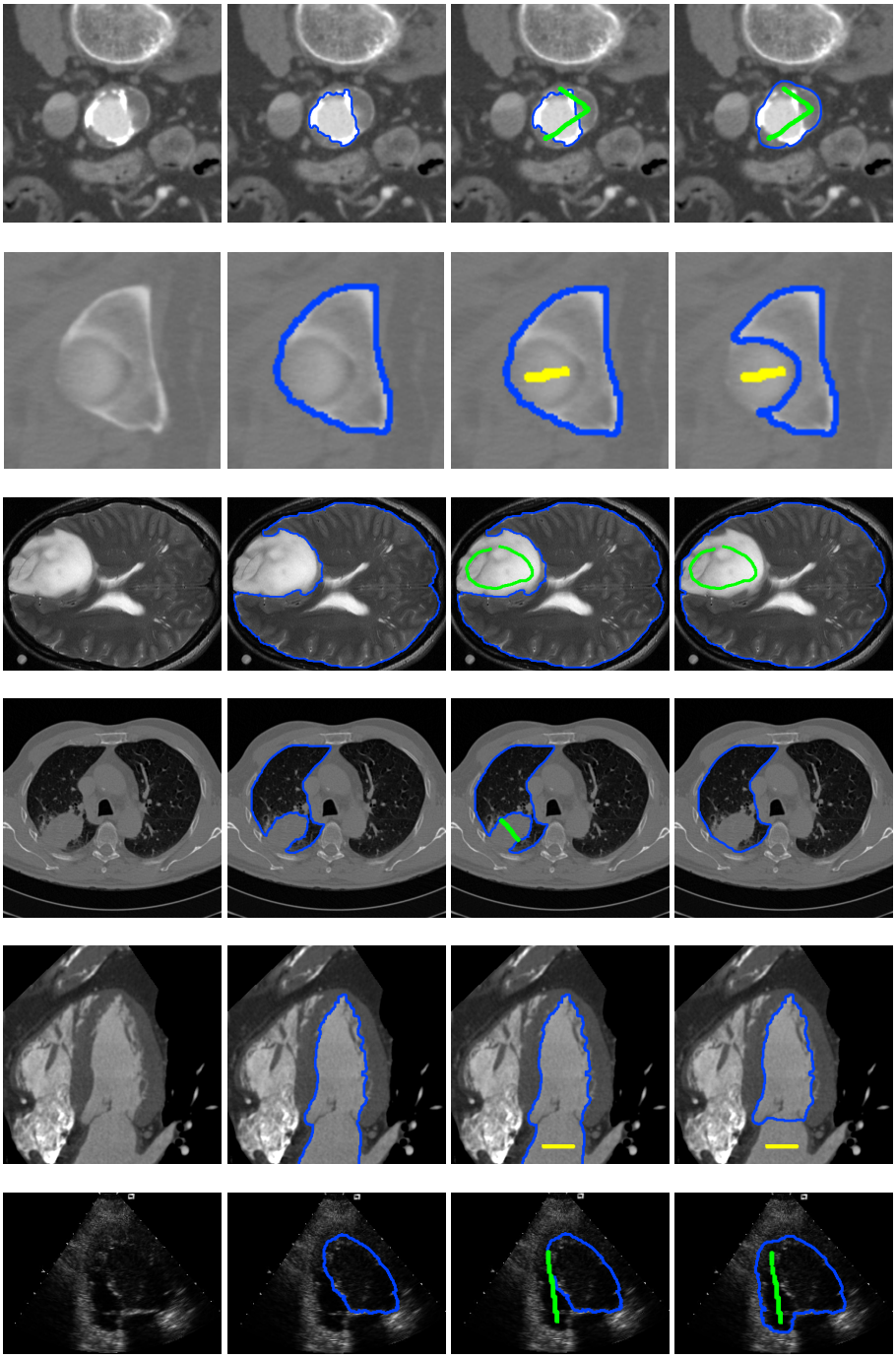

Fig. 2. 2D examples of our editing algorithm. From left to right — First column: The original image. Second column: The presegmentation (outlined in blue/gray) provided by another system (e.g., an automatic algorithm). Third column: The user-placed seeds used to correct the segmentation. Green/dark gray seeds indicate that the user wants to include this region in the segmentation while yellow/light gray seeds indicate that the user wants to exclude this region from the segmentation. Fourth column: The updated segmentation.

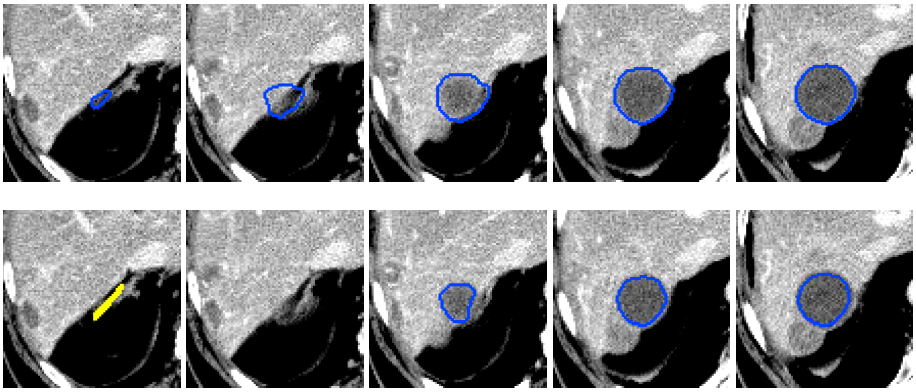

Fig. 3. A 3D example of our editing algorithm. In all slices the blue/gray lines indicate the segmentation border. Top row: Axial slices of the original segmentation of a liver tumor in CT with a leak into the surrounding tissue. Bottom row: The corrected segmentation, after placement of background (exclude) seeds in one slice by the user, given on the leftmost slice by the yellow/light gray seeds.

in the figure) and required 9.5s, the brain tumor MR image was 295×373 and required 1.79s, the lung tumor CT image was 512×512 and required 23.67s, the left ventricle CT image was 256×256 and required 1.95s and the left ventricle ultrasound image was 240×320 (cropped in the figure) and required 2.58s for editing.

Figure 3 shows the same experiment run on a 3D dataset of a liver tumor in which the presegmentation erroneously included surrounding tissue, due to a vessel passing nearby. Background seeds were placed on a single slice but, because the energy minimization is formulated on an arbitrary graph (in this case, a 3D lattice), the resulting solution effects all slices. Given this volume, cropped to $89 \times 90 \times 29$, and using the same computer as in the 2D experiment, producing an edited solution in 3D required 3.95s.

4 Conclusion

Perfectly reliable, automatic segmentation of an object in a medical image is unrealistic with today's segmentation technology. However, an automatic system may get close to the user-desired segmentation. Therefore, the availability of a smart editing tool is essential, since a manual correction is far too time consuming, especially in 3D data. The "scribble" interface of the graph cuts and random walker algorithms provide a natural user interface that allows the user to seed image regions to include or exclude in the edited segmentation. With the native definitions of these algorithms, both foreground and background regions must be seeded and information from a presegmentation cannot be used. In this paper, we have shown how the presegmentation may be used to frame the editing problem in an energy minimization framework that permits optimiza-

tion with either the graph cuts or random walker algorithm, depending on whether or not segmentation confidences are required.

Our energy minimization framework was tested on several 2D and 3D editing examples (using the random walker minimization) and the results were displayed in Figures 2 and 3 with timing information. Overall, the energy minimization framework provides a meaningful, smart, image-dependent method of editing a presegmented volume with known minimization techniques. Finally, the editing presented here satisfies our stated desired qualities of an editing algorithm in that the edited segmentation is obtained locally, quickly, intuitively and operates in 3D.

References

1. Boykov, Y., Jolly, M.P.: *Interactive graph cuts* for optimal boundary & region segmentation of objects in N-D images. In: Proc. of ICCV 2001. (2001) 105–112 1
2. Grady, L., Funka-Lea, G.: Multi-label image segmentation for medical applications based on graph-theoretic electrical potentials. In: Computer Vision and Mathematical Methods in Medical and Biomedical Image Analysis, ECCV 2004. Number LNCS3117, Prague, Czech Republic, Springer (2004) 230–245 1, 3
3. Kang, Y., Engelke, K., Kalender, W.A.: Interactive 3D editing tools for image segmentation. Medical Image Analysis **8** (2004) 35–46 2
4. Falcão, A.X., Udupa, J.K., Samarasekera, S., Sharma, S., Elliot, B.H., de A. Lotufo, R.: User-steered image segmentation paradigms: Live wire and live lane. Graphical Models and Image Processing **60** (1998) 233–260 3
5. Mortensen, E., Barrett, W.: Interactive segmentation with intelligent scissors. Graphical Models in Image Processing **60**(5) (1998) 349–384 3
6. Elder, J.H., Goldberg, R.M.: Image editing in the contour domain. IEEE Transactions on Pattern Analysis and Machine Intelligence **23**(3) (2001) 291–296 3
7. Boykov, Y., Jolly, M.P.: Interactive organ segmentation using graph cuts. In: Medical Image Computing and Computer-Assisted Intervention, Pittsburgh, PA (2000) 276–286 3, 4, 5
8. Lefohn, A.E., Cates, J.E., Whitaker, R.T.: Interactive, GPU-based level sets for 3D segmentation. In: Medical Image Computing and Computer Assisted Intervention (MICCAI). (2003) 564–572 3
9. Grady, L.: Multilabel random walker image segmentation using prior models. In: Proceedings of the 2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. Volume 1 of CVPR., San Diego, IEEE, IEEE (2005) 763–770 3, 5
10. Cootes, T.F., Hill, A., Taylor, C.J., Haslam, J.: Use of active shape models for locating structures in medical images. Image Vision Comput. **12**(6) (1994) 355–365 3
11. Leventon, M.E., Grimson, W.E.L., Faugeras, O.: Statistical shape influence in geodesic active contours. In: Proc. Conf. Computer Vision and Pattern Recognition. Volume 1., Hilton Head Island, SC (2000) 316–323 3
12. Cremers, D., Tischhäuser, F., Weickert, J., Schnörr, C.: Diffusion snakes: Introducing statistical shape knowledge into the Mumford-Shah functional. International Journal of Computer Vision **50**(3) (2002) 295–313 3
13. Rousson, M., Paragios, N.: Shape priors for level set representations. In: Proc. of ECCV 2002. Volume 2. (2002) 78–92 3
14. Kumar, M.P., Torr, P.H., Zisserman, A.: OBJ CUT. In: Proc. of CVPR 2005. Volume 1. (2005) 18–25 3

15. Freedman, D., Zhang, T.: Interactive graph cut based segmentation with shape priors. In: Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR). Volume 1. (2005) 755–762 3
16. Harary, F.: Graph Theory. Addison-Wesley (1994) 3