

Segmentation Techniques for Medical Image Analysis

BindushreeYadav. M¹ Prof. Anand Jatti²

¹MTech, BMSP&I ²Associate Professor, Department of Instrumentation Technology

^{1,2}R. V. College of Engineering, Bangalore, Karnataka, India

Abstract---In day-to-day life, new technologies are emerging in the field of Image processing, especially in the domain of segmentation. Segmentation is also useful in Image Analysis and Image Compression. Segmented images are further used as input for various applications such as classification, recognition and measurement. More specifically, in the case of medical applications image segmentation is an important step towards the study of anatomical structures, diagnosis and the planning of surgeries or other forms of treatment. This paper presents a brief outline on some of the most common Medical segmentation techniques like thresholding, Region growing, snake algorithm, Active contour method etc., mentioning its advantages as well as the drawbacks. Some of the techniques are suitable for noisy images.

I. INTRODUCTION

Medical Image Segmentation is a process of automatic or semi-automatic detection of 2D or 3D image. Segments often correspond to different tissue classes, organs, pathologies, or other biologically relevant structures. A major difficulty of medical image segmentation is the high variability in medical images. First and foremost, the human anatomy itself shows major modes of variation. Furthermore many different modalities (X-ray, CT, MRI, microscopy, PET, SPECT, OCT and many more) are used to create medical images. For years, various aspects of segmentation features and algorithms have been extensively explored in a host of publications however the problem remains challenging with no general and unique solutions due to a large and constantly growing number of different objects of interest, large variations of their properties in images, different medical imaging modalities and associated changes of signal homogeneity, variability and noise for each object. This is where automatic, or semi-automatic, algorithms are of interest. In the best of worlds, we would like to have algorithms which can automatically detect diseases, lesions and tumors, and highlight their locations in the large pile of images. But another complication arises; we also have to *trust* the results of the algorithms. This is especially important in medical applications - we don't want the algorithms to signal false alarms, and we certainly don't want them to miss fatal diseases. Therefore, developing algorithms for medical image analysis requires thorough validation studies to make the results usable in practice. This adds another dimension to the research process which involves communication between two different worlds - the patient-centered medical world, and the computer centered technical world. The symbiosis between these worlds is rare to find and it requires significant efforts from both sides to join on a common goal.

A. MEDICAL IMAGE SEGMENTATION

Segmentation techniques used for medical image analysis can be mainly classified into three types: Thresholding and Region growing, Variational methods, combinatorial methods. A brief review of these techniques is discussed below.

1) Thresholding and region growing

Early, and simple, techniques for segmentation mainly used the assumption that relevant objects in an image can be identified based on intensity values. The most simple approach identifies objects using a single threshold value, such that pixels above and below the threshold are object pixels and background pixels respectively. This works fine for high contrast objects with a sharp edge, but the method often fails as soon as the edges are smooth, of varying intensity and influenced by noise. This is most often the case for natural images, which limits the usefulness of this approach. A slightly more sophisticated version of thresholding is region growing algorithms. The basic idea is to start from a given seed point which is known to be an object pixel. The neighborhood of the pixel is classified as background or object depending on a threshold value. The (connected) object is segmented by a recursive search through the pixels which are classified as object. A typical problem with this type of method is leakage, since it is hard to set a threshold value which confines an actual object.

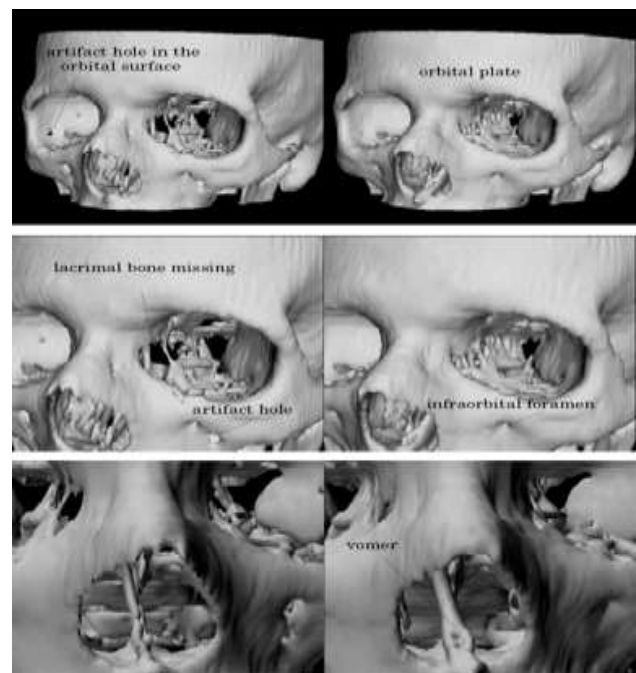


Fig. 1: Segmentation by thresholding CT values. Right Segmentation using additional local shape information.

The top images show an overview of the surface rendered segmentation. Middle images show a close up of the left orbital region. Bottom images show a close up of the nasal cavity.

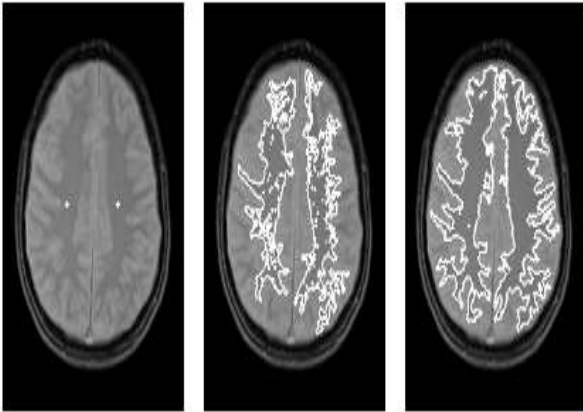


Fig. 2: Region Growing applied for Brain Top view

II. VARIATIONAL METHODS

Variational methods are based upon describing energy functional where the optimum defines a good segmentation. The functional is typically depending on a curve, which defines the partitioning of the image, and a number of images derived terms such as image intensity, image gradients, etc.

A. Snakes

The technique of *snakes* or *active contours* has become quite popular for a variety of applications in the past few years. This methodology is based upon the utilization of deformable contours which conform to various object shapes and motions. Snakes have been used for edge and curve detection, segmentation, shape modeling, and visual tracking. Active contours have also been widely applied for various applications in medical imaging. For example, snakes have been employed for the segmentation of myocardial heart boundaries as a prerequisite from which such vital information such as ejection-fraction ratio, heart output, and ventricular volume ratio can be computed. The first work in this direction was the *Snake* in Kass et al. [1988] which used an explicit type of curve representation. The energy is defined by:

$$E[C(s)] = - \int |\nabla I(C(s))|^2 ds + v_1 \int |C'(s)|^2 ds + v_2 \int |C''(s)|^2 ds$$

Eq. (1)

Where $C(s)$ is a parametric curve with parameter s , I is the image, and C' and C'' are the first and second derivatives of C with respect to its parameter s . The first term is referred to as the *external* energy and the two last terms are the *internal* energies of the snake. The external energy is derived from the image, and is used to drive the curve towards points with high gradient magnitude, i.e. strong edges. The first internal energy, weighted by $v_1 \geq 0$, measures the length of the snake, while the second, weighted by $v_2 \geq 0$ measures the stiffness. Good segmentations are represented by curves which optimize the energy, i.e. where the *first variation* of E vanishes. The first variation can be viewed as a

generalization of the first derivative from “ordinary” calculus. Recall that stationary points (maxima, minima, saddles) of a function can be found by searching for points where the first derivative is zero.

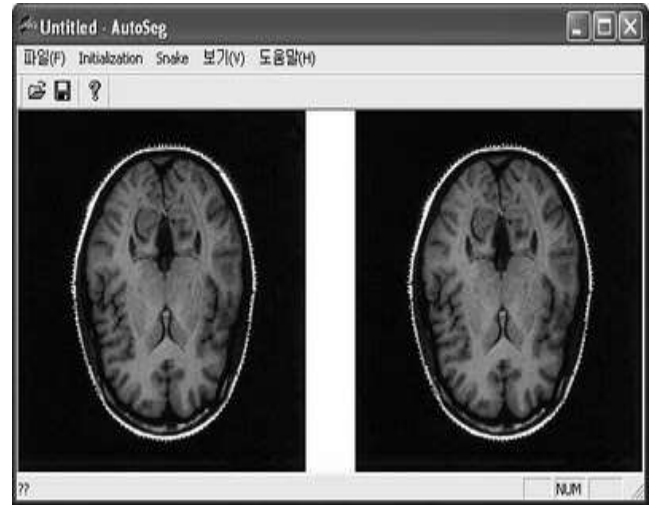


Fig. 3: AutoSeg application (Left: Processing Image, Right: Result Image)

The first variation in the calculus of variations has exactly the same meaning, but for functional (a “function of a function” such as Eq. (1)) instead of functions. Without going into details, this can be expressed by the Euler-Lagrange equation denoted by $\delta E/\delta C = 0$. This equation results in a partial differential equation (PDE) which, in most cases, is hard to solve directly. Instead an energy optimum is found by evolving the curve in the steepest descent direction of the energy, such that $\partial C/\partial t = -\delta E/\delta C$. This is the basic idea behind the popular variational methods used in image processing. Note however that any solution will be a local optimum, so the result strongly depends on a proper initialization, or a proper starting curve. The Snakes algorithm was the starting point for image segmentation using variational methods. However, it suffers from a number of drawbacks. The main drawback is the explicit curve representation which does not easily allow topological changes and requires complex re-parameterization algorithms. Second, since the nodes of the curve are affected by local image features, i.e. there is no region based information; the solutions tend to be very sensitive to a proper initialization. Also, from a mathematical point of view, the energy function is not intrinsic since it depends on the parameterization of the curve. This means that the energy will change if the parameterization changes, which clearly is an undesirable property.

B. Level set methods

To address the first issue regarding curve parameterization, approaches using the *implicit* level set method by Osher and Sethian [1988] were simultaneously proposed by Caselles et al. [1993] and Malladi et al. [1993, 1994]. Using these ideas, a curve is represented implicitly as the zero level set of a time dependent function $\phi : \Omega \rightarrow \mathbb{R}$, such that $C = \{x(t) \in \Omega : \phi(x, t) = 0\}$. By differentiating with respect to time, we can couple the motion of the curve dx/dt with an evolution of the level set function ϕ using a PDE $\partial\phi/\partial t = -dx/dt \cdot \nabla\phi$. A special case is when the motion is restricted to the normal

direction of the curve, i.e. $dx/dt = Fn$, where $n = \nabla\phi / |\nabla\phi|$. In this case, the scalar function F is usually referred to as the *speed function*. This representation has the main advantage of allowing arbitrary topological changes, so the initial curve does not need to have the same topology as the segmented object. Unlike the Snakes algorithm, which was motivated by energy minimization, the work by Caselles et al. and Malladi et al. was motivated by a geometric curve evolution approach. The basic idea was to generate a speed function which “pulled” the curve towards the boundaries of the target object while the curve was regularized with curvature motion. A typical speed function can be based on image gradients, such that it approaches zero when the norm of the image gradient is large (i.e. there is a distinct edge).

The main advantage of this approach is the level set representation, which allows arbitrary topological changes and robust and stable numerical schemes for the curve propagation. However, the motivation is based on geometrical aspects of the image and the curve, so it is hard to validate the solutions and perform “deeper” mathematical analysis.

C. Geodesic active contours

The next major step in variational methods-based image segmentation was the *Geodesic active contour* proposed simultaneously by Caselles et al. [1997] and Kichenassamy et al. [1995]. In this work, it is shown that a special case of the Snakes model can be interpreted as finding geodesics (locally shortest paths) in a space with a metric derived by image content. This formalism provides an analogue between segmentation using active contours and minimal distance (geodesic) computations. Formally, the energy to be minimized has the following form:

$$E_{GAC}[C(s)] = \int_0^{L(C)} g(|\nabla I(C(s))|) ds \quad \text{Eq (2)}$$

Where $L(C)$ is the length of C , $g(|\nabla I|)$ is a strictly decreasing inverse edge indicator function, typically $g(|\nabla I|) = 1/(1 + |\nabla I|)$. We can note that choosing $g = 1$ gives the length of the curve $C(s)$, so minimizing Eq. (3.2) gives a minimal curve where the length of the curve is weighted by the function $g(|\nabla I|)$.

Then, if the curve is represented as a level set, it is shown that the geodesic computations reduce to a form similar to the work in Caselles et al. [1993], Malladi et al. [1993, 1994]. The geodesic active contour model provides a coupling between segmentation based on energy minimization and the level set framework. Thus, it is contained in a rigorous mathematical context of optimization while the curve representation is flexible and robust, allowing both thorough analysis and practical implementation.

D. Active contours without edges

So far, the active contour models have primarily defined the objects by image gradients. This is often problematic due to varying edge contrast and noise. The models will fail completely for objects with too blurred or weak edges, and there are risks of leakage if the edge is not well defined on the complete perimeter of the object. In contrast, the active

contours without edges by Chan and Vese [2001] is based on regional measures and do not depend on any edge definition. This approach incorporates region information and is a special case of the Mumford-Shah functional for segmentation [Mumford and Shah, 1989]. Basically, it solves the minimal partition problem which aims to find a partitioning of the image which best separates the interior of the curve from the exterior. Formally, it is described as:

$$E_{CV}[C(s)] = \mu \int_0^{L(C)} ds + \iint_{\Omega_C} (I(x, y) - c_1)^2 dx dy + \iint_{\Omega \setminus \Omega_C} (I(x, y) - c_2)^2 dx dy \quad \text{Eq (3)}$$

Where Ω_C is the interior of the curve C , c_1 and c_2 are the average image intensities in the interior and exterior of C respectively and $\mu \geq 0$ is a weight parameter. The first term is a regularization which minimizes the length and the second last terms provide the balancing between the interior and exterior. It can be noted that omitting the regularization by $\mu = 0$ leads to a solution equivalent to simple thresholding. This model assumes the image to be separable into two regions (phases) which can be reasonably well approximated by constant values c_1 and c_2 . This is a crude model which was later generalized to multi-phase piecewise-smooth approximations in Vese and Chan [2002]. The main advantage of this model is the global dependence and the tendency to produce globally optimal solutions in practice. However, for many applications in medical image segmentation, the influence of the background intensity can be problematic.

III. COMBINATORIAL METHODS

Whereas the previous section described approaches where the image domain is regarded as *continuous* and segmentation was posed as an optimization a continuous space, there are *combinatorial* methods formulated as a *discrete* optimization problem on the *pixels* of the image. This section will introduce the main directions in this area for completeness, but since this is not the focus of this thesis, the presentation will be brief.

A. Optimality of solutions

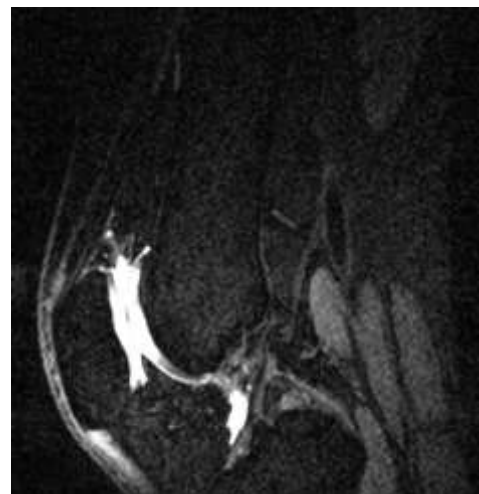


Fig. 4: a) A slice of the Knee dataset

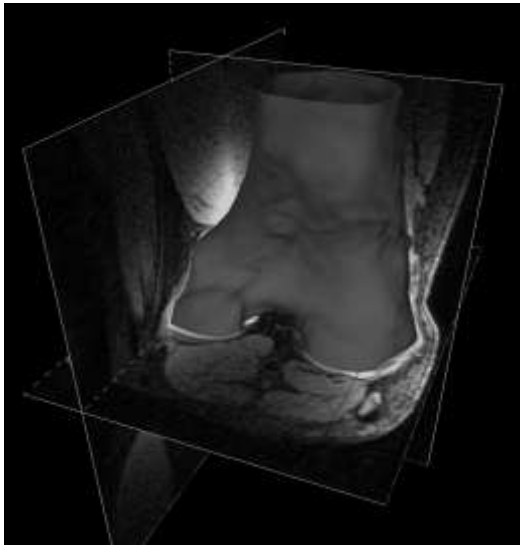


Fig 4: b) Segmentation of the femur in the Knee MRI dataset.

In general, the variational methods previously presented use gradient descent search for optimization which can only be guaranteed to find stationary points. For most cases, it is not known whether the solution found is a local minimum, maxima or a saddle point. Furthermore, the solution depends critically on the initial condition (curve). The main benefit of the combinatorial methods is that a global solution can be guaranteed. This has several advantages: First, the solution is not depending on a good initial condition, so the initial curve can be placed arbitrarily in the image. Second, the quality of the solution relates directly to the energy functional and the parameters controlling the functional, rather than the initialization or other numerical implications of the implementation. On the other hand, some applications might benefit from the possibility of inducing local optima by a controlled initialization, such as extracting a particular branch of a blood vessel tree. But combinatorial methods can in general be considered more robust due to the guaranteed global solution.

B. Dynamic programming and optimal paths

Boundary definition via dynamic programming can be formulated as a graph searching problem where the goal is to find the optimal path between a set of start nodes and a set of end nodes. Typical applications of the use of dynamic programming in boundary tracking problems are tracing borders of elongated objects like roads and rivers in aerial photographs and the segmentation of handwritten characters. Medical applications include the segmentation of spine boundaries and tracing vessel borders. To apply dynamic programming to find the boundary of a mass we notice that the shape of most masses is approximately circular. This circularity constraint is implemented by carrying out the calculations in polar space. For the transform we use a circular region of interest (ROI) with center $(\mu_x; \mu_y)$ and radius R. The center of the ROI defines the origin for the coordinate transform and should be within the suspect lesion. The radius should be chosen large enough to allow application of the algorithm to all masses of interest. We choose a radius of 2.4 cm. Figure 5(a) and 5(b) show the coordinate transform. All pixels inside a circular ROI in the original image are transformed to the polar ROI 5(b). The x-

axis in the polar image represents the angle from $-\pi$ to π , the y-axis the radius from 0 to R. The dynamic programming algorithm is applied to the polar ROI to find the optimal path from one of the pixels in the first column to one of the pixels in the last column. The optimal path is defined as the minimum cumulative cost path, where the cumulative cost of a path is the sum of the local costs of the pixels on the path. We will now describe the local cost.

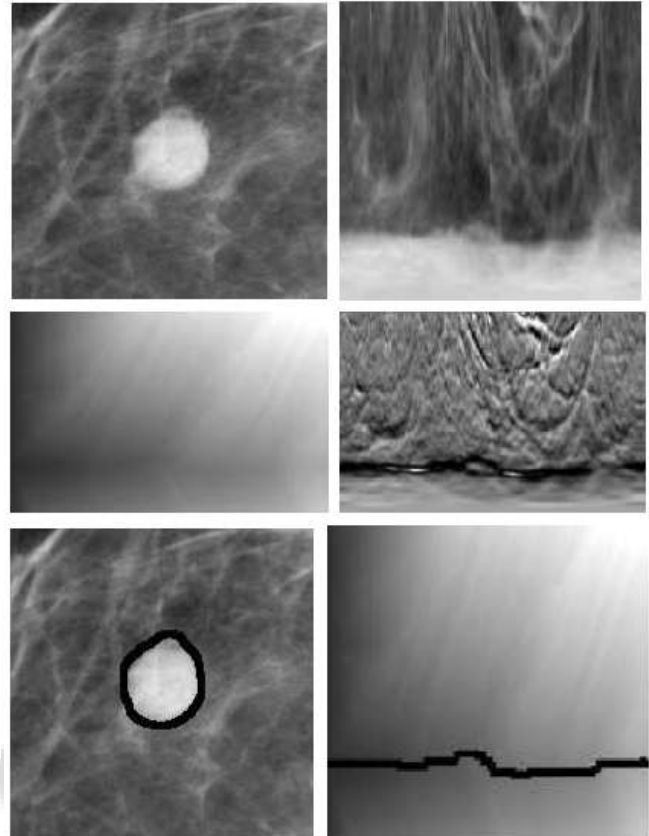


Fig 5: (a) Original ROI (b) Polar ROI (c) Cost matrix (d) Cumulative cost matrix (e) Path found by dynamic programming (f) Segmented mass

Figure 5: Programming algorithm for the segmentation of masses. 5(a) shows the ROI (region of interest) with a benign mass, 5(b) the ROI in polar coordinates. The cost of all pixels forms the cost matrix (5(c)). The dynamic programming algorithm calculates the optimal path in the cumulative cost matrix (5(d) en 5(e)). The final contour is shown in 5(f).

1) Local Cost

Local cost is cost assigned to each pixel in the polar image. This cost should embody a notion of a good boundary: pixels that possess many characteristics of the searched boundary are assigned low cost and vice versa. The local cost components form the following cost function:

$$c(i, j) = w_s s(i, j) + w_d d(i, j) + w_g g(i, j).$$

Eq (4)

Where s represents the edge strength, d the deviation from an expected size, and g the deviation from an expected gray level. The weights for the components are given by w_s , w_d and w_g .

Edge Strength S (i, j) as most contours exhibit strong edges we want to assign pixels with strong edge features low cost.

The edge strength for each pixel is determined by calculating the gradient magnitude in the direction normal to the contour. This corresponds with the gradient in vertical direction in the polar image. Then the relative edge strength is determined by normalizing the gradient values with the maximum gradient $\max(y')$. This normalization ensures that subtle contours with low global but high local edge strength can be found as well. The normalized gradient value is inverted so high gradients produce low costs and vice versa. The gradient component function is

$$s(i, j) = \frac{\max(y') - y'(i, j)}{\max(y')} \quad \text{Eq. (5)}$$

Where y' is the gradient magnitude in vertical direction for $\max(y')$ we took the 99th percentile of the gradient values measured in the ROI. By taking the 99th percentile it is prevented that one outlier, for instance a very bright micro-calcification, decreases the relative edge strength of all other pixels in the ROI.

Mass size $d(i, j)$ Contours that enclose a mass with a size common for masses are assigned low cost value. On the other hand, very small and very large segmentations are assigned higher cost. Most masses have a radius between 5 mm and 15 mm, with a mean radius of about 9 mm. The following formula was used to incorporate size information in the cost function:

$$d(i, j) = \begin{cases} (j - \mu)^2 & : j < m \\ (m - \mu)^2 & : j \geq m \end{cases} \quad \text{Eq. (6)}$$

where μ is the mean radius of masses. A cost limit m is set to prevent that the size component of the cost function completely determines the value of the cost function for large masses. This limit is set to 15 mm. alternatively; the cost component for mass size could be obtained by estimating the size distribution of masses. The probabilities for each size could then be used to determine the cost value $d(i, j)$. To use this method a large representative database with benign and malignant masses of known size is needed to accurately determine the probabilities for each mass size. Currently we used the first method as we do not have an independent database that we can use for this purpose.

Deviation from expected gray level $g(i, j)$ Another characteristic of the mass boundary is the gray level. This gray level should correspond with the edge of the object. A common assumption is that this edge is located at the zero crossing of the second derivative of the edge profile. In projective images however, the real edge is located toward the darker side (background). Consequently, the gray value of the border will have a value close to the background gray level. By estimating the intensity distribution of the mass and the background a preferred gray level for the contour can be determined.

$$g = \alpha \mu_{mass} + (1 - \alpha) \mu_{background} \quad \text{Eq. (7)}$$

Where μ should be smaller than 1/2 to ensure the edge is located more toward the background level.

The gray level component of the cost function is defined as

$$g(i, j) = \text{sqrt}(\text{abs}(G(i, j) - g)) \quad \text{Eq. (8)}$$

Where $G(i, j)$ is the intensity value of the pixel (i, j) . To estimate the gray level of the mass and the background we used two methods. The first method estimates μ_{mass} and $\mu_{background}$ as the mean gray level inside and outside an initial estimate of the contour. As initial estimate we took a circle with radius 0.6 cm. In the second method we used histogram analysis to estimate the gray level distributions. We found that most histograms of the ROI can be modeled reasonably well by a mixture of two Gaussian distributions, one narrow Gaussian in the low intensity range representing the fatty tissue, and a broader one in the middle/high intensity range representing dense tissue and/or masses. The parameters for the Gaussian distributions were estimated using the Levenberg-Marquardt method. The first peak in the histogram is used to estimate $\mu_{background}$, and the second peak to estimate μ_{mass} . If the histogram could not be modeled by a mixture of two Gaussians we used the first method to estimate the preferred gray level.

C. Dynamic programming path finding algorithm

Application of the cost function to all pixels in the polar image results in the so called cost image. This image can be seen as a graph in which the dynamic programming algorithm should find the path with the lowest cost. The first column in the cost image $c(i, 0)$ represents the start nodes for the algorithm, whereas the end nodes are represented by the pixels in the last column of the image. The cumulative cost of each path is stored in the cumulative cost matrix. The cumulative cost matrix is constructed in two steps. First the cumulative costs of pixels in the first column are set equal to the cost of these pixels:

$$C(i, 0) = c(i, 0), \quad \text{Eq. (9)}$$

Where $C(i, j)$ is the cumulative cost and $C(i, j)$ is the cost value for pixel (i, j) in the polar image. For the other pixels the cumulative cost is calculated by a recursive step:

$$C(i, j+1) = \min_{-m \leq l \leq m} C(i+l, j) + c(i, j+1) + h(l). \quad \text{Eq. (10)}$$

The additional cost of a segment of the path for column j to $j+1$ depends on the cost value of pixel (i, j) and the direction l . The cost of the direction is set according to a function $h(l)$ which we use to control smoothness. $h(l)$ is set to zero for directions outside the interval $[-2, \dots, 2]$. The cumulative cost matrix is shown in figure 5(d). The final contour is found by selecting those pixels that linked together from the boundary with the lowest cost. The endpoint $C(i, j)$ of the contour is the pixel in the last column of the cumulative cost matrix with the lowest cost. The optimal path is found by back tracing the path from the end pixel to one of the pixels in the first column (figure 5(e)). The optimal path in the polar image is transformed back to rectangular coordinates in the original image. The resulting segmentation is shown in figure 5(f). Final contour the dynamic programming algorithm does not guarantee the contour to be closed. In our application we consider a contour as closed if the distance between the start and the end points is less than 3 pixels. This is conform the smoothness function $h(l)$ (1) which is

zero outside the interval $[-2, \dots, 2]$. In most cases, especially if the mass is clearly visible, the program will correctly segment the mass by a closed contour. However, if the mass is vague or if other structures obscure the mass boundary, the segmentation program can fail to find a closed contour as the optimal path.

We designed a more efficient method to ensure that the resulting contour is closed. Our solution uses an extended cost matrix where costs are plotted in an interval from $-\pi$ to π . The extension factor determines the size of the extended cost matrix relative to the original cost matrix. The dynamic programming algorithm is used to find the optimal path in this extended cost matrix. The path from $-\pi$ to π is extracted from the extended cost matrix. This path represents the final contour. In the original cost matrix the final contour depended strongly on the initial angle of the polar coordinate transform and the resulting segmentation could be different for different initial angles. A disadvantage of this dependency is that image features near the boundaries of the interval from $-\pi$ to π could have undesired effects on the resulting contour. In the new method the dependence on the choosing initial angle is minimized and consequently the image features near the boundaries of the interval have less effect on the final contour. The avoidance of the discontinuities at $-\pi$ and π also leads to more closed contours.

To determine the efficiency of this method we set up the following experiment. For each extension factor we applied the dynamic programming algorithm to find the optimal contour. Afterward, we calculate for each extension factor the percentage of closed contours. The algorithm is considered efficient if the percentage of closed contours is nearly 100% for a small extension factor.

3.3 Graph cuts

An important contribution for segmentation based on *graph cuts* was presented by Boykov and Jolly [2001] (and extended in [Boykov and Funka-Lea, 2006]). In this work, pixels are classified as being either inside or outside the object, making this a combinatorial method using an implicit-type representation (albeit discrete). Thus, compared to the previously outlined path-based approaches, this method generalizes to higher dimensions more easily. The success of the method relies on an efficient min-cut/max-flow algorithm which computes a globally optimal cut.

First, we describe the basic terminology that pertains to graph cuts in the context of our segmentation method. An undirected graph $G = \langle V, E \rangle$ is defined as a set of nodes (vertices V) and a set of undirected edges (E) that connect these nodes. An example of a graph that we use in this paper is shown in Figure 1(b). Each edge $e \in E$ in the graph is assigned a non-negative weight (cost) w_e . There are also two special nodes called terminals. A cut is a

subset of edges $C \subset E$ such that the terminals become separated on the induced graph $G(C) = \langle V, E \setminus C \rangle$. It is normal in combinatorial optimization to define the cost of a cut as the sum of the costs of the edges that it severs

$$|C| = \sum_{e \in C} w_e. \tag{Eq. (11)}$$

Graph cut formalism is well suited for segmentation of images. In fact, it is completely appropriate for N-dimensional volumes. The nodes of the graph can represent pixels (or voxels) and the edges can represent any neighborhood relationship between the pixels. A cut partitions the nodes in the graph. As illustrated in Figure 1 (c-d), this partitioning corresponds to a segmentation of an underlying image or volume. A minimum cost cut generates a segmentation that is optimal in terms of properties that are built into the edge weights. Our technique is based on a well-known combinatorial optimization fact that a globally minimum cut of a graph with two terminals can be computed efficiently in low-order polynomial time. We show how to set a two terminal graph so that the minimum cut would give a segmentation that minimizes (1) among all segmentations satisfying the given hard constraints. It should be noted that graph cuts were used for image segmentation before.

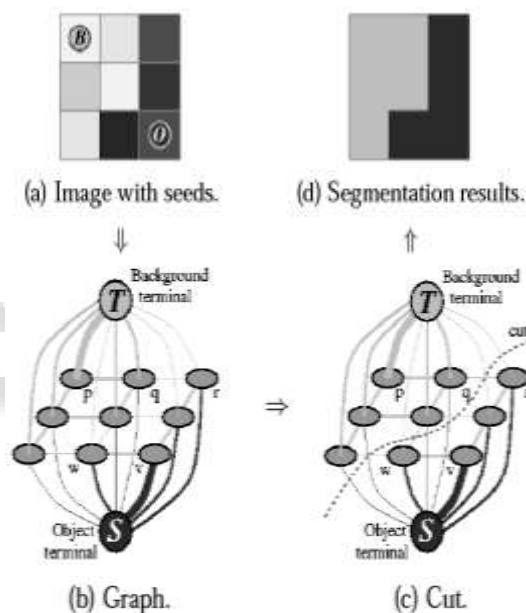


Fig. 6

A simple 2D segmentation example for a 3×3 image. The seeds are $O = \{v\}$ and $B = \{p\}$. The cost of each edge is reflected by the edge's thickness. The regional term (2) and hard constraints (4,5) define the costs of t links. The boundary term (3) defines the costs of n links. Inexpensive edges are attractive choices for the minimum cost cut.

In this formulation, however, the segmentation is strongly biased to very small segments. Shi and Malik [18] try to solve this problem by normalizing the cost of a cut. The resulting optimization problem is NP-hard and they use an approximation technique. Usually graph cuts are applied to minimize certain energy functions used in image restoration, stereo, 3D object reconstruction, and other problems in computer vision. A fast implementation of theoretically polynomial graph cut algorithms can be an issue. The most straight-forward implementations of the standard graph cut algorithms, e.g. max-flow or push-relabel, can be slow. The experiments done compare several well-known "tuned"

versions of these standard algorithms in the context of graph based methods in vision.

a) *Segmentation Technique*

In this section we provide algorithmic details about our segmentation technique. Assume that \mathcal{O} and \mathcal{B} denote the subsets of pixels marked as “object” and “background” seeds. Naturally, the subsets $\mathcal{O} \subset \mathcal{P}$ and $\mathcal{B} \subset \mathcal{P}$ are such that $\mathcal{O} \cap \mathcal{B} = \emptyset$. Remember that our goal is to compute the global minimum of (1) among all segmentations a satisfying hard constraint

$$\begin{aligned} \forall p \in \mathcal{O}, \quad A_p &= \text{“obj”} \\ \forall p \in \mathcal{B}, \quad A_p &= \text{“bkg”}. \end{aligned} \quad \text{Eq. (12)}$$

The general work flow is shown in Figure 6. Given an image (Figure 6(a)) we create a graph with two terminals (Figure 6(b)). The edge weights reflect the parameters in the regional (2) and the boundary (3) terms of the cost function, as well as the known positions of seeds in the image. The next step is to compute the globally optimal minimum cut (Figure 6(c)) separating two terminals. This cut gives segmentation (Figure 6(d)) of the original image. In the simplistic example of Figure 1 the image is divided into exactly one “object” and one “background” regions. In general, our segmentation method generates binary segmentation with arbitrary topological properties. We describe the details of the graph and prove that the obtained segmentation is optimal. To segment an given image we

create a graph $G = \langle \mathcal{V}, \mathcal{E} \rangle$ with nodes corresponding to pixels $p \in \mathcal{P}$ of the image. There are two additional nodes: an “object” terminal (a source S) and a “background” terminal (a sink T). Therefore,

$$\mathcal{V} = \mathcal{P} \cup \{S, T\}. \quad \text{Eq. (13)}$$

The set of edges E consists of two types of undirected edges: *n-links* (neighborhood links) and *t-links* (terminal links). Each pixel p has two t-links $\{p, S\}$ and $\{p, T\}$ connecting it to each terminal. Each pair of neighboring pixels $\{p, q\}$ in \mathcal{N} is connected by an n-link. Without introducing any ambiguity, an n-link connecting a pair of neighbors’ p and q will be denoted by $\{p, q\}$. Therefore,

$$\mathcal{E} = \mathcal{N} \cup \{\{p, S\}, \{p, T\}\}. \quad \text{Eq. (14)}$$

The following table gives weights of edges in \mathcal{E}

edge	weight (cost)	for
$\{p, q\}$	$B_{\{p,q\}}$	$\{p, q\} \in \mathcal{N}$
$\{p, S\}$	$\lambda \cdot R_p(\text{“bkg”})$	$p \in \mathcal{P}, p \notin \mathcal{O} \cup \mathcal{B}$
	K	$p \in \mathcal{O}$
$\{p, T\}$	0	$p \in \mathcal{B}$
	$\lambda \cdot R_p(\text{“obj”})$	$p \in \mathcal{P}, p \notin \mathcal{O} \cup \mathcal{B}$
$\{p, T\}$	0	$p \in \mathcal{O}$
	K	$p \in \mathcal{B}$

Table 1

Where

$$K = 1 + \max_{p \in \mathcal{P}} \sum_{q: \{p,q\} \in \mathcal{N}} B_{\{p,q\}}. \quad \text{Eq. (15)}$$

The graph G is now completely defined. We draw the segmentation boundary between the object and the background by finding the minimum cost cut on the graph

G. The minimum cost cut \hat{C} on G can be computed exactly in polynomial time via algorithms for two terminal graph cuts assuming that the edge weights specified in the table above are non-negative

To conclude this section we would like to show that our algorithm can efficiently adjust the segmentation to incorporate any additional seeds that the user might interactively add. To be specific, assume that a max-flow algorithm is used to determine the minimum cut on G. The max-flow algorithm gradually increases the flow sent from the source S to the sink T along the edges in G given their capacities (weights). Upon termination the maximum flow saturates the graph. The saturated edges correspond to the minimum cost cut on G giving us an optimal segmentation. Assume now that an optimal segmentation is already computed for some initial set of seeds. A user adds a new “object” seed to pixel p that was not previously assigned any seed. We need to change the costs for two t-links at p.

t-link	initial cost	new cost
$\{p, S\}$	$\lambda R_p(\text{“bkg”})$	K
$\{p, T\}$	$\lambda R_p(\text{“obj”})$	0

Table 2

And then compute the maximum flow (minimum cut) on the new graph. In fact, we can start from the flow found at the end of initial computation. The only problem is that reassignment of edge weights as above reduces capacities of some edges. If there is a flow through such an edge then we may break the flow consistency. Increasing an edge capacity, on the other hand, is never a problem. Then, we can solve the problem as follows. To accommodate the new “object” seed at pixel p we increase the t-links weights according to the table

t-link	initial cost	add	new cost
$\{p, S\}$	$\lambda R_p(\text{“bkg”})$	$K + \lambda R_p(\text{“obj”})$	$K + c_p$
$\{p, T\}$	$\lambda R_p(\text{“obj”})$	$\lambda R_p(\text{“bkg”})$	c_p

Table 3

These new costs are consistent with the edge weight table for pixels in O since the extra constant C_p at both t-links of a pixel does not change the optimal cut⁴. Then, a maximum flow (minimum cut) on a new graph can be efficiently obtained starting from the previous flow without re-computing the whole solution from scratch. Note that the same trick can be done to adjust the segmentation when a new “background” seed is added or when a seed is deleted. One has to figure the right amounts that have to be added to the costs of two t-links at the corresponding pixel. The new costs should be consistent with the edge weight table plus or minus the same constant.

IV. CONCLUSION

In conclusion, the role of segmentation is to subdivide the objects in an image; in case of medical image segmentation the method should help with following

- 1) Study anatomical structure
- 2) Identify Region of Interest i.e. locate tumor, lesion and other abnormalities
- 3) Measure tissue volume to measure growth of tumor (also decrease in size of tumor with treatment)
- 4) Help in treatment planning prior to radiation therapy; in radiation dose calculation

Automatic segmentation of medical images is a difficult task as medical images are complex in nature and rarely have any simple linear feature. Further, the output of segmentation algorithm is affected due to partial volume effect, intensity in homogeneity, presence of artifacts; closeness in gray level of different soft tissue the major problem with the current level set approach for segmentation is the long execution time, especially for volumetric images. Because of this, tweaking the parameters, e.g. for regularization, can be cumbersome. This is typical for level set methods and can be improved somewhat with optimized implementations. However, since the segmentation model is basically formulated as a “soft threshold” on the zero-crossing of the filter output, applying the full level set framework is likely overkill. Thus, future work includes investigating alternative, probably simpler, methods for segmentation based on the filter output.

REFERENCES

- [1] Markus Unger, Thomas Pock, and Horst Bischof, Continuous Globally Optimal Image Segmentation with Local Constraints, Institute for Computer Graphics and Vision, Graz University of Technology, Austria
- [2] M. Sonka and J. M. Fitzpatrick, editors. Handbook of Medical Imaging -Volume 2. Medical Image Processing and Analysis. SPIE Press, Bellingham, Washington, USA, 2000.
- [3] Anthony Yezzi, Jr., Member, IEEE, Satyanad Kichenassamy, Arun Kumar, Peter Olver, and Allen Tannenbaum, A Geometric Snake Model for Segmentation of Medical Imagery, IEEE TRANSACTIONS ON MEDICAL IMAGING, VOL. 16, NO. 2, APRIL 1997
- [4] Gunnar L  th  n, Segmentation methods for medical image analysis Blood vessels, multi-scale filtering and level set methods, Thesis No. 1434, Link  ping studies in science and technology
- [5] Yuri Y. Boykov Marie-Pierre Jolly, Interactive Graph Cuts for Optimal Boundary & Region Segmentation of Objects in N-D Images, Proceedings of “International Conference on Computer Vision”, Vancouver, Canada, July 2001
- [5] R.Yogamangalam, B.Karthikeyan, Segmentation Techniques Comparison in Image Processing, International Journal of Engineering and Technology (IJET), Vol 5 No 1 Feb-Mar 2013
- [6] Sheila Timp, Nico Karssemeijer, A new 2D segmentation method based on dynamic programming applied to computer aided detection in

- mammography. Department of Radiology, University Medical Center Nijmegen, the Netherlands
- [7] Automated medical image segmentation techniques neeraj Sharma And Lalit M Aggarwal, Automated Medical Image Segmentation Techniques, Journal Of Medical Physics, Vol 35, Jan-Mar 2010 Automated medical image segmentation techniques