

Three-Dimensional Medical Image Segmentation Using a Graph-Theoretic Energy-Minimisation Approach

Brian Parker

Biomedical & Multimedia Information Technology (BMIT) Group
School of Information Technologies, University of Sydney, Australia

brianp@cs.usyd.edu.au

Abstract

A new graph algorithm for multiscale segmentation of three-dimensional medical data sets is presented. It is a three-dimensional generalisation of an existing two-dimensional Mumford-Shah region-merging segmentation algorithm. The Mumford-Shah functional formulation leads to improved segmentation results compared to alternative approaches; the graph theoretic approach yields improved performance and simplified data structures; and the automated stopping estimation allows a fully non-supervised algorithm with no tuning parameters required (except for an optional parameter to select the depth of segmentation).

1 Introduction

Segmentation is a fundamental low-level operation for visualization and automated analysis and diagnosis of medical images.

Energy-minimisation approaches have been shown to be an effective segmentation scheme. In this approach, an explicit energy is defined by a functional over the set of possible segmentations and the segmentation with the minimum energy is, in theory, selected. In some sense, this framework can be viewed as the most general approach and other, ad hoc, schemes can be viewed as specializations in this framework (Morel and Solimini 1995). The Mumford-Shah functional (described in the sequel) is the most popular such functional.

The problem that needs to be solved in applying this framework in practice is to approximate a numerical solution tractably within this energy-minimisation framework. Several approaches have been applied to this end, including variants of standard numerical optimization routines such as gradient descent, but these suffer from speed problems for large 3D data sets.

In Koepfler, Lopez, and Morel (1994), an efficient algorithm for the solution of a simplified Mumford-Shah functional for 2D images is presented using a pure region-merging approach using only local operations. In this case, local neighbours of minimal energy are merged. (Rudin (1993) briefly discusses extending this algorithm to 3D). Unfortunately, this algorithm uses an iterative approach that requires the user to select, as a parameter, an increasing lambda parameter of the Mumford-Shah

functional for each iteration. The segmentation results are very sensitive to this schedule of parameters, and no technique to automate this is described.

In Redding, Crisp, Tang, and Newsam (1999), a so-called “full-lambda schedule” variant of Koepfler (1994) for 2D images is described which avoids the need to select a vector of lambda parameters and instead has only a single stopping parameter.

In this paper, the algorithm of Redding et al (1999) is generalized in several ways, including recasting it in a graph-theoretic formulation, and applying it to the problem of three-dimensional medical data set segmentation.

2 Algorithm Description

First is described the underlying 2D segmentation theory, then the algorithm itself and its generalisation to 3D segmentation.

2.1 Two-dimensional Mumford-Shah region-merging segmentation theory

As described in Koepfler et al (1994), a variational formulation of the 2D image segmentation problem using a Mumford-Shah functional has been found to give high-quality segmentation results. The input image \mathbf{g} is a (potentially) vector-valued function with m components, defined over a rectangular grid of pixels \mathbf{x} . For example, in colour image segmentation, $\mathbf{g}(\mathbf{x})$ may be a 3-vector in some colour space, but for MRI or CT data, $\mathbf{g}(\mathbf{x})$ is typically a scalar-valued function

The segmentation is obtained by minimising the simplest Mumford-Shah functional

$$E(\mathbf{u}, B) = \sum_i E_i(\mathbf{u}, \mathbf{g}) + \lambda l(B) \quad (1)$$

where B is a set of boundaries with total length $l(B)$ defining the segmentation, \mathbf{u} is an approximating function, E_i denotes the approximation error over region V_i , and λ is a regularisation parameter. Intuitively, the first term limits the error in image approximation resulting from segmentation, and λ controls the fineness of the segmentation by limiting the total boundary length of the segmentation. The approximating function $\mathbf{u}(\mathbf{x})$ is piecewise constant with value \mathbf{u}_i over each region V_i . The approximation error over region V_i is given by

$$E_i(\mathbf{u}, \mathbf{g}) = \sum_{\mathbf{x} \in V_i} (\mathbf{g}(\mathbf{x}) - \mathbf{u}_i)^T (\mathbf{g}(\mathbf{x}) - \mathbf{u}_i) \quad (2)$$

The minimising argument $\bar{\mathbf{u}}_i$ is simply the mean of \mathbf{g} over V_i . The corresponding minimum approximation error (residual) is denoted by \bar{E}_i .

The underlying segmentation strategy is based on the so-called full- λ -schedule 2D segmentation algorithm of Redding et al (1999). By viewing the problem in a graph theoretic formulation, the ad hoc data structures described therein are simplified and improved- in particular, an adjacency list graph representation is used with a Fibonacci heap to order the edges. This improved graph theoretic formulation is described here.

As in Redding (1999), the decision to merge adjacent region pairs (V_i, V_j) occurs when λ exceeds the merging cost $t_{i,j}$ given by

$$t_{i,j} = \frac{\bar{E}_{ij} - (\bar{E}_i + \bar{E}_j)}{l(\partial(V_i, V_j))} \quad (3)$$

for a monotonically increasing λ -schedule. The numerator is the increase in approximation error due to the merge, and the denominator is the length of the common boundary. Using (2), the numerator of $t_{i,j}$ may be shown to evaluate to a simple area-weighted Euclidean distance:

$$t_{i,j} = \frac{\frac{|V_i||V_j|}{|V_i| + |V_j|} \|\bar{\mathbf{u}}_i - \bar{\mathbf{u}}_j\|^2}{l(\partial(V_i, V_j))} \quad (4)$$

where $|V_i|$ is the area of region V_i .

The full- λ -schedule segmentor iteratively performs a global search over all region pairs for the smallest value of $t_{i,j}$, merging the corresponding regions V_i and V_j .

Let the segmented image be represented as a simple, undirected, weighted graph $G(V, B)$, where vertices V represent regions, and the edges B represent the boundaries between regions. A region V_i is the pair $(\bar{\mathbf{u}}_i, |V_i|)$. The edge weights are $l(\partial(V_i, V_j))$. The following algorithm can be used to find a local minimum for functional (1):

1. Populate graph G with the trivial segmentation of the initial image data i.e. add each pixel to G as a separate region, with the four direct neighbours separated by boundaries of length one.
2. Remove the region boundary B_{ij} from G that has the minimum $t_{i,j}$ as calculated by (4).
3. Contract the two regions V_i and V_j to form merged region V_{ij} , where $|V_{ij}| = |V_i| + |V_j|$ and $\bar{\mathbf{u}}_{ij}$ is the area-weighted average of $\bar{\mathbf{u}}_i$ and $\bar{\mathbf{u}}_j$ i.e. $\bar{\mathbf{u}}_{ij} = (\bar{\mathbf{u}}_i|V_i| + \bar{\mathbf{u}}_j|V_j|) / (|V_i| + |V_j|)$.
4. If any multi-edges have been introduced by the merge in step 3, merge the repeated edges to again

make G simple, setting the length of the merged boundary to the combined length of the two constituent boundaries.

5. Repeat steps 2 to 4 until terminating according to some stopping rule (see sec. 2.4).

2.2 Two-dimensional Mumford-Shah region-merging segmentation implementation

To efficiently implement the algorithm described above, graph G is implemented using an adjacency list representation, with a Fibonacci heap (or similar priority queue) used to store and order the boundaries B_{ij} . Fig. 1 gives a block diagram of the complete data structure.

The regions V_i are stored in a list L . The regions V_i themselves are represented as a structure that has members for $\bar{\mathbf{u}}_i$ and $|V_i|$ as described in sec 2.1. It also

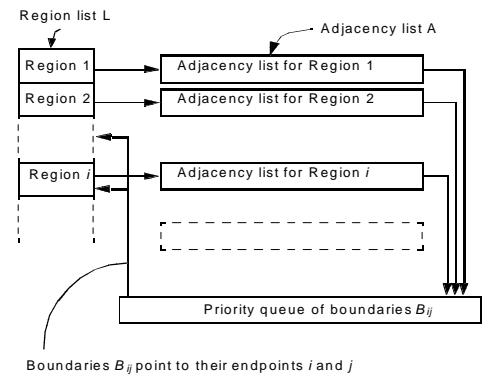


Figure 1: Data structure.

has as a member an adjacency list A_i of pointers to the boundaries B that are incident on V_i , which defines the graph structure. It also stores a list of the pixels defining the region. (Note that the list of pixels is updated during the merging step 3, but is never itself used in the merging calculations and is provided only for later processing of the segmented result).

The boundaries B_{ij} are represented by a structure that, in addition to the edge weight $l(\partial(V_i, V_j))$ as described in sec 2.1, also stores pointers to the endpoint regions V_i and V_j , and the calculated merging cost $t_{i,j}$ for the boundary, to enable fast dynamic modifications to the graph data structure.

The Fibonacci heap is initially sorted on priority order of $t_{i,j}$, as defined by (4), when G is first populated. After steps 3 and 4 above, the Fibonacci heap is updated with the new merging costs $t_{k,ij}$ of all boundaries $B_{k,ij}$ incident on region V_{ij} , causing the heap to reorder. This efficiently maintains the boundaries in priority order on $t_{i,j}$, allowing step 2 to be performed quickly.

Access to the priority queue's minimum element is $O(\log(N))$, so it can be shown, with minor changes to the algorithm analysis in Redding et al (1999), that the algorithm is $O(N \log N)$, where N is the number of pixels or voxels.

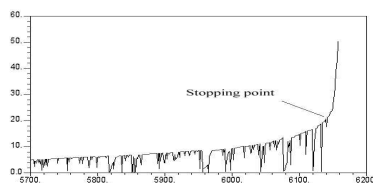
2.3 Extension to three-dimensional segmentation

From this description of the segmentation algorithm as an abstract graph algorithm, it can be seen that, after the graph is first populated, the dimensionality of the original data is not used again. Therefore, the algorithm can be straightforwardly generalised to a 3D segmentor by simply populating the graph with voxels (the 3D analogue of pixels) and the corresponding six direct neighbours in 3D rather than with 2D pixels and the corresponding four direct 2D neighbours as described in sec 2.1.

2.4 Stopping rule

Step 5 of the algorithm (sec 2.1) requires a rule for defining when to halt the segmentation process. There are several stopping rules that are useful in different contexts e.g. empirically determining a final $t_{i,j}$, or halting when a certain fixed number of segments is reached. These suffer from the disadvantage that they are not adaptive to changing image data.

It can be noted that as boundary elements are removed in priority order, the plot of $t_{i,j}$ vs n (where n is the total number of boundaries removed) trends upward but has numerous local energy minima representing the merging of homogeneous regions. The $t_{i,j}$ value directly after the last such significant local minimum is a good indicator that all significant regions in the data set have been segmented, and can be used as a robust default for an automatically determined stopping point; explicitly specifying a lower stopping $t_{i,j}$ allows a multiscale segmentation to be produced. See Fig. 2.



2.5 Optional 3D initialisation preprocessor

As described above, when populating the graph with new data, we start with the trivial segmentation whereby every voxel is its own region. This can lead to a large initial memory cost. This can be lessened by initialising using a tiled octree decomposition of the newly added frame data. In this case, cubic "tiles" of $M \times M \times M$ voxels are added in scan-line order, where M is some fixed small power of two, such as 16. For each cube of $M \times M \times M$ voxels it is determined whether the voxels have the same colour within a predetermined small tolerance. If this condition is met, the cube is added to the graph in its entirety as a single region; otherwise the cube is subdivided into eight equally sized smaller blocks. Each of these smaller blocks are similarly tested for homogeneity, and added as a single region if so, otherwise the process is continued recursively until single voxel regions are reached.

3 Results

The results of segmenting a $256 \times 256 \times 10$ MRI data set are shown at two depths of segmentation.

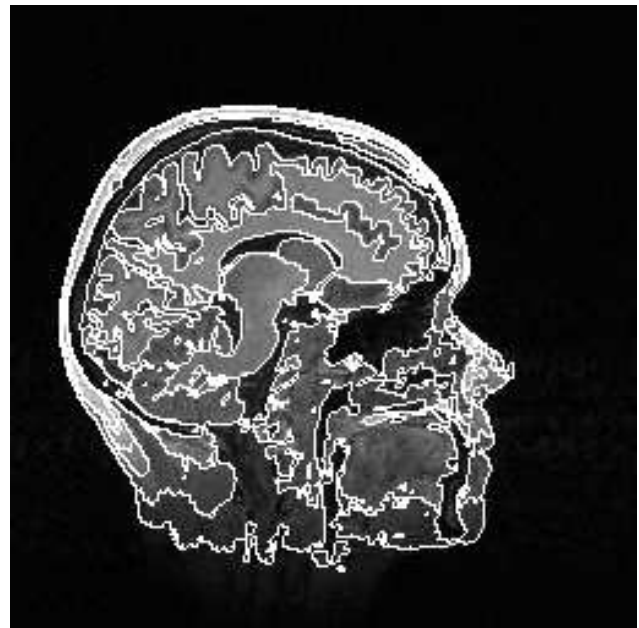


Figure 2



Figure 3

Figure 2 shows a single cross section of the 3D segmentation at high depth of segmentation; at this scale individual structures have been segmented out e.g. cerebellum, brain stem.

Segmentation time was 12 minutes on a Pentium 4 1.7 GHz. Memory usage was approximately 0.5 GBytes.

Figure 3 shows a cross section at a lower depth of 3D segmentation; individual structures are not segmented out at this scale.

4 Conclusion

A new graph algorithm for unsupervised 3D medical image segmentation based on Mumford-Shah energy minimization has been presented.

The advantages of this algorithm include-

- (1) The graph-theoretic abstraction of the basic approach of Redding et al (1999) leads directly to a more efficient implementation using a priority queue, and a simpler underlying data structure.
- (2) The graph-theoretic formulation allows a straightforward extension to three-dimensional segmentation.
- (3) An automatic algorithm for deducing the single stopping parameter of Redding et al (1999) is provided- there are no arbitrary algorithm parameters that require empirical determination.
- (4) The underlying Mumford-Shah energy minimization provides a high quality segmentation due to the variational formulation which explicitly incorporates both region homogeneity and compactness criteria.
- (5) Multiscale segmentations can be produced by explicitly varying the stopping level, and the algorithm applies to arbitrary vector data as per Kopfler et al (1994).

5 Future Work

A problem that remains is that, even though the algorithm is $O(N\log N)$ where N is the number of voxels, three-dimensional data sets have a cubic number of voxels for a given sampling density and so segmenting huge data sets (e.g. whole body scans) is intractable in space and time. Further extensions to the described algorithm to handle this problem are to be described in a future paper.

6 References

- MOREL, J., SOLIMINI, S. (1995): *Variational Methods in Image Segmentation*. Boston, Birkhauser.
- KOEPFLER, G., LOPEZ, C., MOREL, J.M. (1994): A multiscale algorithm for image segmentation by variational method. *SIAM J. Numerical Analysis*, Vol. 31 No. 1, pp. 282-299.
- REDDING, N.J., CRISP, D.J., TANG, D.H. and NEWSAM, G.N. (1999): An efficient algorithm for Mumford-Shah segmentation and its application to SAR imagery, *Proc. Conf. Digital Image Computing Techniques and applications (DICTA 99)*, pp. 35-41.
- RUDIN, L., NORDBY, F., MOREL, J. (1993): Fast variational algorithm for clutter removal through pyramidal domain decomposition. *Proc. SPIE Vol. 2037*, pp. 2-22.