Split-and-Merge Segmentation using Octrees

Tim Weingärtner and Rüdiger Dillmann

Institute for Real-Time Computer Systems & Robotics
Prof. Dr. U. Rembold, Prof. Dr. R. Dillmann
University of Karlsruhe
Department for Computer Science
76128 Karlsruhe, Germany
weinga@ira.uka.de

Abstract

This paper presents an algorithm for the segmentation of 3D datasets which especially occur in medical imaging. A combined split and merge segmentation is used to extract homogeneous regions out off the original image. During this automatic segmentation process an octree is build up and the resulting regions are saved in an octree like structure. This method can deal with large data volumes due to the fact that a compressible data structure is used.

1 Introduction

The segmentation of image data is the process of deciding which parts of the data correspond to real objects. This function is one of the main problems in computer graphics today. The segmentation of 3D datasets as they especially occur in medical imaging brings up many unsolved problems. One of these is the immense data volume required for one 3D image. The more the technology of the tomographic devices is developed, the more the size of the image increases. Today, Computer Tomographs are reaching an image resolution of 512∗512 to $1024 * 1024$ voxels per slice meaning more than 100 MB per 3D image.

The basic idea of this paper is to apply knowledge of data compression into the process of segmentation, and to reduce the size of the resulting regions. We describe the results using a region based segmentation method in combination with octrees. After a brief description of the data structures used and some optimizations, we present a split-and-merge algorithm based on and matched with the previously mentioned structures. Finally, we show our experiments and results reached by segmentation of test images and MRT images.

2 Segmentation

In this chapter, the approach to segmentation of tomographic images is described [5]. Choosing a three dimensional attempt enables the inclusion of the information of the gray value difference between different slices. This information could increase the probability of finding whole regions and not getting stuck in local disturbances.

Segmentation could be divided into three basic techniques: *point-based segmentation, edge-based segmentation*, and *region-based segmentation*. A method out of the region-based class was used due to the fact that in many cases the gray value of a voxel in a tomographic image does not correspond directly with its belonging to a specific structure. The main causes for this are disturbances which mainly arise in MRT images and result from blood flow, motions and low resolution. Region-based segmentation enables the reduction of the influence of such disturbances.

One of the main techniques in region-based segmentation is split-and-merge segmentation. It combines the advantages of simple region splitting and region growing. This kind of segmentation consists of two different parts. The split-process divides the image into small homogeneous regions. In the second process neighbouring regions meeting a homogeneous criterion are merged into larger regions. The use of octrees enables the support of this process and the compression of the data volume at the same time.

Before describing the algorithm we take a closer look at the data structures used because they build the main foundation for our method and understanding them makes it easier to understand the whole process.

2.1 Data structures

A well known method of data compression in computer graphics is the use of octrees [3]. This hierarchical data structure is based on a recursive subdivision of a

cube. The subdivision determines whether the subcube has satisfied a homogeneous criterion or has reached the maximal resolution. Therefore octrees allow a condensed storage of 3D images and are mainly used in computer graphics and robotics.

The former definition of octrees contains only three states for each node of the octree:

- the node *belongs to* the represented region (**black**);
- the node *does not belong to* the represented region (**white**); or
- the node must be *divided* further (**gray**).

This binary definition could be extended by appending individual values, like the Houndsfield number (the density factor of tissue measured by a CT) or the T1 relaxation, to the nodes. Even with the "individuality" of the nodes, the rate of compression is sufficiently large.

Even this good rate of compression could be extended using linear octrees [1]. This data structure only stores the black nodes of an octree and therefore reduces the required space corresponding to the information stored in the image. Each node is stored by its "code" in an array and could be reached directly. The octree-code can be calculated straight from the cartesian coordinates in the image and correspondes to the path from the root to the node in the original octree. Linear octrees also make it easy to deal with neighbourhood requests because the six direct neighbours can be calculated straight out of the octree-code. These advantages make the linear octree one alternative to regular octrees.

After this brief introduction of the data structures used we will now present the main part of our algorithm. First we explain the split-process dividing the 3D data volume into homogeneous regions. Then we describe the merge-process which melds the regions belonging together, calculated in the first part.

2.2 Split-process

The split-process works recursively by dividing the data cube into octants which are registered in an octree. The subdivision ends if a homogeneous criterion is fulfilled. This method makes it possible to build the octree parallel to the split-process and therefore works very efficiently.

In our algorithm four homogeneous criteria (1)-(4) were implemented and tested with different CT and MRT images. The used criteria could be classified into:

Gradient conditions: A good measure for edge detection is the gray value gradient compared with a given threshold. The following condition could be formulated as a homogeneous criterion: *"There is no large gradient (edge) allowed in a homogeneous region"* (see Formula (1)+(2) at the end of this paper). This criterion works well especially in cases of small disturbances where edges are only represented by large gradient values.

Statistical conditions: In a different approach, all gray values in a cube are examined together. Calculating the gray value range or the variance gives information about the global differences in a cube. (see Formula (3)+(4)) These methodes enable the detection of continous transitions but produce more regions than gradient methods.

After applying the split-process, the 3D data volume had been divided into many small homogeneous regions stored in one global octree. Even the good compression rate of an octree can not deal with the problem of the large data volumes occuring in medical imaging. Solving this problem led to two solutions:

- Splitting only one octant at a time enables to reduce the storage space by factor 8. Due to the fact that the resulting regions are stored in an octree this method can be used without further expense. Another positive side effect of this method is the easy parallelization of the split-process.

- Using the previously mentioned linear octrees for further compression of the resulting homogeneous regions and employing them in the merge-process reduces the storage space once more.

2.3 Merge-process

The second part of the segmentation process merges the different regions resulting from the split-process into larger structures corresponding to anatomic tissue. Small homogeneous regions are melted together in a region growing process starting from random regions and continuing with the direct neighbours of each region. The difference between the arithmetic mean of two regions is chosen as melting condition (see Formula (5)).

The neighbourhood of a region consists of the six direct neighbours (see Fig. 1). This arrangement was chosen because using only the direct neighbours avoid a breakout through small borders during the merge-process.

2.4 Optimization

The merge-process can be optimized using a merge-index [2]. This index is a measure of correspondence

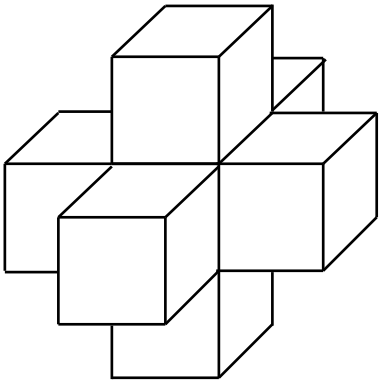

Figure 1: Direct neighbours

between two regions. It depends on three sizes:

- mean gray value and variance:
  $S_{Sim}(R_i, R_j) = \left|\mu_{R_i} - \mu_{R_j}\right| / \max(1, \sigma_{R_i} + \sigma_{R_j})$

- size of the regions and their differance:
  $S_{Size}(R_i, R_j) =$
  $\min(2, \min(size(R_i), size(R_j)))/1000$

- distance between two regions:
  $S_{Conn}(R_i, R_j) =$
  $\min(\text{distance between } R_i \text{ and } R_j)$

These three conditions are weighted differently and combined in a global merge-index making it possible to drastically reduce the number of small regions. This is due to the fact that the index is small even for regions with larger gray value difference and large difference of size. Those arrangement arise from disturbances in the original image and could not be handled with a ordinary region growing algorithm.

3 Experiments and Results

As testbed for the algorithm first four different test images with and without disturbances were used. Fig. 2 shows one of these images. They combine large structures (spheres) including random disturbance and thin structures (lines) crossing one another. Problems arising during the segmentation of MRT images could be simulated with this method.

After successfully segmenting these test images, different kinds of CT and MRT images with resolutions of up to 256^3 were used. Fig. 3 + 4 shows one of these results. For this image the adaptive gray level gradient (2) was used with a split-threshold of $\varepsilon = 70$ and a merge-threshold of $\lambda = 40$. The originally 256 gray values were restricted to 120. After the split-process the octree contained 260774 endnodes. These were melted to 246 homogeneous regions during the merge-process. The optimized merge-process further reduced this number.

4 Conclusion

We have presented an automatic 3D split-and-merge segmentation algorithm using an octree data structure for performance and storage space optimization. The split-process includes four different splitting conditions and could be easily parallelized because of the splitting scheme. In the second part of the segmentation we showed the merge-process optimized with a merge-index combining the gray value, the size of the regions and their distance.

Results have been shown for the segmentation of a test image and an MRT image.

Acknowledgment

This work was performed at the Institute for Real-Time Computer Systems & Robotics, Prof. Dr.-Ing. U. Rembold and Prof. Dr.-Ing. R. Dillmann, Department of Computer Science, University of Karlsruhe, 76128 Karlsruhe, Germany.

The medical images were provided by the Clinic for Mouth-,Tooth- and Jaw-Surgery at the University of Heidelberg.

References

[1] I. Gargantini. Linear octtrees for fast processing of three-dimensional objects. *Computer Graphics and Image Processing*, 20:365–374, 1982.

[2] G. Manos, A.Y. Cairns, I.W. Ricketts, and D. Sinclair. Automatic segmentation of hand-wrist radiographs. *Image and Vision Computing*, 11(2):100–111, March 1993.

[3] D. Meagher. Geometric modeling using octree encoding. *Computer Graphics and Image Processing*, 19:129–147, 1982.

[4] U. Tiede, K.H. Höhne, M. Bomans, A. Pommert, M. Riemer, and G. Wiebecke. Investigation of medical 3d-rendering algorithms. *IEEE Computer Graphics & Applications*, pages 41–53, March 1990.

[5] T. Weingärtner. Untersuchung von volumenorientierten Darstellungen aus Tomographiebildern. Master's thesis, IPR, Universität Karlsruhe, July 1994.

Figure 2: Slice of the test image (Left: Original image; right: Segmented image)

Figure 3: MRT image (Slice 60; left: Original image; right: Segmented image)

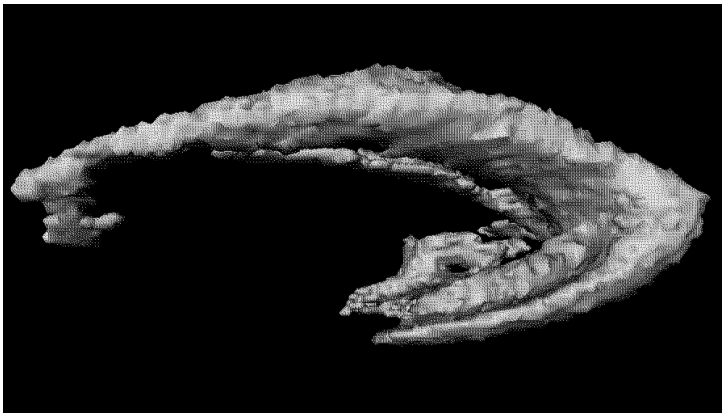

Figure 4: Segmented Thalamus (Right Part)

Formulae

$$C_{s1} = \begin{cases} SPLIT, \text{if } \forall x,y,z \in Cube_i | G(x) \leq \varepsilon \wedge G(y) \leq \varepsilon \wedge G(z) \leq \varepsilon \\ STOP \text{ , else} \end{cases} \tag{1}$$

G = gray level gradient; ε = threshold

$$C_{s2} = \begin{cases} SPLIT, \text{if } \forall x,y,z \in Cube_i | G^*(x) \leq \varepsilon \wedge G^*(y) \leq \varepsilon \wedge G^*(z) \leq \varepsilon \\ STOP \text{ , else} \end{cases} \tag{2}$$

G^* = adaptive gray level gradient (AGG) [4]

$$C_{s3} = \begin{cases} SPLIT, \text{if} \left[\begin{array}{l} \max_{(x,y,z) \in Cube_i}(f(x,y,z)) - \\ \min_{(x,y,z) \in Cube_i}(f(x,y,z)) \end{array} \right] \leq \varepsilon \\ STOP \text{ , else} \end{cases} \tag{3}$$

$$C_{s4} = \begin{cases} SPLIT, \text{if} \left[\begin{array}{l} \left[\max_{(x,y,z) \in Cube_i}(f(x,y,z)) - \right. \\ \left. \text{mean}_{(x,y,z) \in Cube_i}(f(x,y,z))\right] \leq \frac{\varepsilon}{2} \wedge \\ \left[\text{mean}_{(x,y,z) \in Cube_i}(f(x,y,z)) - \right. \\ \left. \min_{(x,y,z) \in Cube_i}(f(x,y,z))\right] \leq \frac{\varepsilon}{2} \end{array} \right] \\ STOP \text{ , else} \end{cases} \tag{4}$$

$$C_{m1}(R_i \cup R_j) = \begin{cases} MERGE, \text{if} \left\| \sum_{(x,y,z) \in R_i} \frac{f(x,y,z)}{Size(R_i)} - \sum_{(x',y',z') \in R_j} \frac{f(x',y',z')}{Size(R_j)} \right\| \leq \lambda \\ STOP \quad \text{, else} \end{cases} \tag{5}$$

λ = threshold