volumetric and tomographic data,
watershed transform,
region based volume processing

Michal HUČKO[1], Miloš ŠRÁMEK[2]

REGION-BASED PROCESSING OF VOLUMETRIC DATA

Measurement of volumetric tomographic data, similar to other measurement techniques, suffers from several classes of artifacts, of which noise presence and the partial volume effect belong to the most prominent ones. These artifacts spoil data analysis and/or visualization, which may, for example in the case of medical imaging, lead to erroneous decisions with severe consequences.

We propose a set of tools for region-based processing of volumetric data. Here, the basic entity is a spectrally homogeneous region instead of the traditional voxel. This provides for, on the one hand, higher robustness and, on the other hand, speeds up processing owing to many times smaller amount of elements to work with. Homogeneous regions are in our approach detected by the well-known segmentation by means of the watershed transform. In this paper we present algorithms for streamed computation of watershed transform, which allows for processing of very large data, region-based data smoothing and region merging based on the spectral distance. Further, we present an interactive tool for volume data segmentation and visualization which takes advantage of region hierarchies obtained by a hierarchical watershed transform.

1. INTRODUCTION

Today's advanced medical tomographic scanners in a daily routine produce high resolution three and even four dimensional data sets, the size of which often reaches hundreds of megabytes and even gigabytes. Data sets of this size, in order to provide the recipient (in this case usually a radiologist) with appropriate information, should usually be processed by state-of-the-art analysis and visualization techniques [2]. In this sense numerous image enhancement, smoothing, denoising, analysis and segmentation techniques exist and several popular software packages were developed [7, 4]. Our group at the Austrian Academy of Sciences in Vienna and Komenský University in Bratislava has in this area contributed by the development of a process- and slice-based streaming technique for processing of volumetric data [9, 10]. In slice-based streaming, rather than to store the whole volume in the main memory, only its part is loaded, the size of which is specified by the algorithm—in the case of point operations only one slice of data is required and in the case of local operations the number of slices is specified by the size of the operator kernel. The technique extends also to numerous global operations, but several passes over the data are then required with intermediate out-of-core (disk) data buffering. We implemented the technique in a set of single-purpose and usually command-line tools[3], which can be easily concatenated by connecting their inputs and outputs to complex processing networks. The side effect of this arrangement is that in this way parallelism of today's multicore/multiprocessor computers can be utilized without the necessity of specialized tool parallelization.

In this paper we present a set of algorithms and tools which support region-based processing of volumetric data. Here, the basic entity for processing is a spectrally homogeneous region instead of the traditional voxel. This provides for, on the one hand, higher robustness owing to the possibility to suppress noise and, on the other hand, enables to speed up processing since the number of regions is usually many times smaller then the number of voxels.

Homogeneous regions are in our approach detected by the well-known segmentation by means of the watershed transform [8]. Here we present two streamed algorithms for its computation (Section 2), a single-pass one which buffers a varying number of slices and a multi-pass one which needs to buffer

[1] Comenius University, Bratislava, Slovakia,
email: michal.hucko@fmph.uniba.sk.
[2] Slovakia, Austrian Academy of Sciences, Vienna, Austria,
email: milos.sramek@oeaw.ac.at.
[3] The open source f3d package, freely available at http://sourceforge.net/projects/f3d.

just three slices. Next, we present two techniques for low-level region based processing (Section 3). The first one is simple region-based smoothing which provides for edge sharpening. The second one enables to reduce the number of regions by merging spectrally similar regions by an order of magnitude. Finally, in Section 3 we present an interactive tool for volume data segmentation and visualization which takes advantage of region hierarchies obtained by the hierarchical watershed transform.

2. THE STREAMED WATERSHED TRANSFORM

The watershed transform (WT) is a popular technique for segmentation of spatial data in homogeneous regions. It usually works on gradient images, which are in the case of the WT treated as topographic height maps: homogeneous regions in gradient images represent valleys while their borders represent ridges. The segmentation is obtained either by simulating downhill water flow or "immersing" the relief into water [12]. In both cases local minima of the gradient image become region seeds which are labeled by individual labels. In computation these labels are propagated uphill until a region with a different label is reached – the set of such border locations creates the watersheds. Since the watersheds are formed exactly on the ridges, the regions correspond to homogeneous image areas.

In the case of real-world noisy images this process results in over-segmentation – the detected regions are too small. Therefore, the WS is usually applied on smoothed gradient images (for example, by the Gaussian filter), which ensures production of larger regions. In the case of large smoothing kernels the watersheds, however, do not follow precisely the edges between the regions. Therefore, smoothing only with some (application specific) maximum level is applicable. If larger regions are required, they can be obtained by, for example, creating region hierarchies [8] or by merging similar neighboring regions (Section 3).

In the downhill techniques for each data voxel one follows the downhill path until a labeled voxel is found (this may be a labeled minimum) and subsequently the label is assigned to all voxels of the path. Although one cannot predict length of the path, our experience has shown that the diameter of the created watershed regions is on average approximately equal to the size of the smoothing kernel. Therefore, the downhill techniques are appropriate for streamed implementations.

The first streamed implementation is a single-pass one in which a slab of slices is stored in the memory. The number of stored slices stored is adaptive: if, while following the downhill path, access to a new slice is required, it is loaded to the bottom of the slab and if all voxels of the top-most slab are processed, the slab is stored to the output and removed from the memory. Although artificial data sets, which would force loading of all slices, can be created, in real data sets the number of stored slices follows the observation mentioned in the previous paragraph.

The second implementation is a multi (three)-pass one, which requires concurrent storage of only three neighboring slides in the memory. In the first run we process the data from the start to the end slice by slice and in each voxel either store the ID of the local minimum to which this voxel belongs to or a pointer to the voxel in the next slice. The type of the stored data depends on the path of the steepest descent. As the previous slice has already been processed, in that direction we either hit a voxel with assigned local minimum ID or move back to current slice. After finite number of steps we either find a voxel with local minimum ID assigned (in the previous or current slice) or end up pointing to a yet unprocessed voxel in the next slice. In both cases we mark all unprocessed voxels on the path with found value – local minimum ID or a pointer. Because the last slice doesn't have a next slice, all its voxels have region IDs assigned.

The data processed in the first pass is stored out-of-core (in a temporary disk file) and in the second pass is loaded in a reversed order from the last slice to the first. Moving up from the last slice to the first we propagate the region IDs in the reverse direction of the stored pointers. After processing the whole volume all voxels which contained a pointer after the first pass have now assigned some region ID. The third pass does no processing at all – we just need it to reverse the order of the slices which, after the second pass, were stored in the reversed order from last to first.

If memory consumption is a real concern and saving memory by applying the multi-pass watershed transform is not enough, one can further reduce the memory requirements of the watershed transform by replacing the standard convolution-based Gaussian smoothing by a multi-pass FIR-based filtering. While

in the convolution the amount of stored slices depends on the filter sigma, in the FIR case the number is fixed to 4 [11]. In this variant the volume is also processed in three passes with out-of-core storage.

Table 1 presents results obtained for a 512×512×1268 CT data set in floating point precision (1.3 GB) for both watershed algorithms. The computation included concurrent execution of the five following processes: Gaussian filtering, gradient magnitude computation, detection of local minima, labeling of local minima and computation of the watersheds. The Time column is the system time required for all these processes.

We can see that the average number of allocated slices in the single-pass method is really approximately equal to the size of the Gaussian kernel. This amount is, on the one hand, much higher than the number of slices required by the multi-pass technique (always just 3), but, on the other hand, is more than order of magnitude lower then the number of slices of the data set. Regarding the execution time the multi-pass technique is slightly slower that the single-pass one.

Table 1: Results obtained for single- and multi-pass streamed watershed based segmentation.

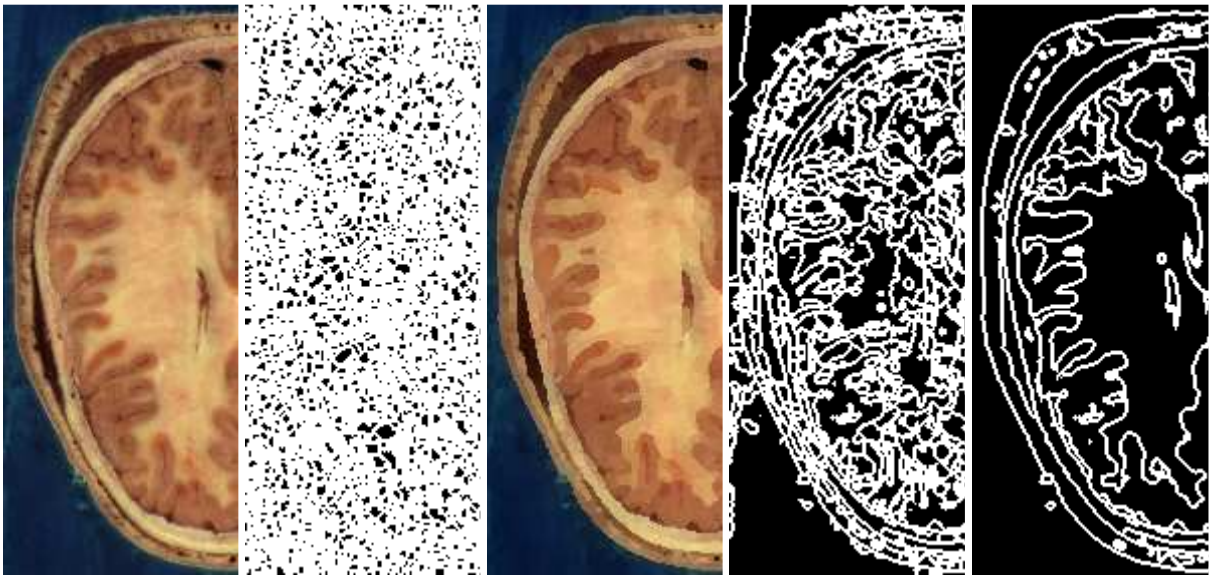
	Single-pass			Multi-pass	
Filter size	Time [s]	Mean slices	Max slices	Time [s]	Max slics
3	549	10	27	584	3
5	547	11	24	607	3
9	551	14	30	598	3
13	563	17	39	612	3
17	554	21	46	616	3
21	553	25	52	600	3
27	559	32	75	605	3
33	561	36	72	610	3

3. REGION-BASED PROCESSING OF VOLUMETRIC DATA

Partition of input data in homogeneous regions brings in two advantages. First, the number of primitives to process decreases many times and, second, processing of regions is less sensitive to noise than processing of individual voxels.

Data smoothing can be regarded as the most basic operation over the regions. If the original noisy region values are replaced by their mean or median, not only the noise is removed, but also the edges, which are often blurred by the partial volume effect, are sharpened. This capability is praised also for other smoothing techniques, for example the anisotropic diffusion based ones [5]. Results of region-based smoothing are presented in Fig. 1 a-c on the female head data of the Visible human project [1]. In this case, only minimal smoothing with Gaussian sigma 0.1 was used, in order to keep as high fidelity with the original as possible. The sharpened edges are clearly visible.

We already mentioned earlier that the watershed transform leads to over-segmentation. Thus, another useful operation may be merging of neighboring regions with identical or similar properties to larger regions. For this purpose we developed a technique which is based on merging neighboring regions according to their spectral distance. As the spectral distance we use the Euclidean distance between average densities of neighboring regions for each band. In merging, all neighbor pairs are first sorted according to their distance and stored in the updatable priority queue data structure [3]. Then, in a loop, a neighbor pair with the smallest distance is popped from the top of the queue, the regions of the pair are merged, new neighbor pairs are created for the new region and inserted into the queue and, finally, the old pairs related to the original regions are removed from the queue. The process is repeated until the required pair distance or the required number of regions is reached. Experiments have shown [6] that the number of regions can be decreased down to 10% without significant increase of homogeneity of the regions. Edges of regions created by reducing the number of regions to 10 and 1% are shown in Fig. 1 d-e. We can see that the region edges follow the edges of the density image as expected.



(a) (b) (c) (d) (e)

Fig. 1. Region-based processing of color data. (a) the original data, (b) watershed regions obtained with Gaussian sigma 0.1, (c) region-based smoothing based on (b), (d-e) number of regions of (b) reduced to 10 and 1 % by merging similar neighbors.

4. INTERACTIVE SEGMENTATION AND VISUALIZATION

Segmentation of the medical volumetric data is a difficult problem which requires skilled operator to control the process and validate the results. A range of segmentation approaches exists each having their own characteristics – amount of operator input required, precision of the resulting segmentation, ability to be used for segmentation of arbitrary objects, etc. Generally, working on more detailed level requires more operator input rewarding us with more precise result. On the other hand, on the coarser level we might obtain less precise result, but faster and with less work.

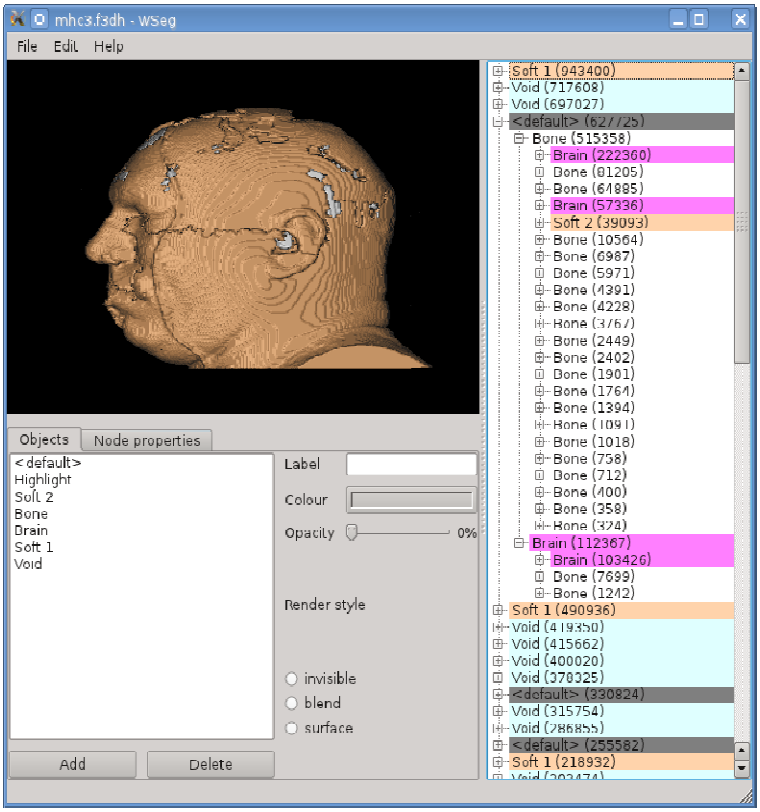
In the proposed region-based interactive segmentation technique we took advantage of the hierarchical watershed transform [8] which provides the user with the possibility to interactively select the detail level on which he/she is working on. The technique is supplemented with region-based interactive visualization of the intermediate or final segmentation results.

The method requires input data to be processed by some operation creating regions of voxels at different scale levels followed by creating of a region hierarchy. One option is to use the watershed transform – the input data is blurred by Gaussian filer with increasing kernel size and then the watershed transform is applied. Another possible approach is to use the watershed transform only for the most detailed level and then to merge neighbouring regions using some similarity criterion. After creating a set of volumes with successively larger regions we create a hierarchy by propagating sharp region boundaries from most detailed volume to regions in volumes which underwent stronger blurring or region merging. Here, regions in neighbouring volumes are compared and parent-child relationship is established assigning to regions from the coarser level those regions from the more detailed level which share with it the most voxels. The result is a tree structure where each node represents one region with each tree level belonging to different detail level.

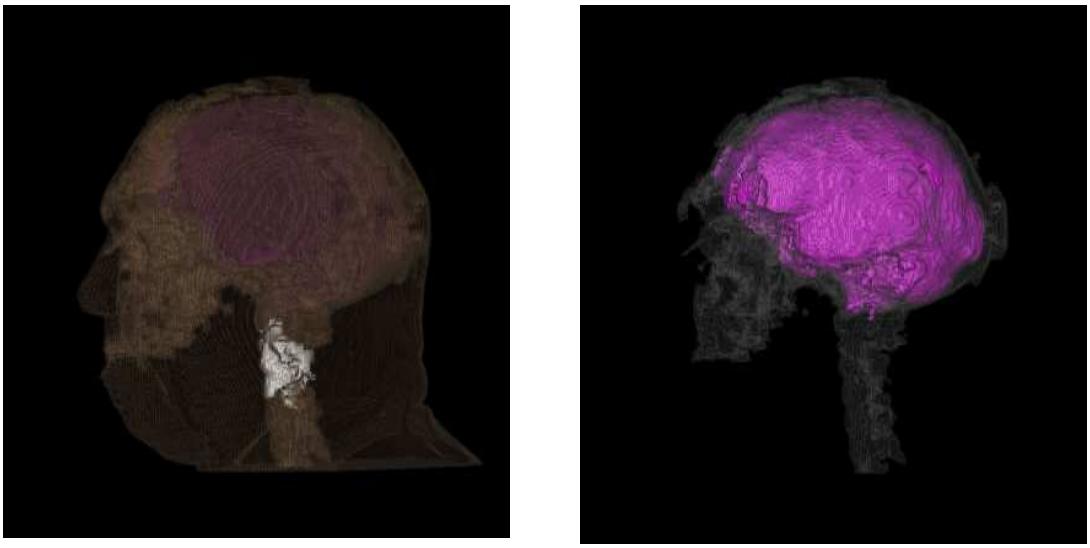
In the interactive segmentation and visualization tool which displays a loaded region hierarchy as a tree structure using a common GUI tree-view widget a user can select any node in the hierarchy and specify to which segmented object this node belongs to. Default setting for all nodes is to inherit properties of the parent node, thus by assigning certain node to some object we actually classify the whole subtree of this node. This is equivalent to performing the assignment on all nodes of the most detailed level of which this node is composed of. For each node it is also possible to override the parent setting and change the assignment to some other segmented object. This allows the user to at first classify nodes at a coarser level and then to move to more detailed level and correct the classification errors if necessary.

For the purpose of visualization the user can specify color and opacity in the form of a transfer function for each segmented object he/she defines. Rendering is performed using the volume with the most detailed region level. Each voxel contains ID of the region to which it belongs to stored as an unsigned int value. Alongside the main volume data which is loaded as a 3D texture a couple of supplementary textures are used – mapping between the region ID and the assigned segmented object ID, optical properties of each defined object and finally a rendering style of each object. The rendering style can be either ‘invisible’, ‘blend’ or ‘surface’. The first is self-describing and the second ignores lighting and just blends values from the objects’ transfer functions. The last rendering style performs lighting and renders only the surface. Since we work only with label volumes and do not store the density data, in order to determine the direction of the surface normal neighbouring voxels are visited noting whether they belong to the same segmented object or not. Two points are computed – center of mass for the neighbouring voxels which belong to the same object as the currently processed voxel and for those which do not. Surface normal is then estimated as a vector defined by these two points. Inner voxels are not rendered.

Figure 2 illustrates work-flow with the segmentation tool on a 128×256×256 CT data set. Loaded hierarchy is shown on the right. Nodes in the hierarchy were sorted according to the number of the voxels they represent therefore user will find largest regions at the top. When hovering a hierarchy node with the mouse cursor the render widget displays shaded surface of the respective region with all other regions rendered with lower opacity to provide context. Hovering region belonging to part of spine is shown on 2 b. After relevant region is found assignment to some segmentation object can be specified. Objects are defined in the lower left part of the window. If the selected region is composed of voxels of various objects classification can be refined on the more detailed level. To view currently processed region from various angles user may assigning the region to a special highlight object with full opacity and bright color. Having specified transparent colors for target objects allows user to easily localize and identify regions being processed in respect to already classified regions. After being properly identified region can be correctly classified or user may classify child regions instead. On the 2 a a partially expanded hierarchy is shown for a node consisting of various tissue types. Parent assignment to default object is ignored by the children for the shown node.



(a)



(b) (c)

Fig. 2. Interactive segmentation and visualization tool. (a) GUI layout with a render widget at top left, segmentation objects at lower left and partly expanded hierarchy on the right. Numbers in the parentheses show region's voxel count. (b) highlight rendering of the region selected by a mouse pointer (a part of spine in this case). (c) rendering of segmented brain regions together with highly transparent bone regions.

5. CONCLUSION

In this paper we presented two algorithms for segmentation of volume data by the watershed transform with data streaming – a single-pass one with higher memory requirements and a multi-pass out-of-core one with fixed storage of three slices. Further, we presented a technique for region-based data smoothing featuring edge sharpening, a technique for merging of neighbor regions based on spectral distance and finally an interactive approach to region-based segmentation and visualization. In the future we plan to extend the interactive technique to segmentation of multi-spectral data by employing various data analysis approaches known in the information visualization area.

6. ACKNOWLEDGMENT

This work was supported by the Slovak Research and Development Agency under the contract No. APVV-20-056105 and Science Grant Agency (VEGA) under contract No. 1/0631/11.

BIBLIOGRAPHY

[1] ACKERMAN M.J., The visible human project, Proceedings of the IEEE, Vol. 86, No. 3, 1998, pp. 504–511.
[2] BANKMAN I.N. (ed.), Handbook of Medical Imaging, Processing and Analysis, Academic Press, 2000.
[3] GREGSON J., An updateable priority queue [online], Available from: http://jamesgregson.blogspot.com/2010/02/updateable-priority-queue.html [cited July 26, 2011].
[4] IBANEZ L., SCHROEDER W., NG L., CATES J., The ITK Software Guide: The Insight Segmentation and Registration Toolkit, Kitware Inc., first edition, 2003.
[5] PERONA P., MALIK J.,. Scale-space and edge detection using anisotropic diffusion, IEEE Trans. Pattern. Anal. Machine Intell., Vol. 12, 1990, pp. 629–639.
[6] PÁLEŠOVÁ E,. Segmentácia farebných objemových dát pomocou klasifikácie. Master's thesis, Comenius University Faculty of Mathematics, Physics and Informatics, Bratislava, Slovakia, June 2010, (in Slovak).
[7] SCHROEDER W.J., MARTIN K.M., LORENSEN W.E., The Visualization Toolkit, Kitware Inc., third edition, 2004.
[8] SRAMEK M., DIMITROV L.I., Segmentation of tomographic data by hierarchical watershed transform, Journal of Medical Informatics and Technologies, Vol. 3, 2002, pp. 161–169.

[9] SRAMEK M., DIMITROV L.I., STRAKA M., CERVENANSKY M., The f3d tools for processing and visualization of volumetric data, Journal of Medical Informatics and Technologies, 2004, pp. 71–79.
[10] VARCHOLA A., VAŠKO A., SOLČÁNY V., DIMITROV L.I., ŠRÁMEK M., Processing of volumetric data by slice and process-based streaming, in SLAY H., BANGAY S., (ed.), Afrigraph'07, ACM Siggraph, Grahamstown, South Africa, 2007, pp. 101–110.
[11] VAŠKO A., ŠRÁMEK M., Optimizing Gaussian filtering of volumetric data using SSE, Concurr. Comput. : Pract. Exper., Vol. 23, No. 1, January 2011, pp. 100–116.
[12] VINCENT L., SOILLE P., Watersheds in digital spaces: An efficient algorithm based on immersion simulation, Vol. 13, No. 6, June 1991, pp. 583–598.