Extracting Regions of Interest Applying a Local Watershed Transformation

Stanislav L. Stoev* Wolfgang Straßer*

University of Tübingen, WSI/GRIS

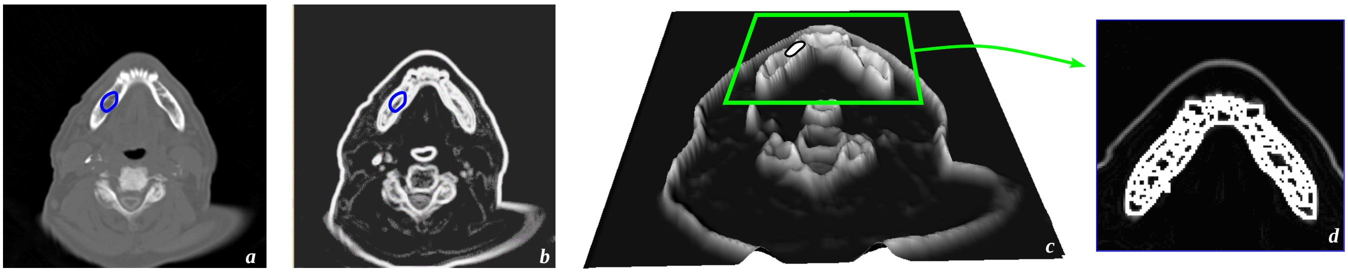


Figure 1: The region extraction pipeline: ***a*** Selection of a region of interest (ROI). ***b*** Performing a simple edge detection. ***c*** Flooding the relief applying the proposed local watershed transformation. ***d*** The result: a set of basins covering the entire ROI.

Abstract

In this paper, we present a new technique for extracting regions of interest (ROI) applying a local watershed transformation. The proposed strategy for computing catchment basins in a given region of interest is based on a rain-falling simulation. Unlike the standard watershed algorithms, which flood the complete (gradient magnitude of an) image, the proposed approach allows us to perform this task locally. Thus, a controlled region growth is performed, saving time and reducing the memory requirement especially when applied on volume data.

A second problem arising from the standard watershed transformation is the over-segmented result and the lack of sound criteria for merging the computed basins. For overcoming this drawback, we present a basin-merging strategy introducing four criteria for merging adjacent basins. The threshold values applied in this strategy are derived from the user input and match rather the attributes of the selected object than of all objects in the image. In doing so, the user is not required to adjust abstract numbers, but to simply select a coarse region of interest. Moreover, the proposed algorithm is not limited to the 2D case. As we show in this work, it is suitable for volume data processing as well. Finally, we present the results of applying the proposed approach on several example images and volume data sets.

CR Categories: I.4.6 [Image processing and computer Vision]: Segmentation; I.4.6 [Image processing and computer Vision]: Reconstruction; I.5.3 [Pattern recognition]: Clustering; J.3 [Life and Medical Sciences]: Medical Information Systems;

Keywords: Computer Vision; Image Processing; Data Visualization; Volume Visualization; Feature Extraction; Morphological Segmentation; Biomedical Image Segmentation; Watershed Transformation;

1 Introduction and Related Work

The extraction of meaningful regions from image and volume data continues to be an important and unsolved topic in the image analysis and image processing area. In the past years, the watershed transformation [8, 12] has proven to be a very useful and powerful tool for morphological image segmentation. Since its introduction in [1], it is becoming more and more popular in different science areas like biomedical, medical image processing [4, 15], computer vision and segmentation [19], even mesh segmentation [7] etc.

The idea of the watershed transformation is quite simple. A (gradient magnitude of a) gray-scale image or volume is considered as a topographic relief [18]. Each pixel in this digital image is assigned during the transformation to the catchment basin of a regional minimum. The *catchment basin* of a regional minimum is defined as the area, in which the pixels hit by a rain-drop will cause the rain-drop to flow to the regional minimum. In this way, influence zones for each of the (ev. pre-determined [8]) regional minima are defined. The watershed lines are now the lines separating influence zones from each other.

An immersion based watershed algorithm was first proposed by Beucher and Lantuéjoul [2]. In [8] and [3] couple of techniques and algorithms related to the problem of watershed computing are described. In his work [8], Meyer defines the watershed transformation in the continuous and in the digital space in terms of a distance function, called topographic distance. One of the classical algorithms for computing the watershed transformation for a gray-scale image is also found in this work. The author predetermines the regional minima (single pixels or plateaus) and starts the flooding process at these minima [8].

Another approach for computing the catchment basins is described in [18]. The authors simulate flooding with water, coming up out of the ground and defining the catchment basins without predetermining the regional minima. This approach processes all image pixels in a sorted order, such that pixels with lower altitude are processed first. The preprocessing step here consists of sort-

*Auf der Morgenstelle 10 C9, 72076 Tübingen, Germany, Phone: +(49) 7071-2975464, Fax: +(49) 7071-295466, Email: {sstoev,strasser}@gris.uni-tuebingen.de

ing all (pointers to) pixels in an array. Utilizing a First-In-First-Out (FIFO) structure, the pixels at altitude *h+1* are processed after those at altitude *h*. This divides the problem into *m* subproblems, where *m* is the number of all present pixel altitudes. Due to the processing of pixels at altitude h in every iteration, the problem is reduced to computing the geodesic *skeleton of influence zones* (SKIZ).

In order to achieve a meaningful segmentation, the watershed transformation is in general performed on gradient images. Since in these pre-processed images the gradient maxima are more often than not (watershed) lines, surrounding homogenous regions.

A disadvantage of the watershed transformation as described in the literature is that it is a global transformation, hence it requires the processing of the entire input data. However, the goal of the segmentation is often the extraction of only a single region of interest (ROI) out of a given image/volume data. Especially when volume data is considered, the process of computing the watersheds is very time and memory consuming step.

Furthermore, the extraction of a sole catchment basin does not provide sufficient results. Since the global watershed transformation produces in general heavily over-segmented results, a sole catchment basin is often meaningless for the segmentation of a region of interest. On the other hand, the correct contours are most of the time present in the transformed image. Thus, an additional task after applying the global watershed transformation has to be performed: the computed catchment basins have to be merged appropriately. Various approaches for accomplishing this task are described in the literature and are based on gradient-watersheds on graphs [18], basin dynamics [13], markers [9], inclusionary and exclusionary cues [4], image component labelling [10], and multi scale gradient analysis [5]. In practice, however, it is very difficult to define sound criteria for all objects in the image. Therefore, the definition of *local* merging criteria, matching the attributes of a given ROI, yields a sound solution of this problem.

In this paper, we present a new approach for *locally* applying the watershed transformation. We first describe how a single basin can be extracted without completely computing the watershed-transformed and without pre-processing the input data. Afterwards, this methods is generalized to the case, where a user makes a coarse region selection, which is flooded and the basins within the selected ROI are merged. These selected basins are now utilized to derive threshold values for a set of merging criteria, which will be discussed later in this work. Thus, a sort of controlled basin growth is performed based on both the regular input data and its gradient magnitude transformed. This makes it possible to precisely define the merging criteria for the object of interest, rather than for the entire input data.

The remainder of this paper is organized as follows. In the next section we describe how the watershed transformation is applied locally, without pre-processing the input data, in order to extract one single basin. In Section 2.2, the processing of the user input is discussed and how this data is utilized for deriving the thresholds for basin merging. The four merging criteria are described in detail in Section 3, their evaluation in Section 4, and their application during the basin growth in Section 5. Finally, in Section 6 we present results produced with the proposed technique. Sections 7 and 8 include topics of future work and offer concluding comments.

2 The Local Watershed Transformation

Unlike the standard watershed algorithms, the aim of the approach described in this section is to provide a strategy for basin computing, which does not require pre-processing or other global information about the data. Unfortunately, all approaches described in the literature require such pre-processing steps, in order to either sort all pixels of the input image [18], to pre-compute the local minima from where the basins are flooded [8], or to introduce a metric for pixels with equal altitude: plateau pre-computing [11]. The last approach seemed to us to be the most suitable for modification toward developing a local working basin extraction technique. Unfortunately, the authors of [11] concentrate on the parallelization and neither on the local applicability of the proposed approach, nor on the processing of large data sets (i.e. volume data). However, since the plateau computing can be performed only when this is required (i.e. a plateau is reached), we utilize a modified version of this technique in our algorithm. In the pre-processing step required in [11] the entire input data is traversed. This makes the approach time and memory consuming, when applied on large volume data sets. Moreover, the authors of [11] do not sort the outdoors of a plateau, before it is flooded, hence an error is introduced compared with the immersion based watershed algorithms (see below). Finally, a second error source is introduced due to the arbitrary choosing of a pixel in case 4 as described in the next section.

2.1 The Local Basin Extraction

During the data processing, a rain-drop-hit is simulated for every pixel. Each rain-drop follows the path toward the line with the steepest descent due to gravity, until a regional minimum is reached. A *regional minimum* M is a single pixel or a set of pixels with equal altitude, from which it is impossible to reach a point of lower altitude on an always descending path. When such a minimum is reached, the pixels attracted on the path of steepest descent are marked with the label of the latter.

The reached minimum M is now flooded in the following way. For each of the (yet unmarked) pixels q adjacent to the minimum M, again the steepest path is followed. In case the minimum reached from q is also M, q is assigned to the catchment basin of M denoted by B (see Figure 2). In this case, the unprocessed pixels q' adjacent to each q are assigned to the set of unprocessed pixels Q.

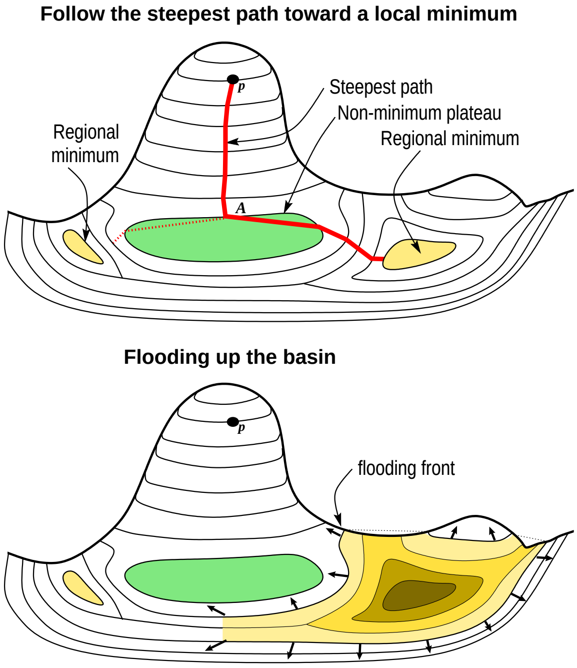


Figure 2: The pixel p is hit by a rain-drop. The algorithm follows the steepest path toward a local minimum (upper diagram). Afterwards, the basin is flooded with water coming up out of the reached minimum (lower diagram).

These pixels are considered in the next pixel iteration for the current basin. Otherwise, if another local minimum M' is reached, M' is put on the queue containing the yet unprocessed basins/minima (considered in the next basin iteration). The pseudo-code outline of the algorithm for computing the catchment basin for a given pixel follows:

```
0   follow_steepest_path($p$,$M_i$);   /* stores reached minimum in $M_i$ */
1   $B_i \leftarrow M_i$;   /* initialize $B_i$ with the pixels in $M_i$ */
2   $Q \leftarrow$ border($M_i$);   /* initialize $Q$ with the pixels surrounding $M_i$ */
3   while (not_empty($Q$))
4       $q \leftarrow$ pop_pixel($Q$);
5       q_id $\leftarrow$ follow_steepest_path($q$);
6       if (q_id == Id($M_i$))
7          add_pixel($B_i$,$q$);   /* assigns $q$ to the basin $B_i$ */
8          get_unmarked_neighbors($Q$, $q$);   /* result- stored in $Q$ */
9       endif
10  end while;
```

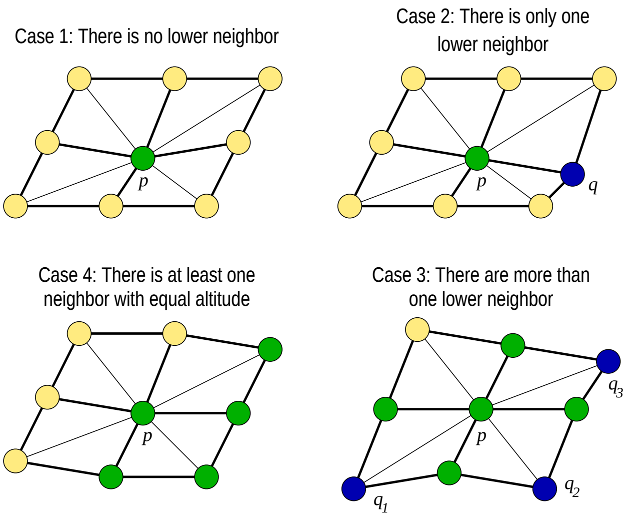
The only complex procedure requiring more detailed discussion is the process of following the steepest path toward a local minimum, starting at a given pixel (see line 0 and line 5 above and Figure 2). In order to explain the single steps, we assume that we start with pixel p, which has not been processed yet. Thus, four cases can occur (see also Figure 3):

1. p has no adjacent pixel with lower altitude, hence p is an isolated regional minimum;

2. p has only one adjacent pixel with lowest altitude q. This is the regular case, where the algorithm follows the steepest path along the shortest topographic distance;

3. p has at least one adjacent pixel with the same altitude which means that p belongs to a (minimum or non-minimum) plateau;

4. p has more than one adjacent pixel with lowest altitude q_i. In this case the algorithm can not determine which of the adjacent pixels is the one, the raindrop should flow to.


Case 1: There is no lower neighbor

Case 2: There is only one lower neighbor

Case 4: There is at least one neighbor with equal altitude

Case 3: There are more than one lower neighbor

Figure 3: The four cases, which can occur when the steepest path toward a local minimum is followed.

The first two cases are the trivial ones. When case 1 occurs, a regional minimum is reached and a new Id is assigned to the basin. In case 2, the current pixel is assigned to the path and if the lowest neighbor q is not marked yet, it is considered as the next processed pixel: $p \leftarrow q$. If q is already marked, the current path is terminated and its pixels are labelled with the label of q (*early path termination*).

In case 3, the reached plateau has to be processed first, since the steepest path can not be unequivocally determined within plateaus. Therefore, when a plateau is reached, its border pixels are determined (see Figure 4). If there are no border pixels with lower altitude, the plateau is a minimum plateau and the algorithm proceeds like in case 1. Otherwise, the *outdoors*[1] of the plateau P are sorted and used as starting points for flooding P, as depicted in Figure 4.

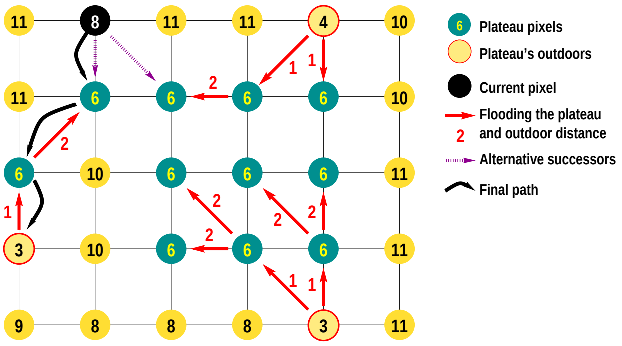


Figure 4: When the plateau with altitude 6 is reached, the algorithm first determines its outdoors and then floods the plateau, wiring the pixels within it. The pixels with the shortest distance and if these are equal, the one with the lowest outdoor is chosen to be processed next.

In order to prevent multiple flooding of the same plateau, the flooding results are saved in a special data structure. In this structure, the pixels are *wired*, storing for each pixel the distance to the nearest outdoor and the direction of the latter (see Figure 4). This approach is similar to the *arrowing* technique described in [18] and [8]. Later on, when a pixel of a processed plateau is reached, the arrows to the nearest outdoor are followed and no additional computations are performed. Thus, the nearest and lowest outdoor considering p is now set to be the next pixel in the current path and is processed next. If a marked pixel is reached within the plateau, the current path is terminated and labelled with the Id of this pixel (as in case 2).

When case 4 occurs, the algorithm can not unequivocally decide which pixel should be processed next. In this case, all adjacent lowest pixels are traversed as if they were hit by a rain-drop. Since the lowest pixel $p_n = p$ of the current path $(p_1, \ldots, p_n)$ has a greater altitude than the pixels q_i and a path always follows the steepest slope, none of the pixels $(p_1, \ldots, p_n)$ is affected while q_i are being processed. This allows for the algorithm to remain consistent in this case. Hence, after processing all q_i, the pixel q_j with the lowest and nearest outdoor is chosen to be the next processed one $p \leftarrow q_j$ and the computation of the steepest path continues. For pixels q_i not belonging to a plateau, the outdoor-distance is considered as one.

This technique allows us to determine for each pixel the regional minimum it belongs to, without performing a global flooding (typically performed in immersion based watershed transformation) or pre-processing the data. In particular, a rain-drop follows the steepest path toward a local (gradient) minimum. This path is now traversed and its pixels are labelled with the Id of the reached min-

[1] *Outdoor*-pixels of a plateau are pixels adjacent to the plateau with altitude lower than the plateau's altitude.

imum. Afterwards, the basin is flooded "piercing" its minimum and letting the water come up out of the ground. This procedure is repeated until all boundary pixels of the current catchment basin have adjacent pixels belonging to other basins. In this way, we are able to extract a single catchment basin given an image/volume data set and a starting pixel/voxel. When volume data is processed, 26-connectivity grid is used, instead of the 8-connectivity grid applied in regular 2D images. All other steps remain unchanged. Moreover, the proposed approach can be applied even on data structures with higher connectivity in the same way.

2.2 Flooding the Selected Region

As introduced above, the proposed approach requires a selection of a region of interest (ROI) from the user. For user convenience, this is performed with the original input image I. In a separate pre-processing step, a simple edge detection based on the standard Sobel edge detector [14] is applied to the entire input data. This is a straightforward process with insignificant time cost, compared to the global watershed transformation, hence not slowing down the entire process.

When this is accomplished, the gradient of the input area is flooded, simulating rain-drops starting with an arbitrary pixel within the selected area. Afterwards, the above transformation is repeatedly performed on I' until all pixels and basins *within* the ROI are processed (as shown in Figure 5 for the 2D case). Since these basins belong to the same region, due to the user selection, they are used to derive thresholds for the *merging criteria*. As will be shown in Section 3, both the original image data I and the gradient magnitude image I' are applied.

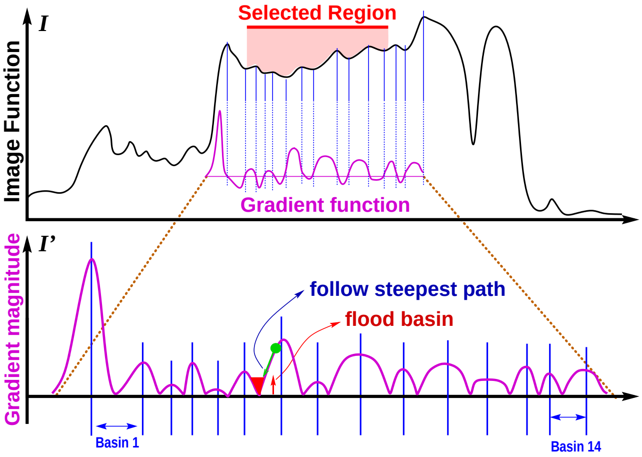


Figure 5: The selected ROI in a simple 2D example, the corresponding gradient magnitude image (lower diagram) and its basins.

Unfortunately, there are basins which are not completely, but only partially within the ROI. These basins are processed as follows: For each pixel, not belonging to a basin with local minimum within the selected region, again a rain-falling simulation is started (see also [16]) and the regional minimum of the new basin is determined as described above (ev. outside the user selection). Each of these minima is used to start a flooding process as described in the Section 2.1. The so computed layer of basins surrounding the user-selected area are utilized to derive a second set of auxiliary thresholds for the merging criteria as will be discussed in Section 3. When this step is completed, all basins which are partially or completely included in the selected region are processed.

3 Merging Criteria

The most important part of the proposed approach is the suitable definition of the basin-merging criteria, since they define the quality of the segmentation results. As introduced before, various merging criteria are described in the literature. The most prominent ones are based on gradient-watersheds on graphs [18], basin dynamics [13], and multi scale gradient analysis [5]. Most of these approaches define appropriate merging criteria based on the basins itself, hence using the information about the gradient image. In contrast, we implemented a method, which uses a combination of the first two approaches, additional information about the original image data I, and the second derivation of the input image I''. Thus, a more sound merging condition is introduced. Additionally, it is very difficult, often even impossible, to define a merging criterion matching the attributes of all objects in the input image. Applying a local watershed transformation and merging only basins fulfilling local ROI-specific merging criteria yields significantly better results as will be shown in section 6.

Merging Criterion A

Now let us consider two adjacent basins as depicted on the left of Figure 6. As proposed in [12] the basin's dynamic introduces a good criterion for merging adjacent basins. The dynamic of a basin is defined by the minimum height, which has to be overcome, in order to reach a basin with lower or equal minimum altitude. Basins with dynamics lower than a given value d_t (*dynamics threshold*) are now merged in larger regions. The usual values of d_t are in the interval $[5, 20]$ for 8-bit gray-scale images and depend on the image content and it's noise. Applying this strategy for basin merging helps us to merge insignificant basins, introduced by various noise sources in I and I' (see Figure 9 image ***b***).

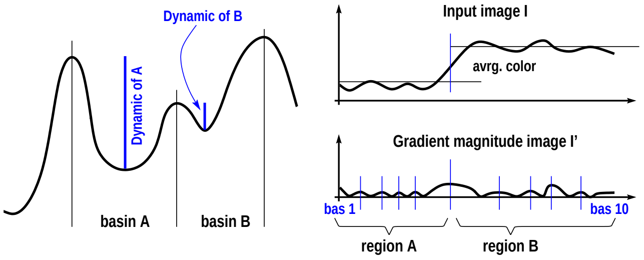


Figure 6: The basin's dynamic is shown on the left. On the right, a case is shown in which the basin's dynamic only does not provide a good merging criterion.

Merging Criterion B

Nevertheless, there are still situations, in which even basins with a great difference between the (mean) gray-values in I have to be merged when only their dynamic is taken into account (as shown on the right of Figure 6). To overcome this problem, we applied additionally to the concept of basin dynamics a second criterion. This is based on the mean gray value of the region in I, corresponding to the basin in I'. A similar strategy is introduced in [18], where the authors use the infimum of the area in I, corresponding to the basin's regional minimum in I'. Furthermore, the authors apply a watershed algorithm on a graph with nodes representing the basins and arcs linking adjacent basins. Thereby, the nodes are colored with the infimum value of the basins minimum in I. For our application, however, it turned out that applying the average color c_a of the catchment basin in the original image I is a better criterion for basin merging. Unlike [18], we compare the values c_a' and c_a'' of each two adjacent basins in order to determine whether they should be merged or not ($|c_a' - c_a''| < c_t$ - *color threshold*). In this way we defined a second criterion for basin merging based on the global basin attributes in I and in I'.

Unfortunately, another problem occurs with these merging criteria: too many basins fulfilling the requirements, but not belonging to the same region are merged. This may significantly worsen the segmentation results (see Figure 9, image ***c***). To circumvent this obstacle, we introduced two additional rules for preventing basins from merging.

Merging Criterion C

The third criterion we applied in our algorithm is based on the characteristics of the dam between two basins B_A and B_B. During the basin extraction, we record the length, the average color, and the color of the lowest pixel p_b on the dam between B_A and B_B in I'. If the altitude difference between the lowest common border pixel p_b and the average height of the border b_a is greater than a given threshold value b_t (*border threshold*, with values typically within the interval $[5, 15]$), the basins are not merged even if the conditions stated in the first two criteria are fulfilled (see Figure 7). In case the basins have more than one common border, the border pieces are processed as if they were connected. This criterion prevents from merging basins, which have similar dynamics due to noise-containing borders.

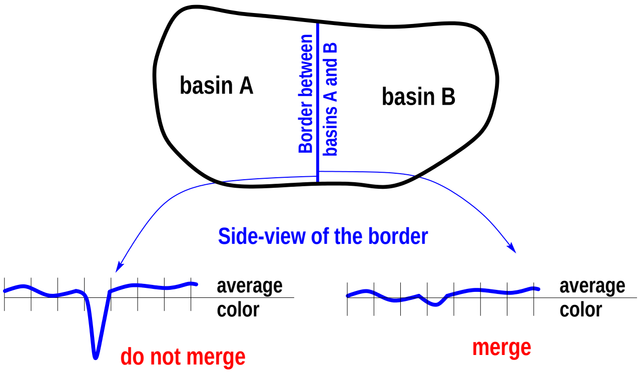


Figure 7: The basins are not merged if the gap between the lowest border pixel, defining the dynamic, and the average border altitude is too large as shown on the right.

Merging Criterion D

Finally, we introduced a measurement for the *steepness* of the border between two basins in I'. For this, we consider each border pixel p_b and the next pixel on the path toward the local minimum p_1, as shown in Figure 8. For each such border pixel p_b, we determine the pixel p_1 and compute the magnitudes of the second derivations p_b'' and p_1'', hence the values of p_b and p_1 in I''. The absolute difference between these values $p_s = |p_b'' - p_1''|$ gives a measurement of the gradient's variation in I'. If p_s is greater than a given threshold s_t (*steepness threshold*), for any pixel on the border between adjacent basins, this basins are not merged even if the criteria A-C are fulfilled (see the right of Figure 8). This criterion is applied to each two adjacent basins (for instance A and B in Figure 8). The border steepness of both of them has to be below the given threshold (in general within the interval $[5, 20]$), in order to merge the basins.

To summarize, for each extracted basin B, we dispose of and evaluate the following data structure, in order to determine whether it should be merged with an adjacent one B':

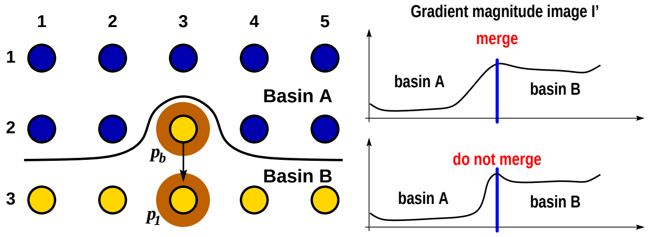


Figure 8: The magnitude of the second derivation is computed at each inner border pixel and at the next pixels on the path toward the basin's minimum (pixels (3,2) and (3,3)). If the absolute difference between these two values is greater than a given threshold s_t, the basins are not merged. On the right, a case is shown in which the steepness of the basin's border prevents from merging two adjacent basins.

- the average grey value of the basin in I (criterion B);
- the lowest and highest altitude of a basin pixel in I' (criterion A,C, and D);
- lowest and average border value for each pair of adjacent basin (B, B') (criterion C);
- for each inner pixel of the border and each adjacent basin the maximum p_s (criterion D).

Results of applying the various merging strategies are shown in Figure 9. We first show the result of applying the standard global watershed transformation and the application of the criteria A and B. Afterwards, the results of applying the local watershed transformation are shown evaluating the criteria A and B in image ***c*** and criteria A-D in image ***d***.

4 Evaluation of the Merging Criteria

After flooding the ROI selected by the user, we compute the values for the thresholds: $T = (d_t, c_t, b_t, s_t)$. The value of d_t is the value of the greatest dynamic for basins within the selected ROI. c_t is defined as the greatest allowed mean color deviation for two adjacent basins. When all basins in the given ROI are computed, the mean color of this regions is used to define the reference grey value. c_t is now applied for defining the deviation threshold relative to the reference grey value. b_t is initialized with the maximum difference between the average border color and the lowest border pixel for each pair of adjacent basins. Finally, the magnitude of the greatest second derivation of the grey value in I' on the inner border is assigned to s_t (see Criterion D above).

In order to determine whether a computed catchment basin matches the initial user selection, we evaluate the following *merging rule*, applying the thresholds T:

```
IF Criterion A AND Criterion B
    AND Criterion C AND Criterion D
    THEN Merge_Basins$(A, B)$.
```

Since there are basins, which are only partially covered by the selected ROI, these border basins are utilized for deriving an additional threshold set: $T' = (d_t', c_t', b_t', s_t')$. When a new basin is extracted, we first check whether the premise of the above rule is true for the threshold set T. If this is the case, the basin is assigned to the initial ROI. Otherwise, the second threshold set T' is applied

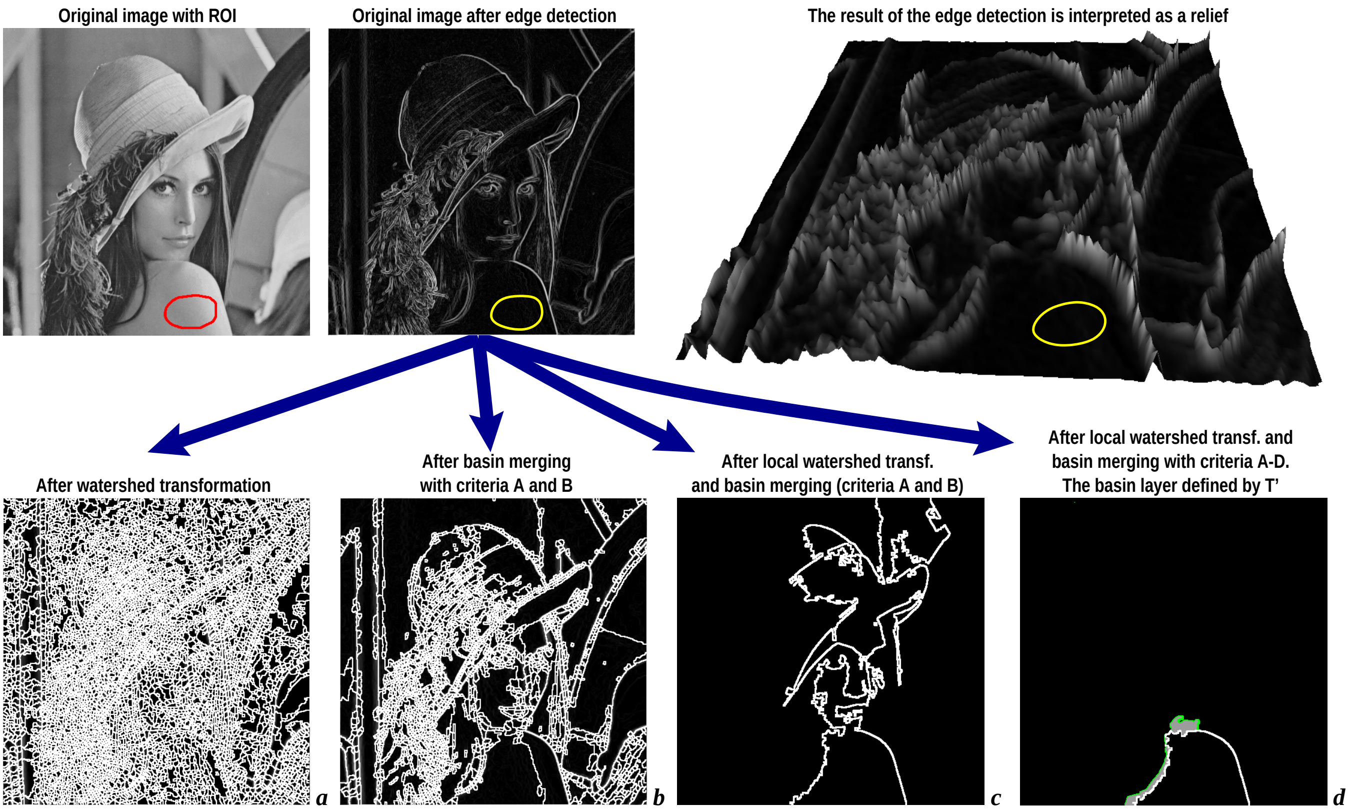

Original image with ROI
Original image after edge detection
The result of the edge detection is interpreted as a relief
After watershed transformation
After basin merging with criteria A and B
After local watershed transf. and basin merging (criteria A and B)
After local watershed transf. and basin merging with criteria A-D. The basin layer defined by T’
a
b
c
d

Figure 9: Results of applying the watershed transformation. Image ***a*** shows the result of applying the standard algorithm, which obviously produces heavily over-segmented images. After merging the basins with the described criteria, more meaningful segmentation is achieved as shown in image ***b***. Image ***c*** shows the results achieved with the proposed local watershed transformation, starting with the selected ROI and applying the criteria A and B. Applying all of the proposed criteria (A-D) results in image ***d***. The basins in $\mathcal{B}_c$ are also shown in image ***d***.

and if the rule’s premise is ‘false’, the basin is definitely marked as not matching the user selection. Finally, if the condition of the above rule is true applying of the second set of thresholds T', the current basin is assigned to the layer of basins surrounding the user selection and processed as described next (see Figure 9 gray area in image ***d*** and Figure 10).

5 Basin Growth

After the user input is processed so far, the algorithm starts with the controlled basin growth. At this stage all basins completely or partially within the selected ROI and a one-pixel thick layer surrounding these basins are processed. When following the path of steepest descent for each of the pixels in this layer, the reached minima are completely outside the given ROI and thus not processed (flooded) yet. The regional minima of these basins, however, are known and the flooding process can be started as described in Section 2.1. In this way, in every basin-iteration a set of basins, surrounding the ones extracted in the previous step are computed.

Every time a new basin B is extracted, it’s attributes are applied for evaluating the merging rule with the threshold set T. If B matches the user selection, that is if the rule’s premise is true, it is assigned to the initial ROI. Unprocessed basins, adjacent to B are pushed on a queue $\mathcal{M}$ containing the yet unprocessed minima and are flooded in the next basin iteration. Otherwise, if B does not match the user selection, the merging rule is evaluated again with the thresholds T'. If the rule’s premise is true this time, the basin is assigned to a set of potential candidates for merging $\mathcal{B}_c$. Finally, in case the rule’s condition is false for both threshold sets T and T', the current basin is a *border basin*, not matching the initial user selection (see Figure 10). In the following, this algorithm is stated in pseudo-code:

```
/* $\mathcal{M}$ – contains yet unprocessed minima */
do
   $N \leftarrow$ card($\mathcal{M}$);
   while ($N > 0$)
      $N \leftarrow N - 1$;
      $M \leftarrow$ pop($\mathcal{M}$);
      $B \leftarrow$ flood($M, \mathcal{M}$);   /* basins adjacent to $B$ are added to $\mathcal{M}$ */
      if (merging_rule($T, B$))   /* check threshold set $T$ */
         assign_to_ROI($B$);
      else if (merging_rule($T', B$))   /* check threshold set $T'$ */
         assign_to_potential_ROI($B$);   /* assign to $\mathcal{B}_c$ */
      endif;
   end;   /* while ($N > 0$) */
while (card($\mathcal{M}$) $\neq N$)   /* until no new basins are added */
```

The mosaic of un-merged basins in $\mathcal{B}_c$ and their adjacency we utilized for the definition of an adjacency graph $\mathcal{G}$. In this graph the nodes correspond to the basins, the arcs link adjacent basins, similar to the approach introduced in [18]. The nodes within the initial ROI and the ones merged during the basin growth are pooled together in a node U (as depicted in Figure 10). The basins, for which the

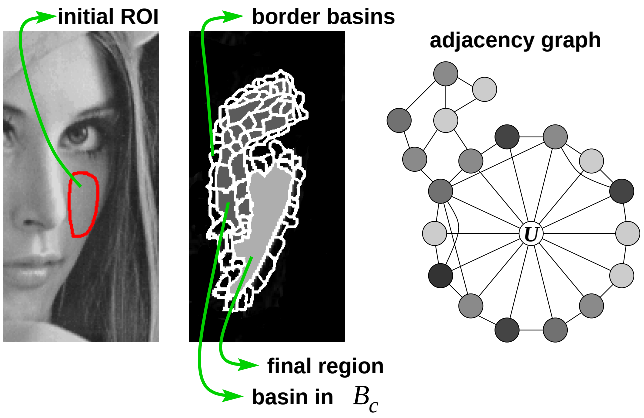


Figure 10: The user selection on the left and the extracted region in the middle. The graph on the right shows (a part of) the (colored) adjacent basins in $\mathcal{B}_c$.

merging rule returns true only for the second threshold set T', are also assigned to the graph. However, they are not merged with the node U, but are connected with each other and with U. Each of these basins contains the attributes listed at the end of Section 3. Thus, in a post-processing step, the user can manually assign basins out of $\mathcal{B}_c$ to the initial ROI. Furthermore, the threshold set T can be updated each time a basin is manually assigned to U. This makes a precise final basin-level-adjustment possible, without having to explicitely manipulate single parameters.

In both cases, the result of the proposed algorithm is a set of connected basins, extending the initial ROI. In other words, the input data is classified and marked in such a way, that there is a (connected) region of interest and an area not belonging to the ROI. In the 3D case, this data can be used to generate a surface surrounding it (see Color plate) for volume-rendering, or for ray-casting [17].

6 Results

To demonstrate the power of the proposed approach, we applied it on several images and two volume data sets. Each data set was first pre-processed and the standard Sobel edge detector [14] was applied on either the image data in the 2D case, or on each slice in the volume data. In the next step, a slice was selected out of the two volume data sets and a region was marked (as depicted in the Color plate). Afterwards, the algorithm described above was applied on a 26-connectivity grid. The computed results, consist of a set of marked pixels. In order to visualize this region, we extracted a surface applying the standard marching cubes algorithm [6].

The image data was processed in a similar way. After selecting a region and applying the proposed technique, the pixels defining the border between the marked region and the unmarked background, were colored red as shown in the lower row in the Color plate. The computation times are displayed in Table 1.

7 Future Work

The approach described above is implemented and works well. However, when the computation is completed, we currently perform surface extraction with the standard marching cubes algorithm [6], in order to visualize it. Unfortunately, this introduces 'stairs' in the surface. To solve this problem, we are working on basin extraction combined with surface extraction. Since borders between adjacent basin are in general smooth, this will improve the quality of the generated surface. In addition, the color values on the extracted region border can be used to assign appropriate color values to the surface. This will even increase the (photo-)realistic effect of the computed results.

Data set	Data size	Number of regions	Computation time
Jaw bone	300x400x50	36542	14.07 sec
Vessel tree	512x512x64	54263	21.6 sec
Peppers Region A	512x512	2320	2.8 sec
Peppers Region B	512x512	3859	3.91 sec
Peppers Region C	512x512	3619	3.34 sec

Table 1: Computation times for the example volumes and images measured on an SGI O2 with 180MHz R5000.

8 Summary

In this paper, we presented a new semi-automatic method for extracting regions of interest based on a local watershed transformation. We described how a catchment basin can be computed, given an initial pixel and a data set. The proposed technique does not require pre-processing of the data, while applying a modified rainfalling simulation. Unlike the standard watershed transformation, which floods the entire input data, the presented approach computes and merges only the catchment basins fulfilling a set of criteria. Therefore, we introduced four criteria for merging the catchment basins in a meaningful region, exploiting the attributes of the manually selected ROI.

Furthermore, the described technique can be applied more precisely, since it is easier to define merging criteria for basins within *one* object of interest in the input data, than for the entire data set. The merging rule introduced in this way, utilizes basin attributes like the basin's dynamics, it's average grey-color, and border characteristics like it's steepness and gradient magnitude. This significantly improves the results computed with the proposed technique, offering a reasonable solution of the segmentation problem addressed in the introduction.

References

[1] L. E. Band. Topographic partition of watersheds with digital elevation models. *Water Recources Res.*, 22(1):15–24, 1986.

[2] S. Beucher and C. Lantuéjoul. Use of watersheds in contour detection. In *International Workshop on Image Processing*, Rennes, Sept. 17-21 1979. CCETT/IRISA.

[3] S. Beucher and F. Meyer. The morphological approach to segmentation: the watershed transformation. In E. R. Dougherty, editor, *Mathematical Morphology in Image Processing*, chapter 12, pages 433–481. Marcel Dekker, New York, 1993.

[4] W. Higgins and E. Ojard. Interactive morphological watershed analysis for 3D medical images. *Computerized Medical Imaging and Graphics*, 17(4/5):387–395, 1993.

[5] Paul Thomas Jackway. Morphological multiscale gradient watershed image analysis. In G Borgefors, editor, *9th SCIA Scandinavian Conference on Image Analysis*, pages 87–94, 6-7 June 1995.

[6] W. E. Lorensen and H. E. Cline. Marching cubes: a high resolution 3D surface construction algorithm. In M. C. Stone, editor, *SIGGRAPH '87 Conference Proceedings (Anaheim, CA,*

July 27–31, 1987), pages 163–170. Computer Graphics, Volume 21, Number 4, July 1987.

[7] Alan P. Mangan and Ross T. Whitaker. Partitioning 3D Susface Meshes Using Watershed Segmentation. *IEICE Transactions on Visualization and Computer Graphics*, 5(4), October-December 1999.

[8] F. Meyer. Topographic distance and watershed lines. *Signal Processing*, 38(1):113–125, July 1994.

[9] F. Meyer and S. Beucher. Morphological segmentation. *Journal of Visual Communication and Image Representation*, 1(1):21–46, September 1990.

[10] A. N. Moga and M. Gabbouj. Parallel image component labeling with watershed transformations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 19(5):441–450, 1997.

[11] Alina N. Moga, Bogdan Cramariuc, and Moncef Gabbouj. A parallel watershed algorithm based on rainfalling simulation. In *European Conference on Circuit Theory and Design*, volume 1, pages 339–342, Istanbul, Turkey, August 1995.

[12] L. Najman and M. Schmitt. Watershed of a continuous function. *Signal Processing*, 38:99–112, 1994.

[13] L. Najman and M. Schmitt. Geodesic saliency of watershed contours and hierarchical segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 18(12):1163–1173, 1996.

[14] William K. Pratt. *Digital Image Processing*. Wiley-Interscience Publication, New York / Chichester / Brisbane / Toronto / Singapore, 1991.

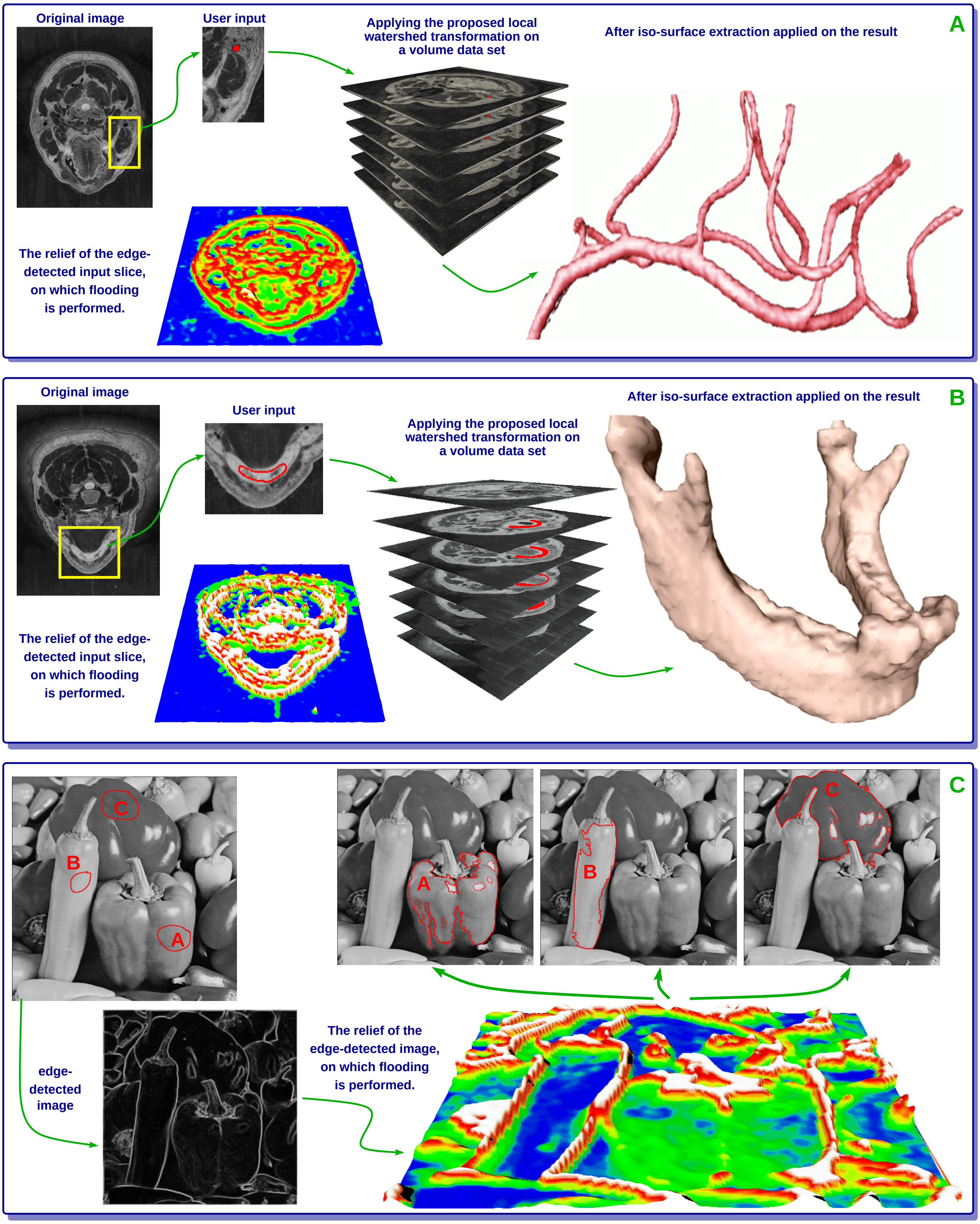
[15] J. Sijbers, P. Scheunders, M. Verhoye, A. Van der Linden, D. Van Dyck, and E. Raman. Watershed-based segmentation of 3D MR data for volume quantization. *Magnetic Resonance Imaging*, 15(4), 1997.

[16] Stanislav L. Stoev. Rafsi - a Fast Watershed Algorithm Based on Rainfalling Simulation. In *Proceedings of 8-th International Conference on Computer Graphics, Visualization, and Interactive Digital Media (WSCG'2000)*, volume 1, pages 100–107, Pilzen, Feb 2000.

[17] Ulf Tiede, Thomas Schiemann, and Karl Heinz Höhne. High quality rendering of attributed volume data (color plate p. 541). In *Proceedings of the 9th Annual IEEE Conference on Visualization (VIS-98)*, pages 255–263, New York, October 18–23 1998. ACM Press.

[18] Lee Vincent and Pierre Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE PAMI, 1991*, 13(6):583–598, 1991.

[19] S. Wegner, T. Harms, J. H. Builtjes, and H. Oswald. The watershed transformation for multiresolution image segmentation. *Lecture Notes in Computer Science*, 974:31–37, 1995.


A
Original image
User input
Applying the proposed local watershed transformation on a volume data set
After iso-surface extraction applied on the result
The relief of the edge-detected input slice, on which flooding is performed.
B
Original image
User input
Applying the proposed local watershed transformation on a volume data set
After iso-surface extraction applied on the result
The relief of the edge-detected input slice, on which flooding is performed.
C
C
B
A
A
B
C
edge-detected image
The relief of the edge-detected image, on which flooding is performed.

Results of applying the proposed local watershed transformation. The surfaces on the right of ***A*** and ***B*** are extracted with the marching cubes algorithm, after performing the basin growth approach.