An Updated Review on Watershed Algorithms

R. Romero-Zaliz and J.F. Reinoso-Gordo

Abstract Watershed identification is one of the main areas of study in the field of topography. It is critical in countless applications including sustainability and flood risk evaluation. Beyond its original conception, the watershed algorithm has proved to be a very useful and powerful tool in many different applications beside topography, such as image segmentation. Although there are a few publications reviewing the state-of-the-art of watershed algorithms, they are now outdated. In this chapter we review the most important works done on watershed algorithms, including the problem over-segmentation and parallel approaches. Open problems and future work are also investigated.

1 Introduction

One of the main topics in the field of topography are *watersheds* [90]. Knowing the right watersheds and their corresponding catchment basins is essential in countless applications in areas such as Civil Engineering, Hydrology, Environmental Science, Ecology, Limnology, Urban Planning, Agriculture and so on [2, 52, 66, 86]. For instance, determining areas where a flood risk exists can help experts to make a decision to forbid the urban construction in such areas [35]; studying the movement of water within the hydrological cycle need the use of catchment basins as units [59]; analyzing water quality inside lakes or reservoirs depends on several factors included at catchment basin scale [71]; determining territorial boundaries (e.g., such as in the case of Hudson Bay basin) require the use of watersheds [80], etc.

R. Romero-Zaliz (✉)
Department of Computer Science and Artificial Intelligence, University of Granada, Granada, Spain
e-mail: rocio@decsai.ugr.es

J.F. Reinoso-Gordo
Department of Architectonic and Engineering Graphic Expression, University of Granada, Granada, Spain
e-mail: jreinoso@ugr.es

C. Cruz Corona (ed.), *Soft Computing for Sustainability Science*,
Studies in Fuzziness and Soft Computing 358, DOI 10.1007/978-3-319-62359-7_12

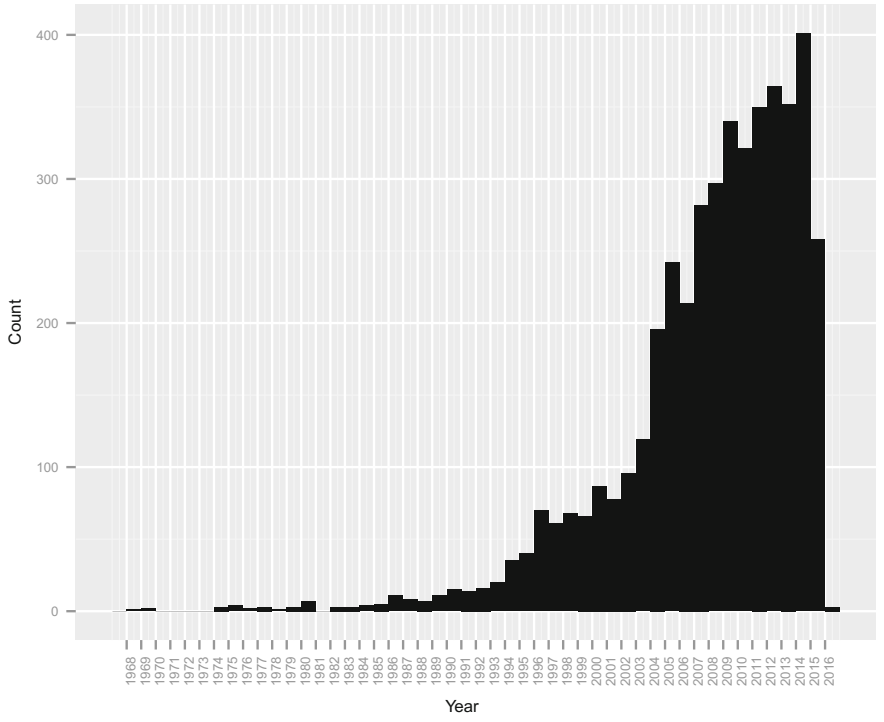


Fig. 1 Number of publications per year. Data obtained from Scopus © and CiteseerX © in October 13th, 2015

In the late '60 and '70s mathematicians and engineers started to work in the first algorithmic solutions for determining catchment basin's boundaries and watersheds [38, 43, 72]. Later on, researchers implemented a standardized version which they called the *watershed algorithm* [22]. The watershed algorithm has proved to be a very useful and powerful tool in many different application fields such as cartography [7, 32, 50, 65, 94], general image segmentation [61, 73, 87, 92, 97], video related issues [17, 19, 34, 95, 98], etc. There are several publications on biological and/or medical applications such as analysis of MRI images [3, 28, 46, 56, 77], cell images [4, 12, 14, 24, 85], ultrasound images [16, 33, 41, 45, 76], mammogram images [13, 23, 40, 79, 89], microscopy images [1, 6, 37, 42], among others.

Although there are a few publications reviewing the state-of-the-art of watershed algorithms [43, 60] and applications [18, 29, 49, 75], they are now outdated. In the last few years there has been an increasing number of publications in journals and conferences (Figs. 1 and 2).

Nowadays, most of the research in watershed algorithms are specifically devoted to image segmentation, but there are still some applications to real topographical watersheds, as shown in the world map of Fig. 3.

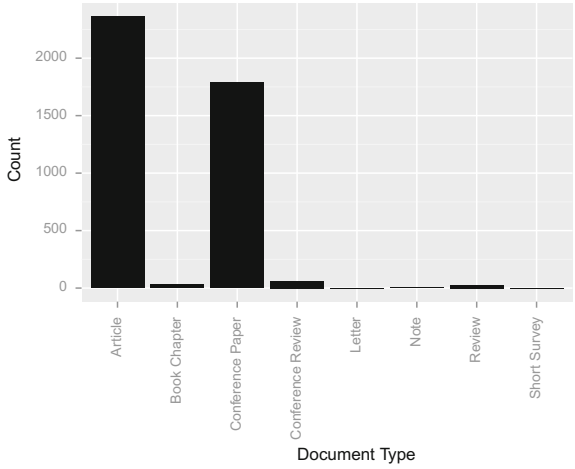


Fig. 2 Number of publications by document type

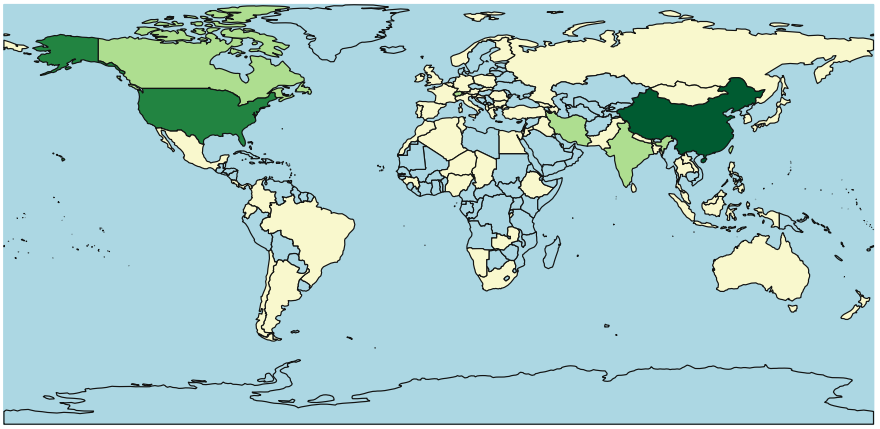


Fig. 3 Publications apply to topographic watersheds by country. The number of publications is color-coded from *white* to *green*, where *darker green* represent a higher number of publications in the given country

The rest of this survey is organized as follows. Section 2 introduces the basic common definitions used in the reviewed papers. Section 3 shows the main algorithms and strategies used for watershed determination. Section 4 is devoted to one of the main problems in watershed algorithms: over-segmentation. Section 5 reviews parallel approaches. Finally, Sect. 6 is reserved for conclusions and discussion.

2 Definitions

Each reviewed paper in this work uses different notation and terminology. Thus, we here propose a unified notation that will be used throughout this work in order to be able to study and compare the main algorithms and strategies cited in the bibliography.

Let us first consider a drop of water on a topographic surface. The water streams down, reaches a minimum of height and stops there. The set of all points of the surface, which the drops of water reaching this minimum can come from, can be associated with each minimum. Such a set of points is a **catchment basin** of the surface. The lines, which separate different catchment basins, are called **watersheds** or **watershed lines**.

Definition 1 We consider a topographic surface represented by a grid structure called *Digital Elevation Model (DEM)*, composed of cells, analogous to a digital image (IMG), composed of pixels. Every cell/pixel has a natural number as value representing heights on DEM or intensity on IMG of size $n \times m$.

$$DEM = \{x_{ij} \in \mathbb{R} | i \in \mathbb{N}, j \in \mathbb{N}, i \in (0, \ldots, n), j \in (0, \ldots, m)\} \quad (1)$$

$$IMG = \{x_{ij} \in \mathbb{N} | i \in \mathbb{N}, j \in \mathbb{N}, i \in (0, \ldots, n), j \in (0, \ldots, m)\} \quad (2)$$

For the sake of simplicity, from now on we will use the notation for topographic surface instead of the image's version in the rest of this manuscript.

Definition 2 Given a cell p defined as its position (i, j) in DEM, we define function I as $I(p) = x_{ij}$.

Definition 3 The set of neighborhood cells of p in the DEM is called $N_G(p)$ and collect all cells adjacent (adj) to p.

$$N_G(p) = \{p' \in I | adj(p, p')\} \quad (3)$$

Definition 4 A path P of length l between two cells p and p' in DEM is a $(l+1)$-tuple of adjacent cells $(p_0, p_1, \ldots, p_{l-1}, p_l)$ such that $p_0 = p$, $p_l = p'$.

Definition 5 Cells belonging to the same connected plateau CP must satisfy the following condition

$$\forall p, p' \in CP, \exists P = (p0, p_1, \ldots, p_l) | p_0 = p \wedge p_l = p' \wedge \forall p_i \in P, I(p_i) = I(p) = I(p') \quad (4)$$

Definition 6 A minimum area M of DEM is a connected plateau of cells from which it is impossible to reach a cell of lower altitude without having to climb (Fig. 4).

$$M = \{p | \forall p' \notin CP(p) \wedge I(p') \leq I(p) \rightarrow \forall P = (p0, p_1, \ldots, p_l) | p_0 = p \wedge p_l = p', \exists i \in [1, l] | I(p_i) \geq I(p_0)\} \quad (5)$$

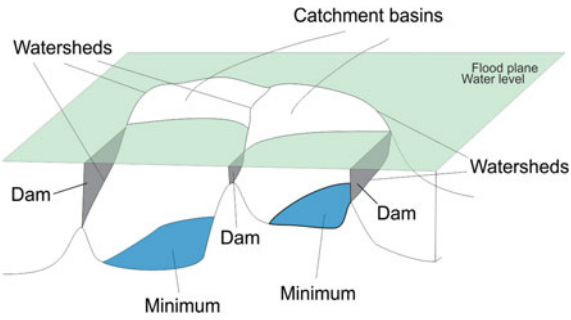


Fig. 4 Connected plateau

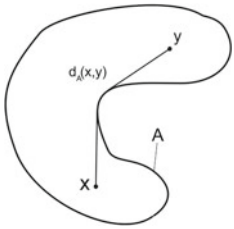


Fig. 5 Geodesic distance

For a particular altitude h we denote the minimum area as M_h.

Definition 7 $T_h(I)$ stands for the threshold of I at level h where h is the value taken by I and $h \in [h_{min}, h_{max}]$.

$$T_h(I) = \{p | I(p) \leq h\} \tag{6}$$

Definition 8 The geodesic distance $d_A(p, p')$ between two cells p and p' in A is the minimun of the length of the paths which join p and p' and are totally included in area A (Fig. 5).

$$d_A(p, p') = min(\{length(P), P = (p, \ldots, p') \text{ which is totally included in } A\}) \tag{7}$$

Definition 9 Let A and B be two sets of cells of a given DEM (Fig. 6), where $B \subset A$. B is composed by k areas not adjacent: $B_1, B_2, \ldots, B_k$ (black areas in Fig. 6) but connected through A. The geodesic influence zone $IZ_A(B_i)$ of a component of B in A is the locus of the cells of A whose geodesic distance to B_i is smaller than their geodesic distance to any other component of B (blue color in Fig. 6 is the $IZ_A(B_1)$).

$$IZ_A(B_i) = \{p \in A, \forall j \in [1, k] \wedge j \neq i | d_A(p, B_i) < d_A(p, B_j)\} \tag{8}$$

Definition 10 The cells of A which do not belong to any geodesic influence zone constitute the skeleton by influence zones (SKIZ) of B inside A, denoted $SKIZ_A(B)$

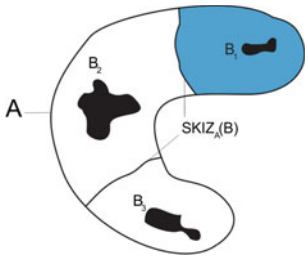


Fig. 6 Geodesic influence zone. A small example with $k = 3$

$$SKIZ_A(B) = A \setminus IZ_A(B) \text{ with } IZ_A(B) = \bigcup_{i \in [1,k]} IZ_A(B_i) \tag{9}$$

Definition 11 The set of catchment basins of DEM is equal to the set $X_{h_{max}}$ composed of not adjacent areas and obtained after the following recursion

$$\begin{cases} X_h = T_h(I) \ , \ h = h_{min} \\ X_{h+1} = M_{h+1} \cup IZ_{T_{h+1}(I)}(X_h) \ , \ h \in (h_{min}, h_{max} - 1] \end{cases} \tag{10}$$

Definition 12 The watersheds are $X^C_{h_{max}}$, i.e. the complement set of $X_{h_{max}}$.

3 Algorithms

Naturally, the first algorithms for computing watersheds are found in the field of topography. Lets recall that topographic surfaces are numerically handled through DEMs, these are arrays of numbers that represent the spatial distribution of terrain altitudes [90]. In image segmentation, its main application, the idea of the watershed construction is quite simple: a gray scale image can be considered as a topographic relief, the gray scale value of a pixel being the altitude at that particular point [69]. Using this analogy we can now review all papers regardless of its application.

There are a few reviews devoted to watershed algorithms in the bibliography. The first one appeared in the early 90s, when Beucher and Meyer publish a book chapter introducing what they have called the "watershed transformation" (basically the flooding process explained below) along with the principles of morphological segmentation and morphological tools [9]. This transformation was rapidly adopted by many other researchers for their own particular applications [30, 96].

Later on in 2003, Najman and Couprie study the behavior of watershed algorithms. Through the introduction of the concept of "pass value" they show that most classical watershed algorithms do not allow the retrieval of some important topological features of the image [60]. An important consequence of this result is that it is not possible to compute sound measures such as depth area or volume of basins using most classical watershed algorithms.

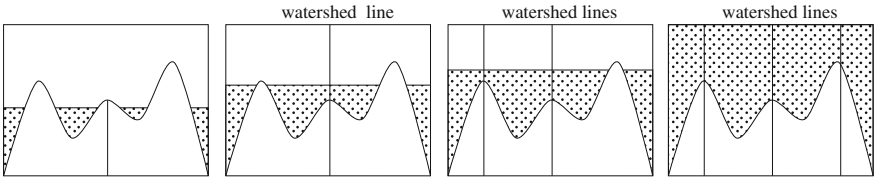


Fig. 7 Immersion-based watersheds

Finally and more than 5 years later, Körbes and Lotufo present a communication in a symposium reviewing fifteen watershed algorithms in a comprehensive way [43]. Afterwards several novel algorithms were developed and thus needed an updated review.

But let us first introduce the two main strategies for determining watersheds, we will study each of them separately in the next subsections.

3.1 By Immersion

The first strategy that we will analyze is called immersion (also called flooding). It was first developed for contour detection in images and introduced by Beucher and Lantuejoul in 1979 [8].

Later, an algorithmic-based definition for the identification of watersheds by immersion was introduced by Vincent and Soille in 1991 [90]. By analogy to the idea of immersion, we can figure that we have pierced holes in each regional minimum of our topographic surface. We then slowly immerse our surface into a lake. Starting from the minima of lowest altitude, the water will progressively fill up the different catchment basins. Now, at each position where the water coming from two different minima would merge, we build a "dam". At the end of this immersion procedure, each minimum is completely surrounded by dams, which delimit its associated catchment basin. The whole set of dams which has been built provides a division of the surface in its different catchment basins. These dams correspond to the watersheds of our surface [90] (Fig. 7). Vincent and Soille algorithm was developed for image segmentation and involves two major steps: (1) sorting of pixels in increasing order of gray value (the gray value of a pixel being the altitude at a particular point), and (2) fast computation of geodesic influence zones by breadth-first scanning of each threshold level using a first-in-first-out (FIFO) data structure (Algorithm 1).

Meijster and Roerdink propose in 1998 [51] an algorithm with two stages: (1) transform a lower complete image using a FIFO-queue algorithm, and (2) calculate the watershed using graph theory and removing the old FIFO-queue (Algorithm 4).

Lotufo and Falcao, in 2000, reviews the watershed in the graph framework of a shortest-path forest problem using a lexicographic path cost formulation. This

Algorithm 1 Vincent and Soille's immersion watershed approach [90]

Require: I_i ▷ I_i *original image of size* $n \times m$
Ensure: I_o ▷ I_o *image of the labeled watersheds*

MASK $\leftarrow -2$
WSHED $\leftarrow 0$
$queue \leftarrow \emptyset$
for all $p \in I_o$ **do**
  $I_o(p) \leftarrow -1$
end for
$current\_label \leftarrow 0$
$dist \leftarrow 0$
Initialize each pixel of I_d with 0 ▷ I_d *work image of distances*
Sort the pixels of I_i in the increasing order of their gray values
$h_{min} \leftarrow min(I_i)$
$h_{max} \leftarrow max(I_i)$

for $h \leftarrow h_{min}$ **to** h_{max} **do** ▷ *geodesic SKIZ of level* $h-1$ *inside level* h
  for all p such that $(I_i(p) = h)$ **do** ▷ *pixels accessed through the sorted array*
    $I_o(p) \leftarrow$ MASK
    if $(\exists p' \in N_G(p) | (I_o(p') > 0)$ **or** $(I_o(p') =$ WSHED$))$ **then**
      $I_d(p) \leftarrow 1$
      fifo_push($queue$, p)
    end if
  end for
  $dist \leftarrow 1$
  fifo_push($queue$, p^0) ▷ p^0 *is a fictitious pixel*
  repeat
    $p \leftarrow$ fifo_get($queue$)
    if $(p = p^0)$ **then**
      if $queue \neq \emptyset$ **then**
        fifo_push($queue$, p^0)
        $dist \leftarrow dist + 1$
        $p \leftarrow$ fifo_get($queue$)
      end if
    end if

formulation reflects the behavior of the ordered queue-based watershed algorithm [47].

In 2001, Chen and Shi [15] modify the original Vincent and Soille immersion-based watershed algorithm to correct some issues: (1) incorrect labeling when a point p is at the same distance from three or more adjacent catchment basins (i.e., it will be labeled as catchment basin instead as watershed), (2) unnecessary computation of geodesic distance between points which belong to the labeled, (3) memory consumption, and (4) incapacity to obtain information about catchment basins during the processing. The modified algorithm proposed introduces a third step call "gushing

```
32:     for all p' ∈ N_G(p) do                    ▷ p' belongs to an already labeled basin or to the
33:       if ((0 < I_d(p') < dist) or (I_o(p') = WSHED)) then                watershed
34:         if (I_o(p') > 0) then
35:           if ((I_o(p) = MASK) or (I_o = WSHED)) then
36:             I_o(p) ← I_o(p')
37:           else if (I_o(p) ≠ I_o(p')) then
38:             I_o(p) ← WSHED
39:           end if
40:         else if (I_o(p) = MASK) then
41:           I_o(p) ← WSHED
42:         end if
43:       else if ((I_o(p') = MASK) and (I_d(p') = 0)) then
44:         I_d(p') ← dist + 1
45:         fifo_push(queue, p')
46:       end if
47:     end for
48:   until (queue = ∅)
49:   for all p|(I_i(p) = h) do                   ▷ checks if new minima have been discovered
50:     I_d(p) ← 0                                ▷ the distance associated with p is reset to 0
51:     if (I_o(p) = MASK) then
52:       current_label ← current_label + 1
53:       fifo_push(queue, p)
54:       I_o(p) ← current_label
55:       while (queue ≠ ∅) do
56:         p' ← fifo_get(queue)
57:         for all p'' ∈ N_G(p') do
58:           if (I_o(p'') = MASK) then
59:             fifo_push(queue, p'')
60:             I_o(p'') ← current_label
61:           end if
62:         end for
63:       end while
64:     end if
65:   end for
66: end for
```

step", that together with the immersion step, produces a fast recognition of pixels of labeled and gets the flood level of the catchment basin.

Later on, Rambabu et al. first, in 2003, propose a new algorithm based on hill climbing simulation, that avoided multiple scanning of the original matrix by using different queues to store pixels [69]. Then, in 2008, Rambabu and Chakrabarti gives an updated and corrected version of this same algorithm [70].

In 2005, Shen and Chang [74] present a nearest-neighbor graph (NNG) based watershed algorithm. The main idea behind this work is to transform the image into a NNG and then partitioned by discovering the defined geographic features in the

Algorithm 2 Meijster and Roerdink's Lower_complete function

Require: I_i ▷ I_i *original image of size* $n \times m$
Ensure: I_{lc} ▷ I_{lc} *lower complete image*

```
1: queue ← ∅
2: for all p ∈ I_i do
3:   I_lc(p) ← 0
4:   if ∃p' ∈ N_G(p)|(I_i(p') < I_i(p)) then
5:     fifo_push(queue, p)
6:     I_lc ← −1
7:   end if
8: end for
9: dist ← 1
10: fifo_push(queue, p^0)                    ▷ p^0 is a fictitious pixel
11: while (queue ≠ ∅) do
12:   p ← fifo_get(queue)
13:   if (p = p^0) then
14:     if (queue ≠ ∅) then
15:       fifo_push(queue, p^0)
16:       dist ← dist + 1
17:     end if
18:   else
19:     I_lc ← dist
20:     for all p' ∈ N_G(p)|((I_i(p') = I_i(p)) and (I_lc(p') = 0)) do
21:       fifo_push(queue, p')
22:       I_lc(p') ← −1                      ▷ to prevent from queuing twice
23:     end for
24:   end if
25: end while
26: for all p ∈ I_i|(I_lc(p) ≠ 0) do
27:   I_lc ← dist × I_i(p) + I_lc(p) − 1
28: end for
```

first step. The initial population result is also transformed into the NNG again and the recursively distilled by the proposed algorithm.

3.2 By Rainfall

The second strategy is called rainfall and has two main steps: (1) the weakest edges are removed by "drowning" the image, creating a number of "lakes" grouping all the pixels that lie below a certain threshold (this is useful to reduce the influence of noise, and reduces the over-segmentation), and (2) the direction of a raindrop from each pixel would flow if it would fall on the topographic activity surface. This steepest descent neighbor and the pixel under consideration are then merged, finally enabling the localization of the remaining edges and segments [21] (Fig. 8).

Algorithm 3 Meijster and Roerdink's Resolve function [51]

Require: p ▷ *p input pixel*
Ensure: ce ▷ *ce canonical element (label assigned to a minimum) of p or WSHED in case p lies on a watershed*

```
Create array sln where sln[p, i] is a pointer to the i-th steepest lower neighbor of pixel p
WSHED ← −1
i ← 1
ce ← 1
while (i ≤ 4) and (ce ≠ WSHED)) do
    if ((sln[p, i] ≠ p) and (sln[p, i] ≠ WSHED)) then
        sln[p, i] ← Resolve(sln[p, i])
    end if
    if (i = 1) then
        ce ← sln[p, 1]
    else if (sln[p, i] ≠ ce) then
        ce ← WSHED
        for j ← 1 to 4 do
            sln[p, i] ← WSHED
        end for
    end if
    i ← i + 1
end while
```

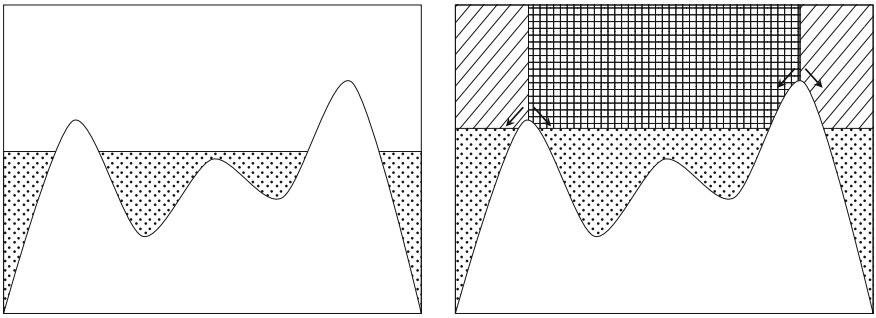
Algorithm 4 Meijster and Roerdink's immersion watershed approach [51]

Require: I_i ▷ *I_i original image of size $n \times m$*
Ensure: I_o ▷ *I_o image of the labeled watersheds*

```
WSHED ← 0
I_lc ← Lower_complete(I_i)        ▷ transform the image (Algorithm 2)
for all p ∈ I_lc do
    I_o(p) ← Resolve(p)           ▷ see Algorithm 3
end for
```

Mortensen and Barret in 1999 introduces an optimization variation for the rainfall-based watershed algorithm using a tobogganing technique, which makes a much more computationally efficient algorithm [57]. In this version, tobogganing over-segments an image into small regions by sliding in the derivative terrain. The basic idea is that given the gradient magnitude of an image, each pixel determines a slide direction by finding the pixel in a neighborhood with the lowest gradient magnitude. Pixels that "slide" to the same local minimum are grouped together, thus segmenting the image into a collection of small regions.

A year later, Bieniek and Moga [11] propose an efficient algorithm based on connected components that generates the same results as the original Meyer's algorithm [53] but with a simpler algorithmic construction and, hence, a lower complexity (it can label all catchment basins by only scanning the image four times).



(a) drowning threshold (b) steepest descent rain falling principle

Fig. 8 Rainfall-based watersheds

Later on, in 2007, Osma-Ruiz et al. implement a more efficient algorithm by computation of shortest paths [63] (Algorithm 5). This algorithm produces the same result as the previous works using only two scans (plus another to initialize the data structures), thus decreasing the running time obtained in [11, 81].

One of the latest work in the area was developed by Świercz and Iwanowski in 2010 [82]. In their work, a mechanism called directional code is used to code descent paths, where each visited pixel receives a temporary marking in the output label array. The value of this temporary marking represents the position of pixel in the image. A path can be therefore viewed as a series of pointers to pixels stored in the output array.

3.3 Mixed Approaches

In 2005 Sun et al. modifies Bieniek and Moga's work simulating raining to generate the connected components using chain code instead of pixel address. Afterwards, they simulate flooding to label catchment basins by tracing chain codes [81]. This algorithm not only can label catchment basins by scanning the image only four times, but also is more helpful to the following image processing.

Cousty et al. in 2009 introduces the first work that mathematically prove equivalence to both immersion-based and rainfall-based watersheds [20]. For this purpose, the authors propose a new definition of watershed, called *watershed cut* and give a liner-time algorithm to compute the watershed cuts of an edge-weighted graph (pre-processed image). The proposed algorithm does not require any sorting step or the use of any sophisticated data structure such as a hierarchical queue or a representation to maintain unions of disjoint sets.

Algorithm 5 Osma-Ruiz's rainfall watershed approach [63]

Require: I_i ▷ I_i *original image of size* $n \times m$
Ensure: I_o ▷ I_o *image of the labeled watersheds*

UNVISITED $\leftarrow -8$ ▷ *Step 1: Initialization*
PENDING $\leftarrow -9$
for all $p \in I_i$ **do**
  $I_o(p) \leftarrow$ UNVISITED
end for
$qPending \leftarrow \emptyset$
$qEdge \leftarrow \emptyset$
$qInner \leftarrow \emptyset$
$qDescending \leftarrow \emptyset$

$ncatch \leftarrow 1$ ▷ *Step 2: Identifying regional minima and steepest descending paths*
for all $p \in I_i$ **do**
  if $(I_o(p) =$ UNVISITED$)$ **then** ▷ *if the point has not been analyzed yet, study it*
    for all $p' \in N_G(p)$ **do**
      if $(I_i(p) = I_i(p'))$ **then** ▷ *this is a plateau*
        if $(qPending = \emptyset)$ **then**
          $I_o(p) \leftarrow$ PENDING
          queue_push$(qPending, p)$
        end if
        $I_o(p') \leftarrow$ PENDING
        queue_push$(qPending, p')$
      else if $(I_i(p') = min(I_i(N_G(p))))$ **then**
        $min \leftarrow p'$
      end if
    end for
    if $(qPending \neq \emptyset)$ **then** ▷ *if p belongs to a plateau*
      while $(qPending \neq \emptyset)$ **do** *make it lower-complete image*
        $p' \leftarrow$ queue_pop$(qPending)$ *if not, p is considered a minimum*
        $min \leftarrow \emptyset$ *unless there is a lower neighbor*
        if $(p \neq p')$ **then** ▷ *calculations already done for seed*
          for all $p'' \in N_G(p')$ **do** ▷ *put in the queue all the points in the plateau*
            if $(I_i(p'') = I_i(p))$ **then**
              if $(I_o(p'') =$ UNVISITED$)$ **then**
                $I_o(p'') \leftarrow$ PENDING
                queue_push$(qPending, p'')$
              end if

```
36:           else if (I_i(p'') = min(I_i(N_G(p')))) then min ← p''
37:           end if
38:         end for
39:       end if
40:       if (min ≠ ∅) then                          ▷ classify p' as either an edge or inner point
41:         min ← I_o(p')
42:         queue_push(qEdge, p')
43:       else
44:         queue_push(qInner, p')
45:       end if
46:     end while
47:     if (qEdge ≠ ∅) then                          ▷ if the plateau has no edge points,
48:       if (qInner ≠ ∅) then                         it is a minimum
49:         while (qEdge ≠ ∅) do                       else, make it lower complete
50:           p' ← queue_pop(qEdge)
51:           for all p'' ∈ N_G(p') do
52:             if ((I_i(p'') = I_i(p)) and (I_o(p'') = PENDING)) then
53:               I_o(p'') ← p'
54:               queue_push(qEdge, p'')
55:             end if
56:           end for
57:         end while
58:       end if
59:     else
60:       while (qInner ≠ ∅) do
61:         p' ← queue_pop(qInner)
62:         I_o(p') ← ncatch
63:       end while
64:       ncatch ← ncatch + 1
65:     end if
66:   else
67:     if (min = ∅) then
68:       I_o(p) ← ncatch
69:       ncatch ← ncatch + 1
70:     else
71:       min ← I_o(p)
72:     end if
73:   end if
74: end if
75: end for
```

```
76: for all p ∈ I_i do                          ▷ Step 3: Assignment of pixels to catchment basins
77:   p' ← p
78:   while (I_o(p') ≤ 0) do                    ▷ it is not a minimum
79:     queue_push(qDescending, p')
80:     p' ← p'_ref                             ▷ p'_ref point pointed to by p'
81:   end while
82:   while (qDescending ≠ ∅) do
83:     p'' ← queue_pop(qDescending)
84:     I_o(p'') ← I_o(p')
85:   end while
86: end for
```

4 Over-Segmentation

Researcher noticed that it was hard to apply watershed transformations since they are very sensitive to noise. One of the main drawbacks of the classical watershed algorithm is a phenomenon known as over-segmentation. There are several approaches to reduce the impact of this issue, which can be categorized into two approaches: *pre-processing* and *post-processing*. Several pre-processing and post-processing methods were reviewed by Bieniecki in 2004 applied to color images. Between the studied pre-processing methods, the author analyze noise removal by using a median filter, color morphology and other smoothing filters. Between the post-processing methods, the author research merging basins by gradient watersheds on graphs, basin dynamics, inclusionary and exclusionary cues, image component labeling and multi-scale gradient analysis [10].

4.1 Pre-processing

The most efficient pre-processing techniques are based on markers. In a marker-based algorithm, the gradient image is first modified by a marker image, which is a binary image with the object interiors (markers) being set to 0 and the uncertainty areas being set to 255 (Fig. 9). Each marker indicates the presence of an object [25].

In Moga and Gabbouj watershed transformation is augmented to perform with the aid of a priori supplied image markers. In this method pixels are first clustered based on spatial proximity and gray level homogeneity with the watershed transformation. Boundary-based region merging is the effected to condense non-marked regions into marked catchment basins The agglomeration strategy works with a weighted neighborhood graph representation of the over-segmented image [55].

Some years later Gao et al. propose a marker-based algorithm using a disjoint set data structure with a linear complexity [25]. Later on they present an updated algorithm to extract the regional minima from the low frequency components in the gradient map. The extracted minima constitute the binary marker image. Then the

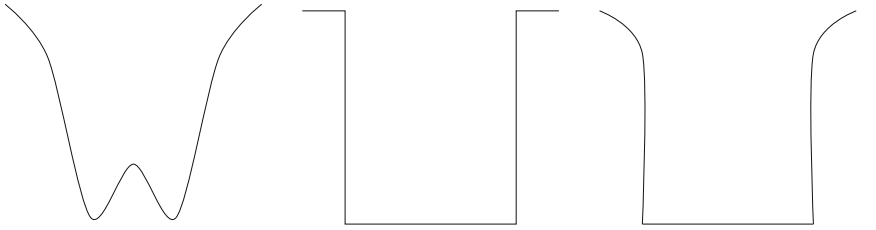


Fig. 9 Modification process of the gradient image by the marker image

original gradient map is modified by suppressing its all-intrinsic minima around these extracted markers. Thus, compared with traditional approaches, both the spurious minima are more effectively removed and meanwhile the boundaries of objects are more effectively protected [26].

In 2009, Zhu et al. apply a multi-scale alternating sequential filtering by reconstruction to simplify the input image in order to remove local minima which are caused by irregular gray disturbance and noise, and preserve important contour information [99]. After this procedure is done, there still exists some local minima problem which is reduced by a marker-extracted method that uses minima imposition to make a marked image before watershed transformation. Markers are a set of components marking flat regions of an image. The mark extraction can suppress all of is intrinsic minima. Finally, the watershed algorithm is applied to the modified gradients by the markers.

A stochastic version of the watershed algorithm is obtained by choosing randomly in the image the seeds from which the watershed regions are grown. In the 2009's work Tolosa et al. explore two seed-generation processes to avoid over-segmentation. The first is a non-uniform Poisson process, the density of which is optimized on the basis of the opening granulometry. The second process positions the seed randomly within disks centered on the maxima of a distance map [84].

Procházka et al. proposes in 2010 a smoothing procedure to reduce noise and over-segmentation. This procedure first applies wavelet image de-noising, and afterwards a smoothing phase. The main idea in this phase is to remove image elements smaller than the size of structuring element and to fill gaps between pixels and smoothes their outer edges [68].

Also in 2010, Moumoun et al. introduces a filtering step to eliminate insignificant minima that take into account not only the depth of the pixel but also the change in concavity [58]. This is done by defining a hierarchy between minima using a generic graph. In this graph each region is represented by a node. The minimum curvature of a region is associated with the corresponding node. The arcs express the adjacencies relations between regions and two regions are called adjacent if they have adjacent faces. The weight of each arc is defined by the depth measurement already mentioned.

4.2 Post-processing

Gies and Bernard propose in 2004 a merging phase of basin using statistical information. With regions of an unspecified size, the merging criterion must follow statistical rules accounting for the region size. The authors consider the regions as the outcome of stochastic processes. The merging criterion is based on the knowledge of region area and statistical regional measures, determining the statistical reliability of the merging [27].

Patino publishes in 2005 an approach that characterizes each of the segmented regions and then employs the composition of fuzzy relations to group together similar regions. A fuzzy c-means algorithm along with fuzzy relations can group together similar gray-level values, but only between adjacent regions [64].

Jianhua et al. propose a more complex strategy for basin merging in 2005. Their method pre-segments the image by watersheds and then merges it by Immune Clonal Algorithm (ICA). To implement the task, several operators are proposed such as the DC operator, the Proportional Creation of the First generation operator, and fitness function based on JND and average gray value [36].

Moumoun et al. also propose a post-processing technique based on region merging by the depth of the watershed segmentation. This depth is defined by the difference between the height of the saddle point and the minimum of adjacent regions [58].

4.3 Other Approaches

There are very few approach to reduce over-segmentation that do not use pre-processing or post-processing. Swiercz and Iwanowsky propose in 2011 a "water-ball method designed to counter over-segmentation during the actual calculation of watersheds. This proposal can be perceived as a composite method for object extraction, combining several techniques and mechanisms to produce satisfactory macro-scale results [83]. The "water-ball method uses two distinct mechanisms. The first one consists of a "rolling ball" based on the simulation of a larger object rolling down the slope (contrary to classic rainfall methods where a single drop of water is used). This ball has the ability to cross small ridges and ignore small, insignificant local minima. The second mechanism, weakening with edge enhancement, makes it possible to eliminate low, insignificant interior boundaries.

5 Parallel Approaches

When watershed algorithms are applied in large images or big DEMs, serial versions can take many hours to return the solution. therefore, researchers dedicated their effort to implement parallelized versions, but do to the recursive nature of the watershed

transformation, its parallelization is not a trivial task [54] and cannot suit real-time operations in most cases [62].

5.1 Parallelization Using Distributed Memory

Most of the existing papers on parallel watershed algorithms are based on a divide-and-conquer approach [54, 55, 67, 72, 93] with regular domain decomposition into blocks of data. Each block is then processed by a different processor independently, thus using distributed memory. Afterwards all blocks are merged together. In 2010 Świercz and Iwanowski [82] present an interesting approach to reduce the merging process by pre-calculating the adjacent sections between the different blocks. Thus, the merging process reduces to a bulk memory copy of each processed block.

5.2 Parallelization Using Shared Memory

There are a few publications that uses a shared memory approach where there is a need of constant synchronization between the processors that process adjacent blocks. In Wagner et al. [91] the authors propose a chromatic ordering that allows to gain a correct segmentation without an examination of adjacent domains or a final relabeling. Later, VanNeerbos et al. [88] parallelize topological watersheds in such a way that border pixels between threads are not calculated at the same time. To do this, each thread process its tile in different stages and synchronizing between all threads is performed after each stage. Also in 2011 Mahmoudi et al. [48] introduce an adapted parallelization strategy called split, distribute and merge strategy which allows efficient parallelization of a large class of topological operators including, mainly, smoothing, skeletonization, and watershed algorithms. To achieve a good speedup they focus on task scheduling.

5.3 Parallelization Using Graphics Processor Units

There is a rising tendency in research in parallelization using Graphics Processor Units (GPUs) and watershed algorithms are not an exception. Kauffmann and Piche [39] describe a cellular automaton (CA) to perform the watershed transform in N-D images. Due to the local nature of CA algorithms they show that they are designed to run on massively parallel processors and therefore, be efficiently implemented on low cost consumer GPUs. A few years later Körbes et al. review the advances in watershed processing on GPU architecture [44] on two algorithms: one inspired by the drop of water paradigm and depth-search approaches; and one based on cellular automata to process a shortest-path forest with sum cost function. In 2012 Hucko

and Srámek [31] present the first GPU watershed algorithm able to process data larger than the available memory, as the whole data has to be present in the memory of the device. In their manuscript they present two versions of a streamed multi-pass algorithm for watershed computation on a GPU. As the slice-based streaming approach is used both variants are capable of processing data exceeding the size of the available graphics accelerator memory. Another interesting approach is shown by Quedada-Barriuso et al. [5] where the watershed transform based on a cellular automaton, especially when the synchronization rules are relaxed. In particular, they compare a synchronous and an asynchronous implementation of the algorithm. One of the main applications for watersheds are medical related issues, therefore Smistad et al. recently published a comprehensive survey on medical image segmentation on GPUs [78].

6 Conclusions

Recent advances in watershed algorithms focus on the use of different data structures to improve the efficiency of the proposed algorithms. Most of the work done has been developed for image segmentation purposes, specially on medical and biological images. Also, parallel approaches seem to be an area of constant development. Curiously, many of the most interesting proposals were published in conferences instead of in journals. This tendency seems to change in the latest years of research.

Although there are many different watershed algorithmic solutions, there are still many problems to solve. Thus, there is an increasing amount of publications over the last few years. That is, watershed algorithms are still an open problem and there are more and more new fields starting to use and develop this kind of techniques from different points of view.

Another open issue is the oversegmentation watershed algortihms produce. Although there are several approches to minimize this problem, there is still no perfect solution available yet. Oversegmentation can be an important issue when determining countours on images that will be used for counting purposes (such as blood cells).

Many applications that use watershed algorithms work with a small dataset or datasets. For real-world applications in topography that includes large regions of terrain, watershed algorithms became slow and inefficient. Sometimes they are not capable to work since memory becomes a crucial issue. For this kind of big data processes the only available solution is the partition of the dataset into smaller sets and the posterior merge of each partial solution.

We believe there is still a lot of improvement to be developed in this field. For instance, there is a lack of techniques for real-time and streaming applications. Also, we could not retrieve many studies on real watersheds in many countries, thus indicating that many catchment basins worldwide have not been studied yet.

References

1. Ancin, H., Roysam, B., Dufresne, T., Chestnut, M., Ridder, G., Szarowski, D., Turner, J.: Advances in automated 3-d image analysis of cell populations imaged by confocal microscopy. Cytometry **25**(3), 221–234 (1996)
2. Andreatta, S., Wallinger, M., Posch, T., Psenner, R.: Detection of subgroups from flow cytometry measurements of heterotrophic bacterioplankton by image analysis. Cytometry **44**(3), 218–225 (2001)
3. Angel Viji, K., Jayakumari, J.: Performance evaluation of standard image segmentation methods and clustering algorithms for segmentation of mri brain tumor images. Euro. J. Sci. Res. **79**(2), 166–179 (2012)
4. Arslan, S., Ozyurek, E., Gunduz-Demir, C.: A color and shape based algorithm for segmentation of white blood cells in peripheral blood and bone marrow images. Cytometry Part A **85**(6), 480–490 (2014)
5. Barriuso, P.Q., Heras, D., Argüello, F.: Efficient gpu asynchronous implementation of a watershed algorithm based on cellular automata, pp. 79–86, (2012)
6. Becattini, G., Mattos, L., Caldwell, D.: A novel framework for automated targeting of unstained living cells in bright field microscopy. pp. 195–198 (2011)
7. Beucher, S., Bilodeau, M.: Road segmentation and obstacle detection by a fast watershed transformation. pp. 296–301 (1994)
8. Beucher, S., Lantuéjoul, C.: Use of watersheds in contour detection. workshop published (1979)
9. Beucher, S., Meyer, F.: The morphological approach to segmentation: the watershed transformation. mathematical morphology in image processing. Opt. Eng. **34**, 433–481 (1993)
10. Bieniecki, W.: Oversegmentation avoidance in watershed-based algorithms for color images. pp. 169–172 (2004)
11. Bieniek, A., Moga, A.: An efficient watershed algorithm based on connected components. Pattern Recognit. **33**(6), 907–916 (2000)
12. Bullet, J., Gaujoux, T., Borderie, V., Bloch, I., Laroche, L.: A reproducible automated segmentation algorithm for corneal epithelium cell images from in vivo laser scanning confocal microscopy. Acta Ophthalmol. **92**(4), e312–e316 (2014)
13. Camilus, K., Govindan, V., Sathidevi, P.: Pectoral muscle identification in mammograms. J. Appl. Clin. Med. Phys. **12**(3), 215–230 (2011)
14. Chen, L.-C., Nguyen, T.-H., Lin, S.-T.: Viewpoint-independent 3d object segmentation for randomly stacked objects using optical object detection. Meas. Sci. Technol. **26**(10) (2015)
15. Chen, T.: Gushing and immersion alternative watershed algorithm, pp. 246–248 (2001)
16. Cheng, J.-Z., Chen, K.-W., Chou, Y.-H., Chen, C.-M.: Cell-Based Image Partition and Edge Grouping: A Nearly Automatic Ultrasound Image Segmentation Algorithm for Breast Cancer Computer Aided Diagnosis. vol. 6915 (2008)
17. Chien, S.-Y., Chen, L.-G.: Reconfigurable morphological image processing accelerator for video object segmentation. J. Signal Process. Syst. **62**(1), 77–96 (2011)
18. Christ, M.., Parvathi, R.: Segmentation of medical image using clustering and watershed algorithms. Am. J. Appl. Sci. **8**(12), 1349–1352 (2011)
19. Chung, K.-L., Lai, Y.-S., Huang, P.-L.: An efficient predictive watershed video segmentation algorithm using motion vectors. J. Inf. Sci. Eng. **26**(2), 699–711 (2010)
20. Cousty, J., Bertrand, G., Najman, L., Couprie, M.: Watershed cuts: Minimum spanning forests and the drop of water principle. IEEE Trans. Pattern Anal. Mach. Intell. **31**(8), 1362–1374 (2009)
21. De Smet, P., Pires, R.L.V.: Implementation and analysis of an optimized rainfalling watershed algorithm. SPIE Int. Soc. Opt. Eng. **3974**, 759–766 (2000)
22. Digabel, H., Lantuejoul, C.: Iterative algorithms. pp. 85–89 (1978)
23. Dubey, R., Hanmandlu, M., Gupta, S.: A comparison of two methods for the segmentation of masses in the digital mammograms. Comput. Med. Imaging Graph. **34**(3), 185–191 (2010)

24. Elsalamony, H.: Detecting distorted and benign blood cells using the hough transform based on neural networks and decision trees. In: Deligiannidis, L., Arabnia, H. (eds.) Emerging Trends in Image Processing, Computer Vision and Pattern Recognition, pp. 457–473 (2014)
25. Gao, H., Xue, P., Lin, W.: A new marker-based watershed algorithm. vol. 2, pp. II81–II84 (2004)
26. Gao, L., Yang, S., Xia, J., Liang, J., Qin, Y.: A new marker-based watershed algorithm (2007)
27. Gies, V., Bernard, T.: Statistical solution to watershed over-segmentation. Int. Conf. Image Process. **3**, 1863–1866 (2004)
28. Guo, Z., Xin, Y., Liu, S., Lv, X., Li, S.: Comparisons of fat quantification methods based on mri segmentation, pp. 1817–1821 (2014)
29. Hagyard, D., Razaz, M., Atkin, P.: Analysis of watershed algorithms for greyscale images. Proc. 3rd IEEE Int. Conf. Image Process. **3**, 41–44 (1996)
30. Held, C., Wenzel, J., Webel, R., Marschall, M., Lang, R., Palmisano, R., Wittenberg, T.: Using multimodal information for the segmentation of fluorescent micrographs with application to virology and microbiology. In: Conference proceedings : ... Annual International Conference of the IEEE Engineering in Medicine and Biology Society. IEEE Engineering in Medicine and Biology Society. Conference, pp. 6487–6490 (2011)
31. Hucko, M., Srámek, M.: Streamed watershed transform on gpu for processing of large volume data, pp. 137–141 (2012)
32. Idbraim, S., Mammass, D., Aboutajdine, D., Ducrot, D.: An automatic system for urban road extraction from satellite and aerial images. WSEAS Trans. Signal Process. **4**(10), 563–572 (2008)
33. Ikedo, Y., Fukuoka, D., Hara, T., Fujita, H., Takada, E., Endo, T., Morita, T.: Development of a fully automatic scheme for detection of masses in whole breast ultrasound images. Med. Phys. **34**(11), 4378–4388 (2007)
34. Jabid, T., Mohammad, T., Ahsan, T., Abdullah-Al-Wadud, M., Chae, O.: An edge-texture based moving object detection for video content based application. pp. 112–116 (2011)
35. Jiafu, L., Yan, L., Wenfeng, Z., Jing, L.: Storm floods risk assessments by ga-bp: a case study of seven countries in Asia. Int. J. Adv. Comput. Technol. **3**(10), 323–329 (2011)
36. Jianhua, L., Shuang, W., Licheng, J.: Method to reduce over-segmentation of images using immune clonal algorithm. vol. 6786 (2007)
37. Jouini, M., Vega, S., Mokhtar, E.: Multiscale characterization of pore spaces using multifractals analysis of scanning electronic microscopy images of carbonates. Nonlinear Process. Geophys. **18**(6), 941–953 (2011)
38. JW, L., JA, D.: Optimal identification of lumped watershed models. Water Resour. Res. **5**(3), 583–590 (1969)
39. Kauffmann, C., Piche, N.: A cellular automaton for ultra-fast watershed transform on gpu (2008)
40. Kekre, H., Sarode, T., Gharge, S.: Vector quantization for tumor demarcation of mammograms. Commun. Comput. Inf. Sci. **70**, 157–163 (2010)
41. Kollorz, E., Angelopoulou, E., Beck, M., Schmidt, D., Kuwert, T.: Using power watersheds to segment benign thyroid nodules in ultrasound image data, pp. 124–128 (2011)
42. Kong, J., Cooper, L., Kurc, T., Brat, D., Saltz, J.: Towards building computerized image analysis framework for nucleus discrimination in microscopy images of diffuse glioma, pp. 6605–6608 (2011)
43. Körbes, A., Lotufo, R.: Analysis of the watershed algorithms based on the breadth-first and depth-first exploring methods. pp. 133–140 (2009)
44. Körbes, A., Vitor, G., De Alencar Lotufo, R., Ferreira, J.: Advances on watershed processing on gpu architecture. In: Lecture Notes in Computer Science. Lecture Notes in Artificial Intelligence. Lecture Notes in Bioinformatics, vol. 6671, pp. 260–271, LNCS (2011)
45. Linguraru, M., Howe, R.: Texture-based instrument segmentation in 3d ultrasound images, vol. 6144 II (2006)
46. Liu, J., Chen, K.: Novel method of mri medical image segmentation combining watershed algorithm and wkfcm algorithm. Appl. Mech. Mater. **121–126**, 4518–4522 (2012)

47. Lotufo, R., Falcao, A.: The ordered queue and the optimality of the watershed approaches. Math. Morphol. Appl. Image Signal Process. **18**, 341–350 (2000)
48. Mahmoudi, R., Akil, M.: Real time topological image smoothing on shared memory parallel machines, vol. 7871 (2011)
49. Malpica, N., De Solórzano, C., Vaquero, J., Santos, A., Vallcorba, I., García-Sagredo, J., Del Pozo, F.: Applying watershed algorithms to the segmentation of clustered nuclei. Cytometry **28**(4), 289–297 (1997)
50. Mei, T., Li, D., Qin, Q.: Application of knowledge based watershed transform approach to road detection, vol. 6045 II (2005)
51. Meijster, A., Roerdink, J.B.T.M.: A disjoint set algorithm for the watershed transform. In: Proceedings EUSIPCO '98, IX European Signal Processing Conference, pp. 1665–1668 (1998)
52. Mendonca, A.S., Rezende, R.A.: Application of geographical information systems and stochastic hydrology to the siting and design of water reservoirs. In: International Geoscience and Remote Sensing Symposium. IGARSS'99, vol. 2, pp. 1220–1222 (1999)
53. Meyer, F.: Topographic distance and watershed lines. Signal Process. **38**(1), 113–125 (1994)
54. Moga, A., Cramariuc, B., Gabbouj, M.: Parallel watershed transformation algorithms for image segmentation. Parallel Comput. **24**(14), 1981–2001 (1998)
55. Moga, A., Gabbouj, M.: Parallel marker-based image segmentation with watershed transformation. J. Parallel Distrib. Comput. **51**(1), 27–45 (1998)
56. Mohan, E., Sugumaran, R., Venkatachalam, K.: Automatic brain and tumor segmentation in mri using fuzzy classification with integrated bayesian model. Int. J. Appl. Eng. Res. **9**(24), 25859–25870 (2014)
57. Mortensen, E.N., Barrett, W.A.: Toboggan-based intelligent scissors with a four-parameter edge model. In: CVPR, IEEE Computer Society, pp. 2452–2458 (1999)
58. Moumoun, L., El Far, M., Chahhou, M., Gadi, T., Benslimane, R.: Solving the 3d watershed over-segmentation problem using the generic adjacency graph (2010)
59. Muzylev, E., Uspensky, A.: Modelling the Hydrological Cycle of River Basins Using High Resolution Satellite Information, pp. 241–248. IAHS-AISH Publication, Wallingford (2001)
60. Najman, L., Couprie, M.: Watershed algorithms and contrast preservation. In: Lecture Notes in Computer Science. Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 2886, pp. 62–71 (2003)
61. Nithya, A., Kayalvizhi, R.: Extended fuzzy switching median filter and morphological algorithm for medical image segmentation. ARPN J. Eng. Appl. Sci. **10**(1), 80–90 (2015)
62. Noguet, D.: Massively parallel implementation of the watershed based on cellular automata, pp. 42–52 (1997)
63. Osma-Ruiz, V., Godino-Llorente, J., Sáenz-Lechón, N., Gómez-Vilda, P.: An improved watershed algorithm based on efficient computation of shortest paths. Pattern Recognit. **40**(3), 1078–1090 (2007)
64. Patino, L.: Fuzzy relations applied to minimize over segmentation in watershed algorithms. Pattern Recognit. Lett. **26**(6), 819–828 (2005)
65. Peng, B., Xu, A., Li, H., Han, Y.: Road extraction based on object-oriented from high-resolution remote sensing images (2011)
66. Perry, E., Norton, S., Kamman, N., Lorey, P., Driscoll, C.: Deconstruction of historic mercury accumulation in lake sediments, Northeastern United States. Ecotoxicology **14**(1–2), 85–99 (2005)
67. Plaza, A., Plaza, J., Valencia, D., Martinez, P.: Parallel segmentation of multi-channel images using multi-dimensional mathematical morphology (2008)
68. Procházka, A., Vysata, O., Jerhotova, E.: Wavelet use for reduction of watershed transform over-segmentation in biomedical images processing (2010)
69. Rambabu, C., Rathore, T., Chakrabarti, I.: A new watershed algorithm based on hillclimbing technique for image segmentation. In: TENCON 2003. Conference on Convergent Technologies for Asia-Pacific Region, vol. 4, pp. 1404–1408 (2003)
70. Rambabu, C., Chakrabarti, I.: An efficient hillclimbing-based watershed algorithm and its prototype hardware architecture. J. Signal Process. Syst. **52**(3), 281–295 (2008)

71. Randhir, T., Lee, J., Engel, B.: Multiple criteria dynamic spatial optimization to manage water quality on a watershed scale. Trans. Am. Soc. Agric. Eng. **43**(2), 291–299 (2000)
72. Roerdink, J., Meijster, A.: The watershed transform: definitions, algorithms and parallelization strategies. Fundamenta Informaticae **41**(1–2), 187–228 (2000)
73. Rong, J., Pan, Y.-L.: Accuracy improvement of graph-cut image segmentation by using watershed. Adv. Mater. Res. **341–342**, 546–549 (2012)
74. Shen, W.-C., Chang, R.-F.: A nearest neighbor graph based watershed algorithm, pp. 6300–6303 (2005)
75. Sheshadri, H., Kandaswamy, A.: Application of watershed algorithms to mammogram image analysis. IETE Tech. Rev. **23**(3), 173–178 (2006)
76. Shrimali, V., Anand, R., Kumar, V.: Current trends in segmentation of medical ultrasound b-mode images: a review. IETE Tech. Rev. **26**(1), 8–17 (2009)
77. Sinha, K., Sinha, G.: Efficient segmentation methods for tumor detection in mri images (2014)
78. Smistad, E., Falch, T., Bozorgi, M., Elster, A., Lindseth, F.: Medical image segmentation on gpus - a comprehensive review. Med. Image Anal. **20**(1), 1–18 (2015)
79. Sridhar, B., Reddy, K., Prasad, A.: An artificial neural network and mathematical morphology based computer aided detection system for cancer detection in mammograms. Int. J. Appl. Eng. Res. **9**(23), 21079–21097 (2014)
80. Su, H., Yang, Z.-L., Dickinson, R., Wilson, C., Niu, G.-Y.: Multisensor snow data assimilation at the continental scale: the value of gravity recovery and climate experiment terrestrial water storage information. J. Geophys. Res. Atmospheres **115**(10) (2010)
81. Sun, H., Yang, J., Ren, M.: A fast watershed algorithm based on chain code and its application in image segmentation. Pattern Recognit. Lett. **26**(9), 1266–1274 (2005)
82. Świercz, M., Iwanowski, M.: Fast, parallel watershed algorithm based on path tracing. In: Lecture Notes in Computer Science. Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 6375, pp. 317–324, LNCS(PART 2) (2010)
83. Swiercz, M., Iwanowski, M.: Waterball-iterative watershed algorithm with reduced oversegmentation. Adv. Intell. Soft Comput. **95**(4), 385–394 (2011)
84. Tolosa, S., Blacher, S., Denis, A., Marajofsky, A., Pirard, J.-P., Gommes, C.: Two methods of random seed generation to avoid over-segmentation with stochastic watershed: application to nuclear fuel micrographs. J. Microsc. **236**(1), 79–86 (2009)
85. Tonti, S., Di Cataldo, S., Bottino, A., Ficarra, E.: An automated approach to the segmentation of hep-2 cells for the indirect immunofluorescence ana test. Comput. Med. Imaging Graph. **40**, 62–69 (2015)
86. Tung, C.-P.: Climate change impacts on water resources of the tsengwen creek watershed in Taiwan. J. Am. Water Resour. Assoc. **37**(1), 167–176 (2001)
87. Uchida, S.: Image processing and recognition for biological images. Dev. Growth Differ. **55**(4), 523–549 (2013)
88. Van Neerbos, J., Najman, L., Wilkinson, M.: Towards a parallel topological watershed: First results. In: Lecture Notes in Computer Science Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 6671, pp. 248–259, LNCS (2011)
89. Vibha, L., Harshavardhan, G., Pranaw, K., Shenoy, P., Venugopal, K., Patnaik, L.: Lesion detection using segmentation and classification of mammograms, pp. 311–316(2007)
90. Vincent, L., Soille, P.: Watersheds in digital spaces: an efficient algorithm based on immersion simulations. IEEE Trans. Pattern Anal. Mach. Intell. **13**(6), 583–598 (1991)
91. Wagner, B., Dinges, A., Müller, P., Haase, G.: Parallel volume image segmentation with watershed transformation. In: Lecture Notes in Computer Science. Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics, vol. 5575, pp. 420–429, LNCS (2009)
92. Wang, W., Shi, H., Wang, A.: Analysis on the future runoff changes in shiyang river basin based on genetic algorithm models (2012)
93. Wu, S., Hu, Y.: Parallelization research watershed algorithm **2012**, 1524–1527 (2012)
94. Xu, G., Zhang, D., Liu, X.: Road extraction in high resolution images from google earth (2009)
95. Yang, F., Li, J., Xu, S.-H., Pan, G.-F.: The research of video segmentation algorithm based on image fusion in the wavelet domain, vol. 7659 (2010)

96. Yu, P.-Y., Zhang, G.-P., Yan, J.-W., Liu, M.-S.: The application of the watershed algorithm based on line-encoded in lung ct image segmentation (2011)
97. Zhang, X., Chen, L., Pan, L., Xiong, L.: Study on the image segmentation based on ica and watershed algorithm, pp. 505–508 (2012)
98. Zhang, X., Cheng, Y., Qian, Y., Zhuang, X.: Automatic video object segmentation based on spatio-temporal information, pp. 5314–5317 (2011)
99. Zhu, H., Zhang, B., Song, A., Zhang, W.: An improved method to reduce over-segmentation of watershed transformation and its application in the contour extraction of brain image, pp. 407–412 (2009)