Analysis of the Variants of Watershed Algorithm as a Segmentation Technique in Image Processing

Namrata Puri and Sumit Kaushik

[1]*Research scholar,* [2]*Assistant Professor*
Ambala College of Engineering & Applied Research, Devsthatli, Ambala Cantt, India
E-mail:namratapuri12@yahoo.in,sumitkaushik24@gmail.com

Abstract

The watershed transform is a popular image segmentation algorithm for grey scale images. It is the method of choice for image segmentation in the field of mathematical morphology. Watershed segmentation is based on sets of neighboring pixels. We present a critical review of several definitions of the watershed transform and the associated sequential algorithms, immersion models and therefore parallel implementation of these immersion and sequential models. In this paper, procedure regarding performance analysis of these three variants is drawn and further the OpenCV tool is used to calculate the final results.

Keywords: Watershed transform, Mathematical morphology, Immersion, OpenCV

Introduction

Image processing is defined as a technique in which the data from an image are digitized and various mathematical operations are applied to the data, generally with a digital computer, in order to create an enhanced image that is more useful or pleasing to a human observer, or to perform some of the interpretation and recognition tasks usually performed by humans. Also known as picture processing[7].

In computer vision, segmentation refers to the process of partitioning a digital image into multiple segments (sets of pixels, also known as superpixels). The goal of segmentation is to simplify and/or change the representation of an image into something that is more meaningful and easier to analyze. Image segmentation is typically used to locate objects and boundaries (lines, curves, etc.) in images. More precisely, image segmentation is the process of assigning a label to every pixel in an image such that pixels with the same label share certain visual characteristics. The result of image segmentation is a set of segments that collectively cover the entire image, or a set of contours extracted from the image[6].

The watershed transformation considers the gradient magnitude of an image as a topographic surface. Pixels having the highest gradient magnitude intensities (GMIs) correspond to watershed lines, which represent the region boundaries. Water placed on any pixel enclosed by a common watershed line flows downhill to a common local intensity minimum (LIM). Pixels draining to a common minimum form a catch basin, which represents a segment.

Generally spoken, image segmentation is the process of isolating objects in the image from the background, i.e., partitioning the image into disjoint regions, such that each region is homogeneous with respect to some property, such as grey value or texture [1].

The watershed transform can be classified as a region-based segmentation approach. The intuitive idea underlying this method comes from geography: it is that of a landscape or topographic relief which is flooded by water, watersheds being the divide lines of the domains of attraction of rain falling over the region [2].

An alternative approach is to imagine the landscape being immersed in a lake, with holes pierced in local minima. Basins (also called `*catchment basins*') will fill up with water starting at these local minima, and, at points where water coming from different basins would meet, dams are built. When the water level has reached the highest peak in the landscape, the process is stopped. As a result, the landscape is partitioned into regions or basins separated by dams, called *watershed lines* or simply *watersheds*[2].

When simulating this process for image segmentation, two approaches may be used: either one first finds basins, then watersheds by taking a set complement; or one computes a complete partition of the image into basins, and subsequently finds the watersheds by boundary detection.

To be more explicit, we will use the expression `watershed transform' to denote a labelling of the image, such that all points of a given catchment basin have the same unique label, and a special label, distinct from all the labels of the catchment basins, is assigned to all points of the watersheds.

We note in passing that in practice one often does not apply the watershed transform to the original image, but to its (morphological) gradient [3]. This produces watersheds at the points of grey value discontinuity, as is commonly desired in image segmentation.

The most frequently used definition of the watershedding operation follows a geographical analogy (Fig. 1). If a grayscale image is viewed as if high intensity colors were high ground then the image becomes a 3D landscape.

The catchment basin of this minimum is the area, where water falling on the landscape would flow down to the minimum. The watershed of the image is the set of lines (dams) that separate the catchment basins on the image. The “height” in topographic surface may be any measurable property of image pixel: lightness, gradient of lightness, saturation or other. That makes watershed algorithm useful for color image processing.

The watershed transformation performs very accurate segmentation, which is beneficial in case when objects overlap and their borders are hardly detectable[3].

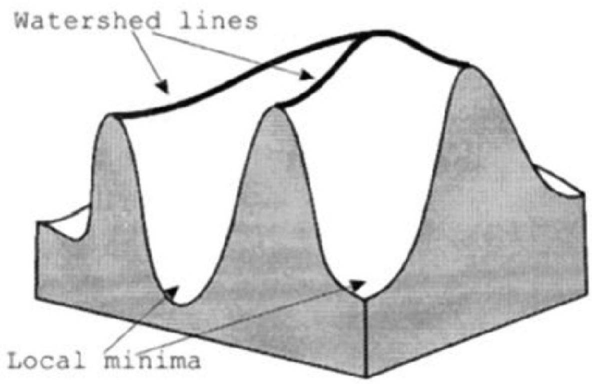


Fig 1. Local minima of a surface

Literature Survey

Generally spoken, existing watershed algorithms either simulate the flooding process, or directly detect the watershed points. In some implementations, one computes basins which touch, i.e., no watershed pixels are generated at all[1].

Watershed algorithms by immersion

Vincent-Soille algorithm

An algorithmic definition of the watershed transform by simulated immersion was given by Vincent and Soille.

The immersion approach [6] is also referred to as the flooding analogy. In the immersion simulation, we first pierce a hole in every local minimum of the topographic surface formed by the gradient magnitude image. Then, we slowly immerse the topographic surface in water. Starting from the minima of lowest altitude, the water will progressively fill up all the different catchment basins. At some point, the rising water in any one specific basin will start to merge with water coming from its neighboring basins. Suppose that this merging can be prevented by constructing dams at the merging sites all the way up to the highest surface altitude (or until the immersion procedure ends). At the end of this immersion simulation, each basin will be completely surrounded by dams and the location of dams corresponds to watershed line.

Order-Invariant Immersion Algorithm

This section presents the details of our order-invariant immersion algorithm for image segmentation. Similar to the one proposed by Vincent and Soille in [6], this algorithm first requires a sorting of the pixels in the increasing order of their gradient magnitude values before running the level-by-level flooding step. It is this sorting step that has made the level-by-level flooding step efficient enough so that the Vincent and Soille algorithm can surpass its predecessors in computational efficiency.

Let $G: D \rightarrow R^+$ be a gradient magnitude image, where D is the indexing domain of the image (e.g., $D = Z^2$) and max(G) be the minimum value and the maximum value of G, respectively. By sorting the pixels of G in the increasing order of their gradient magnitude values, we can easily decompose D into a finite number of disjoint level sets, each denoted by

$$D_h = \{p \in D \mid G(p) = h\} \quad (1)$$

That is, we have $D = \cup_h D_h$,$\min(G) \le h \le \max(G)$, and $D_k \cap D_l$ if $k \neq l$. Different sorting techniques can be used here. For better efficiency, our algorithm uses counting sort if the data type of the gradient magnitude is fixed point, while it uses quick sort if the data type is floating point.

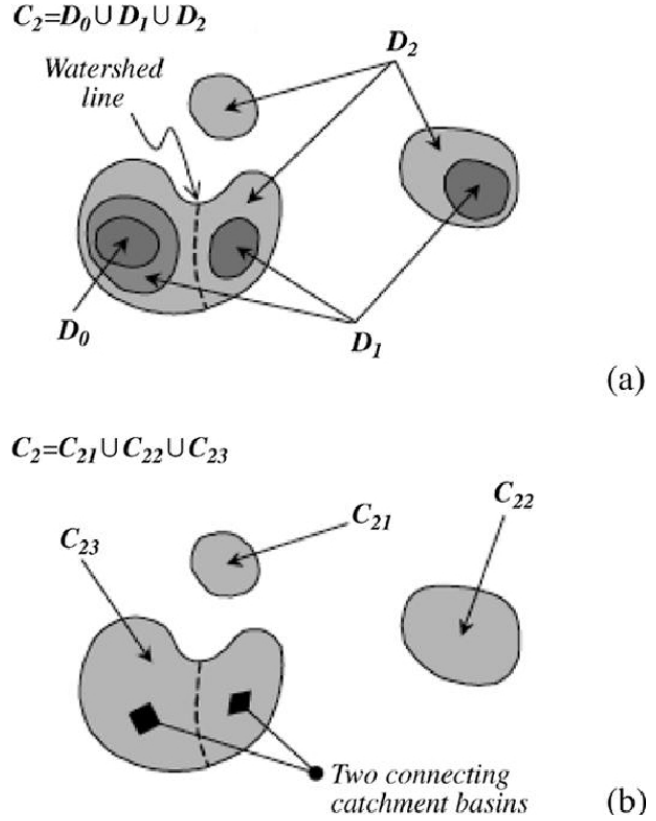


Fig 2. Disjoint level sets

If it is assumed that we have reached to level h-1 after the pixel sorting step during immersion process. C_{h-1} can also be viewed as a union of connected components, i.e. where each connected component contains one or more than one catchment basin. The catchment basin here can be referred to as a pre-h catchment basin because it is formed right before the water level rises up to level-h. For example, in Fig. 2(b) contains three connected components, denoted by C_{21},C_{22} ,and C_{23} . Note that C_{23} contains two pre-2catchment basins separated by a watershed line, while both others contains only one pre-2 catchment basin.

Next, by letting the water level goes up to level h, we have a new level set D_h , which can also be viewed as a union of connected components, i.e., . An example is given in Fig. 3

Here, the three different components are classified.

***Type-2 Component*:** More than one pre-h catchment basin is connected to this type of connected component. For example, in Fig. 3 D31 is a type-2 component because it connects to three pre-2 catchment basins. Notice that actually contains two pre-2 catchment basins.

Type-1 Component: Exactly one pre-h catchment basin is connected to this type of connected component. For example, in Fig. 3 D32 is a type-1 component because it has only one pre-2 catchment basin, .

***Type-0 Component*:** No pre-h catchment basin is connected to this type of connected component. For example, in Fig. 3 D33 is a type-0 component.

Notice that when the flooding has been completed up to level h-1, every pixel having altitude less than or equal to h-1 will have already been assigned a unique catchment basin label.

To implement the flooding step, we divide the pixels in into the following three classes and label the pixels in each class one by one. It is worth mentioning that the classification of the connected components in is for understanding the algorithm, while the following classification of the individual pixels in is for implementing the algorithm.

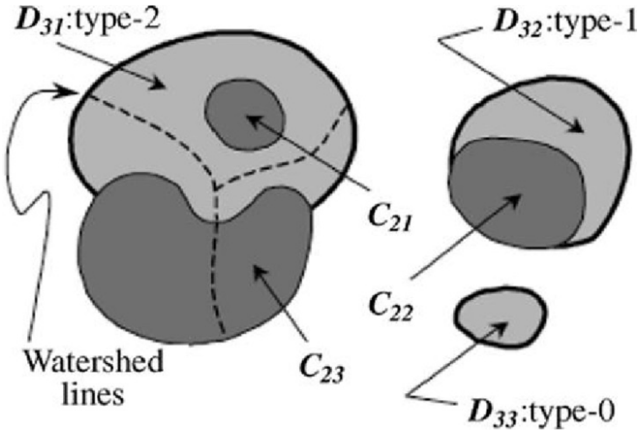


Fig 4. relation between sets and components

***Class-I Pixel*:** A pixel p in Dh is called a class-I pixel if its altitude is strictly greater than the altitude of its lowest neighbor.

Class-II Pixel: This class of pixels can be viewed as interior pixels of nonlocal-minimum plateaus. Fig. 4 shows some examples of class-II pixels.

***Class-III Pixel*:** Pixels in type-0 components of are class-III pixels. All the pixels in one type-0 component will be assigned a new and unique label.

Algorithm 1. Immersion Approach

Step 1. Sorting Step: Sort all the pixels in the gradient magnitude image G to obtain level sets D_h in increasing h.

Step 2. Flooding Step:

For each level set Dh , in the increasing order of h. Step 2.1.Simulate flooding for all the class-I pixels in Dh by labeling each class-I pixel with the label of its lowest neighbor . All these class-I pixels are pushed into a FIFO queue for region growing in

Step 2.2.
Step 2.2.Simulate flooding for all class-II pixels in Dh by region growing from class-I pixels using the FIFO queue initialized in Step 2.1.

Step 2.3. Simulate flooding for class-III pixels in Dh by assigning a new and unique label to each of the type-0 components in Dh.

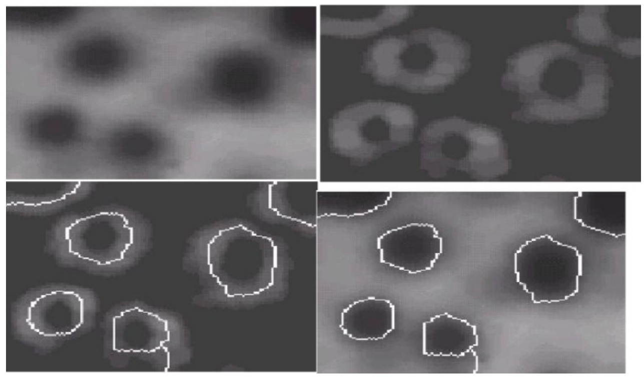


Fig 4. a) image of blobs b) image of gradient c) watershed lines d)watershed lines superimposed on original image.

Sequential Rainfalling Algorithm
The watershed transformation was widely and successfully applied in different domains (e.g., in biomedicine, industry, and, generally in computer vision applications) as a powerful segmentation tool. The idea behind it is to split the morphological gradient of an original image, seen as a topographic surface, into geodesic influence zones . Unlike the other methods the herein introduced algorithm has a higher degree of locality such that it can generate faster parallel implementations.

Description of the Method
The watershed algorithm based on rain falling simulation performs segmentation by labeling connected areas within the gradient of an image. Regarding the morphological gradient of the original image as a topographic surface, the rule of assigning labels can be derived from physics: a particle in free fall on a topographic surface will move due to gravity downward to the deepest neighboring location. On flat areas, the rule is overloaded, such that the motion of the particle is directed toward the nearest brim of a downward slope, or it stops if the particle has reached a regional minimum.

The task performed by the present algorithm is to trace a path for each non-minimum point on the surface (origin) to a minimum (destination), and to mark all pixels along the path with the label of the minimum. This path is a steepest slope line in a lower-complete image. The latter is the transformed gradient image such that any non-minimum pixel has a lower neighboring one. The result is a partition of the image which has the following properties: regions are connected, they do not overlap, and the partition is complete.

Advantage of the Proposed Method over the Watershed by Immersion

An important advantage of this watershed algorithm is its suitability for parallel implementation. While immersion is a global method (water arising from many sources progressively floods all the surface and interactions between waters coming from different sources are taken into consideration), raining can be denoted as a local method because each droplet follows on its own way regardless of neighboring droplets.

Parallel Watershed algorithm

The computation of the watershed transform of a gray scale image is a relatively time consuming task and therefore usually one of the slow step in this chain. A common solution for such computationally expensive algorithms is to search for implicit parallelism in the algorithm and use this to implement the algorithm on a parallel computer.

As the watershed is sequential it is implemented in parallel by splitting the computation in three stages:

1. In the first stage of algorithm, input image is transformed into a directed components graph.

In the second stage of algorithm, the watershed of this graph is computed by breadth first coloring algorithm.

In the final stage, the flooded graph is transformed back into image domain.

If pixels € watershed nodes, then pixels are colored white and all pixels€ non- watershed nodes are colored black.

Watersheds are "thick" and thinning is done by skeletonization.

Problem Formulation

In order to analyze the different variants of watershed transform, it is necessary to implement the different variants such as immersion and sequential techniques and then compare the results.

Problem in Mathematical Form

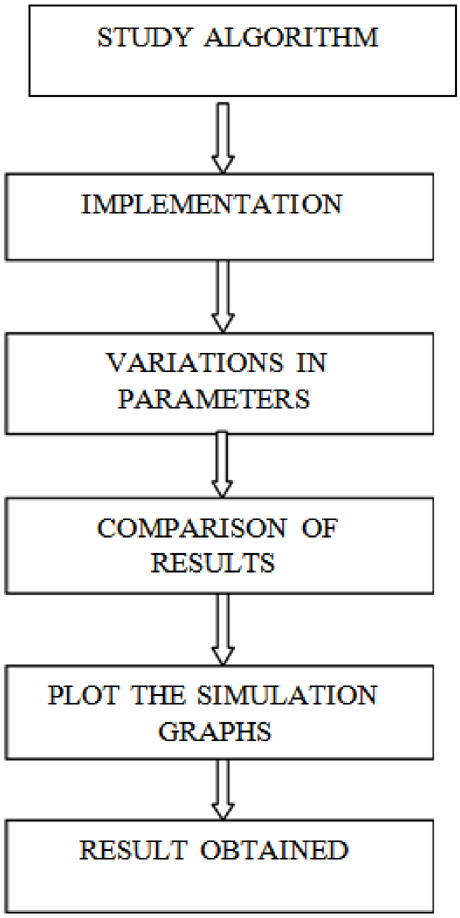
Let us suppose we have set of n watershed algorithms denoted

by array W={w1,w2,……………wn}

Using size of the image when taken as input as a parameter for comparison, we will find out w_i €W such that 1<i<n in order to find out the effect of increasing the size on w_i is minimum.

Proposed Model

We will do parallel implementation of the given algorithms and input the image of different sizes and then compare the outputs of different variants in order to plot simulation graphs of various outputs and hence it will be identified which algorithm gives the minimized effect of the image as a output.



The tool that will be used for analysis is open CV tool. OpenCV (Open Source Computer Vision Library) is a library of programming functions mainly aimed at real time computer vision.

Conclusion

In this paper, we have presented a definition of the watershed segmentation which is consistent with the behavior of most implementations of the watershed algorithm. All the algorithms described extend to the three dimensional case in a straightforward manner. The watershed algorithm by immersion is hard to parallelize because of its inherently sequential nature. A parallel implementation of this algorithm can be based upon a transformation to a component graph. After analyzing the various algorithms using graphs, we will find that varying the size of image, length of watershed line, intensity value of gradient image as parameters which variant of watershed has given the best performance.

Future Scope

As we have used size of image as a parameter, in future work length of watershed line, intensity value of gradient image, texture can be taken as parameter for performance analysis of variants of watershed transform. It is thus expected to contribute to new insights into the use of watersheds in the field of image analysis. In particular, more experiments are currently being carried on to evaluate the interest on watershed of graphs with respect to picture segmentation.

References

[1] Meyer, F., and Beucher, S. Morphological segmentation. J. Visual Commun. and Image Repres. 1,(1990).

[2] Serra, J. Image Analysis and Mathematical Morphology. Academic Press,New York, 1982.

[3] S. BEUCHER, C. LANTUEJOUL, Use of watersheds in contour detection. International Workshop on image processing,real-time edge and motion detection/estimation, Rennes, France, Sept. 1979.

[4] Oversegmentation avoidance in watershed-based algorithms for color images Wojciech Bieniecki.

[5] Haralick, R. M., and Shapiro, L. G. Survey : image segmentation techniques. Comp. Vis. Graph

[6] Im. Proc. 29 (1985), 100-132.

[7] http://en.wikipedia.org/wiki/Segmentation_%28image_processing%29

[8] http://www.answers.com/topic/image-processing

[9] http://www.cs.toronto.edu/~jepson/csc2503/segmentation.pdf

[10] L. Vincent and P. Soille, "Watersheds in digital spaces: an efficient algorithm based on immersion simulations, *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 13, no. 6, pp. 583–598, Jun. 1991.

[11] A.Meijister and J.B.T.M.Roerdink.proposal for the implememtation of a parallel watershed analysis. In :proceedings of computer analysis of images & Pattern(CAIP"95),Springer Verlag,1995.

[12] http://en.wikipedia.org/wiki/OpenCV

[13] Digital image Processing 2nd ed.. R. C. Gonzalez & R. E. Woods,2002..