Scientific Research

Fingerprint Database Optimization Using Watershed Transformation Algorithm

Owoeye Kolade, Ajayi Adedoyin Olayinka, Ukorigho Ovie

Department of Mathematical Sciences, Ekiti State University, Ado Ekiti, Nigeria
Email: kolade_owoeye@yahoo.com, dedoyyin@gmail.com, ovi.uko@gmail.com

Abstract

Fingerprints are a unique feature for identification and verification of humans. The need to optimise several databases for storing the images of fingerprints is a major concerning issue. Several segmentation algorithms have been used in the time past but there are still several challenges facing some current segmentation algorithms like computational efficiency. Another challenge is that segmentation procedure can be impractically slow, or requires extremely large amounts of memory. This paper addresses the challenges by employing watershed flooding algorithm on the fingerprint images so as to optimize the sizes of the databases. A pre-processing plug-in that implements this segmentation process is developed using Java. We showed its effectiveness by testing it on fingerprint image dataset and the entropy showed that the segmented images sizes were reduced.

Keywords

Watershed, Transformation, Fingerprint, Segmentation, Entrophy

1. Introduction

Image Segmentation is a fundamental step in analysing and understanding images. It is a process of partitioning the image into multiple segments [1] [2]. It is the first important step in many image processing applications like image analysis, image description and recognition, image visualization and object based image compression. Image segmentation means assigning a label to each pixel in the image such that pixels with same labels share common visual characteristics.

For more than a century, fingerprints were considered to be the identifying mark for the human beings. Fingerprint is a protected human organ and an effective biometric approach to human or personal identification. It acts like living passwords for humans as its texture is stable throughout the human life. Fingerprints are an im-

pression left by the friction ridges of human finger.

For several reasons, we need to store these fingerprints in a database and among them one of the main reasons is they are used for analysis of forensic evidence worldwide. For storing several fingerprint impressions a huge database is needed, where the size of the database is also a matter of consideration. A huge database needs a huge amount of memory space. If we can reduce the size of the data then we can store more number of data in the same memory space.

Watershed transformation can be applied to human fingerprints segmentation by taking the idea from friction ridges of human finger and also with an effective storage capacity for the segmented images. Watershed algorithm depends on ridges to perform a proper segmentation, a property that is often fulfilled in contour detection where the boundaries of the objects are expressed as ridges.

In grey scale mathematical morphology the watershed transform, which was originally proposed by [3] and later improved by [4], was the method of choice for image segmentation [5]. When simulating the watershed transform 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.

The basic idea behind watershed algorithm by immersion comes from Geography. It requires that one think of an image as a surface; that bright areas are "high" surfaces and dark areas are "low" surfaces. With surfaces, it is natural to think in terms of catchment basins and watershed lines. Basins, also called catchment basins (**Figure 1**), 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 watersheds.

Finally, this paper is organized in sections. Section 2 explained the related work. Watershed algorithmic definitions were discussed in Section 3. While the implementation process in Section 4. Lastly, summary and conclusion is in Section 5.

2. Related Work

Although the research by [5] is the most related to our research, comprehensive reviews of early segmentation techniques can be found in [7] and [8]. Some classes of segmentation methods are considered below: Gray level thresholding, and region growing/merging techniques.

Gray level thresholding is a generalization of binary thresholding [9]. Binary thresholding works by determining the gray level value that separates pixels in the foreground from pixels in the background, and generating a "threshold image" where pixels are assigned one of two possible values corresponding to "foreground" and "background" depending on whether their gray level is above or below the selected threshold.

The work of Beveridge *et al*. [10] offers a good example of a procedure that integrates both gray level thresholding and region merging. In their paper, an input image (which can be either grayscale or colour) is divided into sectors of fixed size and fixed location. An intensity histogram is calculated for each sector (and on colour images, for each colour channel), and used to produce a local segmentation. For every sector, information from its neighbors is used to detect clusters for which there may not be enough local support due to the artificially induced partition of the image. After the local segmentations are complete, the sector boundaries are removed by merging together similar regions in neighboring sectors.

(a)   (b)
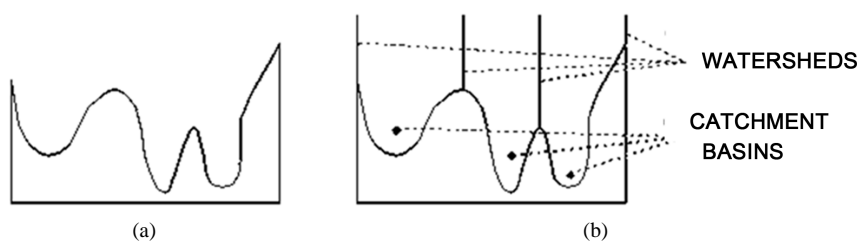
Figure 1. One dimensional example of watershed segmentation. (a) Gray level profile of Image data; (b) Watershed segmentation—local minima of gray level (altitude) yield catchment basins; local maxima define the watershed lines (Source: [6]).

The above measure is computed for both the complete regions, and a band that is within a fixed, small distance on both sides of the boundary. Two regions are merged if the merge score is below a specified threshold for both the global and local measure.

The last step in the segmentation is region merging; this step uses a merge score composed of a pairwise comparison of several region features. Since the algorithm can only merge regions, the thresholds used during the local, threshold based segmentation stage are selected so that they'll yield a significantly over-segmented image; the merging step is then relied upon to turn the un-segmented image into a reasonable segmentation. Results presented in [10] show that this algorithm produces good segmentations in parts of the image that are reasonably homogeneous, and over-segmented regions when there is texture, significant intensity gradients, or objects with non-uniform coloring. The algorithm is not without problems, as there are several thresholds that must be chosen carefully depending on the image, and the region boundaries themselves have slight artifacts introduced by the sector-based initial segmentation. Even so, the algorithm illustrates what can be achieved with thresholding merging schemes.

The works in the literature showed that many current algorithms are able to produce reasonable results on images of moderate complexity; several of these algorithms are efficient enough that they can be used as a pre-processing stage for higher level vision tasks such as recognition and tracking.

However, there are still several challenges facing some current segmentation algorithms. Computational efficiency is still a concern issue when the processing of large affinity matrices is a part of the segmentation process, ultimately, a segmentation procedure can become impractically slow, or require extremely large amounts of memory. These have limited the size of the images that can be processed using many recent algorithms. However, we expect that the constant increase in computational power and storage capacity of modern computers should progressively reduce these limitations. The definition of a good similarity measure for general images remains an open issue. There is a general consensus that a robust image segmentation algorithm should combine multiple image cues and estimate similarity based on this combination, but so far there are few algorithms that use more than a single cue as a similarity measure, and only recently has a significant effort been dedicated to designing similarity measures based on the statistics of natural images, and human-generated segmentations.

The design of a good similarity measure is tied to the robustness of the segmentation algorithm in dealing with surface markings, lighting artifacts, and image texture. Evaluating the output of segmentation algorithms is still problematic. Therefore, watershed algorithm is adopted in this paper.

3. Watershed Algorithmic Definition

The diagrammatic descriptions of watershed lines and catchment basins have been presented in **Figure 1**. An algorithmic definition of the watershed transform by simulated immersion was given by [2]. Consider a digital grey value image, $f : D \to \mathbb{N}$ with minimum value $h_{\min}$ and maximum value $h_{\max}$ of f. We plunge this surface into a lake with a constant vertical speed, with water entering through the holes and flood the surface. Define a recursion with the grey level h increasing from $h_{\min}$ to $h_{\max}$, in which the basins associated with the minima of f are successively expanded. Let X_h denote the union of the set of basins computed at level h. A connected component of the threshold set T_{h+1} at level $h+1$ can be either a new minimum, or an extension of a basin in X_h: in the latter case one computes the geodesic influence zone of X_h within T_{h+1}, resulting in an update X_{h+1}.

Let MIN_h denote the union of all regional minima at altitude h.

Definition (Watershed Transform): Let $f \in C(D)$ have a minima $\{m_k\}_{k \in I}$ for some index set I. The catchment basin $\mathrm{CB}(m_i)$ of a minimum m_i is defines as the set of points $x \in D$ which are topographically closer to m_i than to any other regional minimum m_j.

$$\mathrm{CB}(m_i) = \left\{ x \in D \middle| \forall j \in I / \{i\} : f(m_i) + T_{f(x,m_i)} < f(m_j) + T_{f(x,m_j)} \right\}$$

The watershed of f is the set of points which do not belong to any catchment basins:

$$W\mathrm{shed}(f) = D \cap \left(\bigcup_{i \in I} \mathrm{CB}(m_i) \right)^c$$

Let W be some label, $W \notin I$. The watershed transform of f is a mapping $\lambda : D \to I \cup \{W\}$, such that

$\lambda(p) = i$, if $p \in CB(m_i)$, and $\lambda(p) = W$ if $p \in W\text{shed}(f)$.

So the watershed transform of f assigns labels to the points of D, such that i) different catchment basins are uniquely labeled, and ii) a special label W is assigned to all points of the watershed of f.

We assume, from [11], a geodesic distance $d_A(a,b)$ between a and b within A for $A \in \varepsilon$, with $\varepsilon = \mathbb{R}^d$ where $d_A(a,b)$ is the minimum path length among all paths within A from a to b (in the continuous case, read "infimum" instead of "minimum"). If B is a subset of A, define $d_A(a,B) = \text{MIN}_{b \in B}(d_A(a,b))$.

Let $B \in A$ be partitioned in k connected components B_i, $i = 1, \cdots, k$. The *geodesic influence zone* of the set B_i within A is defined as:

$$iz_A(B_i) = \left\{ p \in A \middle| \forall j \in \{1, \cdots, k\} \setminus \{i\} : d_A(p, B_i) < d_A(p, B_j) \right\}$$

Let $B \in A$, The set $\text{IZ}_A(B)$ is the union of the geodesic influence zones of the connected components of B [12], *i.e.*,

$$\text{IZ}_A(B) = \bigcup_{i=1}^{k} iz_A(B_i)$$

The definition of $\text{IZ}_A(B)$ is used in the construction of a watershed algorithm by immersion.

Definition (watershed by immersion):

Define the following recursion : $\begin{cases} X_{h+1} = \{p \in D | f(p) = h_{\min}\} = T_{h_{\min}} \\ X_{h+1} = \text{MIN}_h \cup \text{IZ}_{T_{h+1}}(X_h) \qquad h \in [h_{\min}, h_{\max}) \end{cases}$

The watershed $W\text{shed}(f)$ of f is the complement of $X_{h_{\max}}$ in D:

$$W\text{shed}(f) = D / X_{h_{\max}} .$$

4. The Implementation Process

A pre-processing plug-in that implements this segmentation process is developed using Java PL. This plug-in would be compatible with an application called Image J [12], a general purpose image-processing and image-analysis package. Image J is of choice because it has a public domain licence, it runs on several operating system platforms. This application would apply the watershed flooding algorithm which can be interrupted to a user-specified level. Each particle should have a local maximum (or local minimum when the objects are dark) to define a catchment basin.

The Watershed algorithm by immersion is presented in **Table 1**.

The SKIZ (skeleton by influence zones) function is the complement of the set $\text{IZ}_A(B)$ (defined in Section 3) within A, and is defined as:

$$\text{SKIZ}_A(B) = A \setminus \text{IZ}_A(B)$$

So the SKIZ consists of all points which are equidistant (in the sense of the geodesic distance) to at least two nearest connected components. For a binary image f with domain A, the SKIZ can be defined by identifying B with the set of foreground pixels [11].

This approach is being implemented on different fingers, where the segmented results are very clear. **Figures 2-6** presents the application of the approach on the little finger, the ring finger, the middle finger, grooming finger and thumb finger respectively. In the figures presented, (a) is the original image, (b) is the grey scale image of the original one, (c) shows the overlaid basins of the grey scale image, (d) shows the catchment basins, (e) shows the composite image and finally one gets the segmented image (f) after applying watershed algorithm.

From **Table 2**, the entropy for original images with their segmented results has been measured.

Entropy is defined as:

$$H(x) = -\sum_x p(x) \log_2 [p(x)]$$

where $p(x)$ is the probability that X is in the state x and $p\log_2 p$ is defined as 0 if $p = 0$. The joint en-

Table 1. Watershed algorithm by immersion.

Vincent and Soille watershed algorithm by immersion [2]

1: **procedure** Watershed by Immersion

2: INPUT: digital grey scale image G = (D, E, *im*)

3: OUTPUT: labeled watershed image *lab* on *D*.

4: #define INIT −1 (*initial value of lab image*)

5: #define MASK −2 (*initial value at each level*)

6: #define WSHED 0 (*label of the watershed pixels*)

7: #define FICTITIOUS (−1, −1) (*fictitious pixel $\notin D$ *)

8: curlab ← 0 (*curlab* is the current label*)

9: *fifo_init* (*queue*)

10: **for all $p \in D$ do**

11: *lab* [p]← INIT; *dist* [p]← 0 (*dist* is a work image of distances*)

12: **end for**

13: SORT pixels in increasing order of grey values (minimum *hmin*, maximum *hmax*)

14: (*Start the Flooding*)

15: **for** h = *hmin* **to** *hmax* **do** (*Geodesic SKIZ of level h −1 inside level h*)

16: **for all** $p \in D$ with *im* [p] = **do** (*mask all pixels at level h*)

17: (*these are directly accessible because of the sorting step*)

18: *lab* [p] ← mask

19: **if** p has a neighbor q **with** (*lab* [q] > 0 or *lab* [q] = WSHED) **then**

20: (*Initialize queue with neighbours at level h of current basins or watersheds*)

21: *dist* [p] ← 1; *fifo_add* (p, queue)

22: **end if**

23: **end for**

24: curdist ← 1; *fifo_add* (*FICTITIOUS, queue*)

25: **loop** (*extend basins*)

26: p ← *fifo_remove* (*queue*)

27: **if** p = *FICTITIOUS* **then**

28: **if** *fifo_empty* (*queue*) **then**

29: BREAK

30: **else**

31: *fifo_add* (*FICTITIOUS, queue*); *curdist* ← *curdist* + 1;

32: p ← *fifo_remove* (*queue*)

33: **end if**

34: **end if**

35: **for all** $q \in N_G(p)$ **do** (*labeling p by inspecting neighbors*)

36: **if** *dist*[q] < *curdist* **and** (*lab*[q] > 0 or lab[q] = WSHED) **then**

37: (*q belongs to an existing basin or to watersheds *)

38: **if** *lab*[q] > 0 **then**

39: **if** *lab*[p] = MASK **or** *lab*[p] = WSHED **then**

40: *lab*[p] ← *lab*[q]

41: **else if** *lab*[p] ≠ *lab*[q] **then**

42: *lab*[p] ← WSHED

43: **end if**

44: **else if** *lab*[p] = MASK **then**

45: *lab*[p] WSHED

46: **end if**

47: **else if** *lab*[q] = MASK **and** *dsit*[q] = 0 then (*q is plateau pixel*)

58: *dsit*[q] ← *curdist* + 1; *fifo_add* (p, queue)

49: **end if**

50: **end for**

Continued

51: **end loop**

52: (*detect and process new minima at level *h**)

53: **for all** $p \in D$ **with** *im*[p] = h **do**

54: *dist* [p] ← 0 (*reset distance to zero*)

55: **if** *lab* [p] = MASK **then** (*p is inside a new minimum*)

56: curlab ← curlab + 1; (*create new label*)

57: *fifo_add* (p, queue); lab[p] ← curlab

58: **while not** *fifo_empty* (queue) **do**

59: q ← *fifo_remove* (queue)

60: **for all r**   ∈_G (q) do   (*inspect neighbours of *q**)

61: **if** lab [r] = MASK **then**

62: *fifo_add* (r, queue); *lab* [r] ← cur lab

63: **end if**

64: **end for**

65: **end while**

66: **end if**

67: **end for**

68: **end for**

69: (*End Flooding*)

**Table 2.** The entropy for original images with segmented results.

Name of Finger	Entropy		
	Entropy of Original Image	Entropy of Segmented Image	Percentage of Reduction
Little Finger	5.491650659963482	0.9847813571240879	82.07%
Ring Finger	6.147308152748742	0.9960550245994975	83.79%
Middle Finger	5.924018496000674	0.987188916509631	83.33%
Grooming Finger	5.739202551639768	0.9866855429259958	82.81%
Thumb Finger	5.879363918691973	0.9948162715104257	83.08%

(a)   (b)   (c)   (d)   (e)   (f)
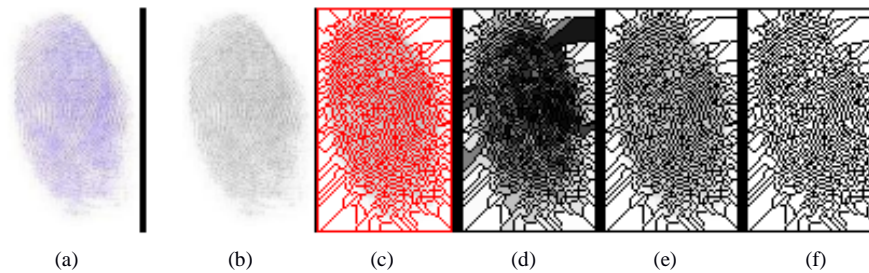
Figure 2. Application of this approach on the little finger.

(a)   (b)   (c)   (d)   (e)   (f)
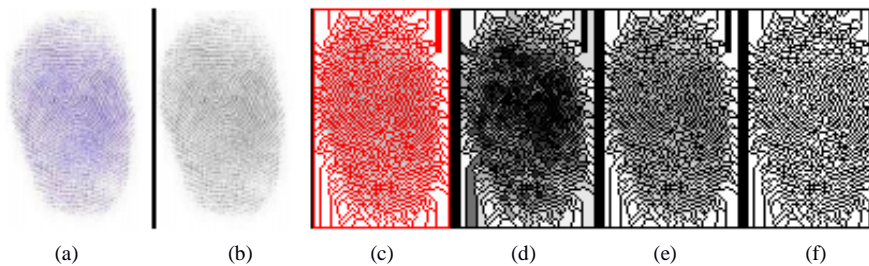
Figure 3. Application of this approach on the ring finger.
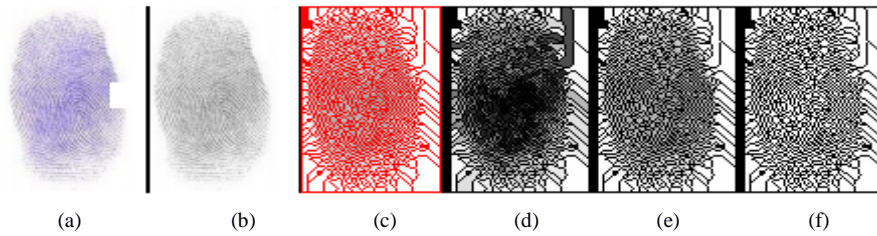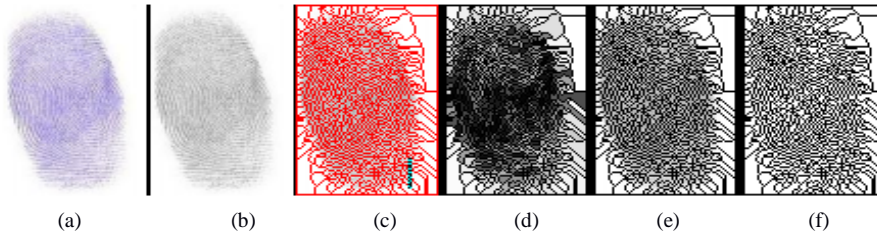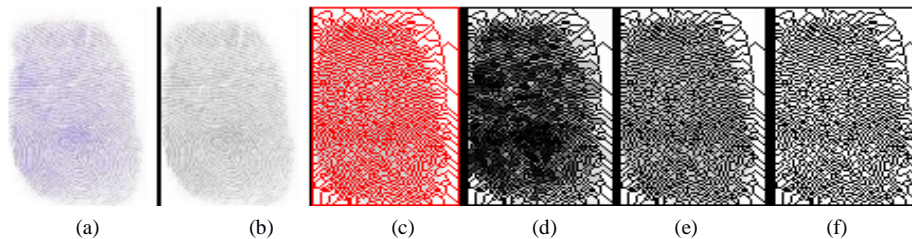
Figure 4. Application of this approach on the middle finger.

Figure 5. Application of this approach on the grooming finger.

Figure 6. Application of the approach on the thumb finger.

tropy of variable $x_1, \cdots, x_n$ is defined by:

$$H(x_1, \cdots, x_n) = -\sum_{x_1} p \cdots \sum_{x_n} p(x_1, \cdots, x_n) \log_2 \left[p(x_1, \cdots, x_n) \right]$$

The performance of segmentation algorithm can be measured with the help of entropy and as in term of visual quality of the original image and the resulted image. The image entropy can provide a good level of information to describe a given image.

Low entropy images have very little contrast and large runs of pixels with the same values. An image that is perfectly flat will have entropy of zero. Consequently, they can be compressed to a relatively small size. On the other hand, high entropy images such as an image of heavily cratered areas on the moon have a great deal of contrast from one pixel to the next and consequently cannot be compressed as much as low entropy images.

Comparison with Related Works

As depicted earlier, the research by [5] is a good threshold to adjudge the efficiency of our algorithm. In the end, the case study images selected is random and entropies also become random. However, the images selected by [5] have almost similar entropies to our images and so the percentage of reduction of their original images to segmented images is presented in **Table 3**.

The difference comparison is shown in **Table 4**.

As shown, we can see that the implementation of watershed algorithm in our scheme offers better computation (and hence, better storage reduction) capacity.

5. Summary and Conclusions

This approach has provided an easy method of the segmentation of the human fingerprints based on watershed

Table 3. The entropy for original images with segmented results [5].

Name of Finger	Entropy		
	Entropy of Original Image	Entropy of Segmented Image	Percentage of Reduction
Little Finger	5.4766	1.0281	$[(5.4766 - 1.0281)/5.4766 * 100] = 81.23\%$
Ring Finger	5.5974	0.9537	$[(5.5974 - 0.9537)/5.5974 * 100] = \mathbf{82.96\%}$
Middle Finger	5.6865	0.9820	$[(5.6865 - 0.9820)/5.6865 * 100] = \mathbf{82.73\%}$
Grooming Finger	5.2182	0.9620	$[(5.2182 - 0.9620)/5.2182 * 100] = \mathbf{81.56\%}$
Thumb Finger	5.4733	0.9969	$[(5.4733 - 0.9969)/5.4733 * 100] = \mathbf{81.78\%}$

Table 4. Comparing our scheme and [5].

Entropy		
Percentage of Reduction, [5]	Percentage of Reduction, Our Scheme	Difference in Percentages
81.23%	82.07%	0.84
82.96%	83.79%	0.83
82.73%	83.33%	0.60
81.56%	82.81%	1.25
81.78%	83.08%	1.30

transformation. The concept of watershed algorithm has been used for segmentation purpose. In practical applications, fingerprints are unique feature for identification and verification of humans, and as well as we need to maintain several databases for storing the images of fingerprints and the sizes of the databases are a major concerned issue. We can store the segmented images of fingerprints instead of the original images to reduce the size of the databases. The final result of segmentation depends upon the quality of scanner and the inkpad which we use.

Thus method is therefore recommended in achieving the aim of optimizing the size of image database.

References

[1] Das, S. Lecture Notes. IIT Madras, India. http://vplab.iitm.ac.in/courses/CV_DIP/PDF/lect-Segmen.pdf

[2] Vincent, L. and Soille, P. (1991) Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, **13**, 583-598. http://dx.doi.org/10.1109/34.87344

[3] Digabel, H. and Lantuejoul, C. (1978) Iterative Algorithms. *Proceedings of the 2nd European Symposium Quantitative Analysis of Microstructures in Material Science*, *Biology and Medicine*, 85-89.

[4] Beucher, S. and Meyer, F. (1993) The Morphological Approach to Segmentation: The Watershed Transformation, Mathematical Morphology in Image Processing. Marcel Dekker Inc., New York, 433-481.

[5] Acharjya, P.P. and Ghoshal, D. (2012) An Effective Human Fingerprint Segmentation Method Using Watershed Algorithm. *International Journal of Computer Applications*, **53**. http://dx.doi.org/10.5120/8482-2422

[6] Ulbsibiu, R. (2014) Watershed Segmentation. http://remus.ulbsibiu.ro/teaching/courses/docs/acs/Watershed%20Segmentation.doc

[7] Haralick, R.M. and Shapiro, L.G. (1985) Image Segmentation Techniques. *Computer Vision*, *Graphics and Image Processing*, **29**, 100-132. http://dx.doi.org/10.1016/s0734-189x(85)90153-7

[8] Pal, N.R. and Pal, S.K. (1993) A Review on Image Segmentation Techniques. *Pattern Recognition*, **26**, 1277-1294. http://dx.doi.org/10.1007/978-3-662-21817-4_10

[9] Berthold, K. and Horn, P. (1986) Robot Vision. MIT Press, Cambridge.

[10] Ross Beveridge, J., Griffith, J., Kohler, R.R., Hanson, A.R. and Rise-Man, E.M. (1989) Segmenting Images Using Localized Histograms and Region Merging. *International Journal of Computer Vision*, **2**, 311-347.

http://dx.doi.org/10.1007/bf00158168

[11] Roerdink, J.B.T.M. and Meijster, A. (2001) The Watershed Transform: Definitions, Algorithms and Parallelization Strategies. *Fundamenta Informaticae*, **41**, 187-228.

[12] http://imagej.nih.gov/ij/

Scientific Research Publishing (SCIRP) is one of the largest Open Access journal publishers. It is currently publishing more than 200 open access, online, peer-reviewed journals covering a wide range of academic disciplines. SCIRP serves the worldwide academic communities and contributes to the progress and application of science with its publication.

Other selected journals from SCIRP are listed as below. Submit your manuscript to us via either submit@scirp.org or Online Submission Portal.