Watershed Segmentation for Binary Images with Different Distance Transforms

Qing Chen[1], Xiaoli Yang[2], Emil M. Petriu[1]
[1]*University of Ottawa, Ottawa, ON, Canada*
[2]*Lakehead University, Thunder Bay, ON, Canada*
E-mail: {qingchen, petriu}@site.uottawa.ca, lucy.yang@lakeheadu.ca

Abstract

This paper evaluates and compares the performances of watershed segmentation for binary images with different distance transforms including Euclidean, City block and Chessboard.

1. Introduction

Image segmentation is a process that partitions an image into its constituent regions or objects [1]. Effective segmentation of complex images is one of the most difficult tasks in image processing. Various image segmentation algorithms have been proposed to achieve efficient and accurate results. Among these algorithms, watershed segmentation is a particularly attractive method.

The major idea of watershed segmentation is based on the concept of topographic representation of image intensity. Meanwhile, Watershed segmentation also embodies other principal image segmentation methods including discontinuity detection, thresholding and region processing [1]. Because of these factors, watershed segmentation displays more effectiveness and stableness than other segmentation algorithms.

We evaluated and compared the performance of watershed segmentation for binary images with different distance transforms. In this paper, image segmentation is more specifically referred to the task of extracting components or particles of interest from the image as precisely as possible. Segmentation is not only simply counting the number of the components and pointing out them from the image, it also encompasses the extraction of the components contours.

2. Basic Concepts

As pointed already, the basic concept of watershed is based on visualizing a gray level image into its topographic representation, which includes three basic notions: minima, catchment basins and watershed lines. Fig. 1 illustrates the meanings of these definitions. In the image of Fig. 1.(a), if we imagine the bright areas have "high" altitudes and dark areas have "low" altitudes, then it might look like the topographic surface illustrated by Fig. 1.(b). In this surface, it is natural to consider three types of points: (1) points belonging to the different minima; (2) points at which water would fall with certainty to a single minimum; and (3) points at which water would be equally likely to fall to more than one minimum. The first type of points forms different minima of the topographic surface. The second type points which construct a gradient interior region is called catchment basin. The third type of points form crest lines dividing different catchment basins, which is termed by watershed lines [1].

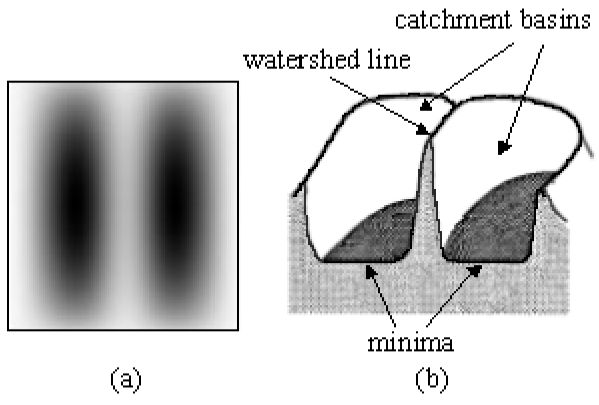


Fig. 1. (a) A gray level image. (b) The topographic surface of (a).

The objective of watershed segmentation is to find all of the watershed lines (the highest gray level). The most intuitive way to explain watershed segmentation is the *Immersion Approach*: imagine that a hole is drilled in each minimum of the surface, and we flood water into different catchment basins from the holes. If the water of different catchment basins is likely to merge due to further immersion, a dam is built to prevent the merging. This flooding process will eventually reach a stage when only the top of dam (the watershed lines) is visible above the water line.

An efficient algorithm to implement this approach proposed by Luc Vincent and Pierre Soille [3] involves two steps:

(1) *The sorting step*. In this step, the algorithm first determines the frequency of each gray level of the image, which allows allocating the needed memory for each list of pixels. Then, the algorithm scans the image a second time and inserts the pixels in the list corresponding to their gray level value in increasing order.

(2) *The flooding step*. Suppose we begin the flooding from the minimum of gray level h, and X_h is made of the pixels belonging to the minimum. Now, we consider the gray level $h+1$. Y_{h+1} is one component made of the pixels of gray level $\leq h+1$. We can find three possible relations between X_h and Y_{h+1} illustrated by Fig. 2:

- $Y_{h+1} \cap X_h = \varnothing$: in this case Y_{h+1} is obviously a new minimum at gray level $h+1$;
- $Y_{h+1} \cap X_h \neq \varnothing$ and is connected: in this case, Y_{h+1} belongs to the catchment basin corresponding to the minimum of X_h.
- $Y_{h+1} \cap X_h \neq \varnothing$ and is not connected: in this case, we can infer that Y_{h+1} must contain different minima at gray level h.

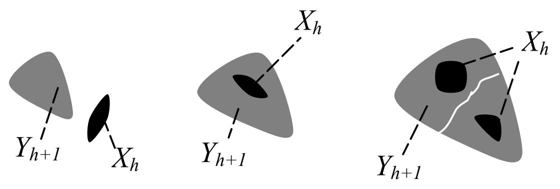


Fig. 2. The possible relationships between X_h and Y_{h+1}.

3. Distance Transforms

Watershed segmentation produces good results for gray level images with different minima and catchment basins. For binary images, however, there are only two gray levels 0 and 1 standing for black and white. If two black blobs are connected together in a binary image like Fig. 3, only one minimum and catchment basin will

be formed in the topographic surface. To use watershed to segment the connected blobs, we need to use distance transforms (DTs) to preprocess the image to make it suitable for watershed segmentation.

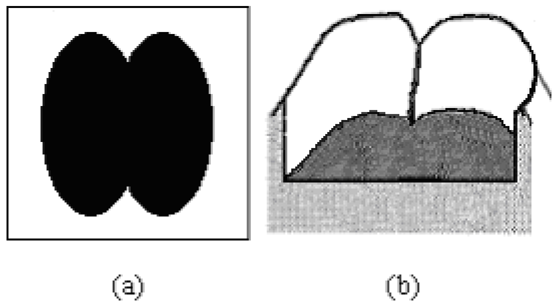


Fig. 3. (a) A binary image. (b) The topographic surface of (a).

In this article, we define the DT of a binary image as the distance from every pixel of the object component (black pixels) to the nearest white pixel. There are many different ways to define the distance between two pixels $[i_1, j_1]$ and $[i_2, j_2]$ in a digital image. Several commonly used DT functions [4] for image processing are:

Euclidean:

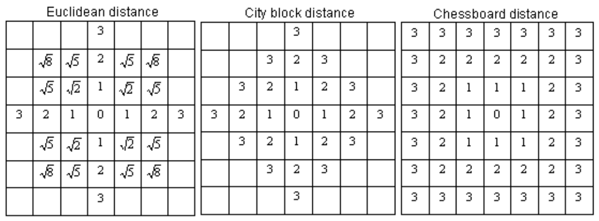
$$d_{Euclidean}([i_1, j_1],[i_2, j_2]) = \sqrt{(i_1 - i_2)^2 + (j_1 - j_2)^2}$$

City block:

$$d_{Cityblock}([i_1, j_1],[i_2, j_2]) = | i_1 - i_2 | + | j_1 - j_2 |$$

Chessboard:

$$d_{Chessboard}([i_1, j_1],[i_2, j_2]) = \max(| i_1 - i_2 |, | j_1 - j_2 |)$$



Euclidean distance

			3			
	$\sqrt{8}$	$\sqrt{5}$	2	$\sqrt{5}$	$\sqrt{8}$	
	$\sqrt{5}$	$\sqrt{2}$	1	$\sqrt{2}$	$\sqrt{5}$	
3	2	1	0	1	2	3
	$\sqrt{5}$	$\sqrt{2}$	1	$\sqrt{2}$	$\sqrt{5}$	
	$\sqrt{8}$	$\sqrt{5}$	2	$\sqrt{5}$	$\sqrt{8}$	
			3			

City block distance

			3			
		3	2	3		
	3	2	1	2	3	
3	2	1	0	1	2	3
	3	2	1	2	3	
		3	2	3		
			3			

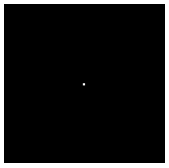
Chessboard distance

3	3	3	3	3	3	3
3	2	2	2	2	2	3
3	2	1	1	1	2	3
3	2	1	0	1	2	3
3	2	1	1	1	2	3
3	2	2	2	2	2	3
3	3	3	3	3	3	3

Fig. 4. Different DT discs of radius $k = 3$.

A set of pixels at distance $\leq k$ according to a distance metric is called a *disc* of radius k [2]. Fig. 4 shows discs of 3 according to the distance functions.

Fig. 5 illustrates the effects of applying different DTs on a binary image of 200×200 pixels with the center pixel set to 1 (white) with gray level lines.



A binary image of 200×200 pixels

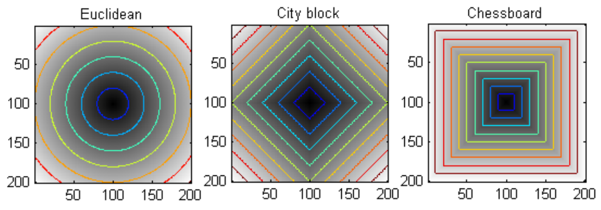
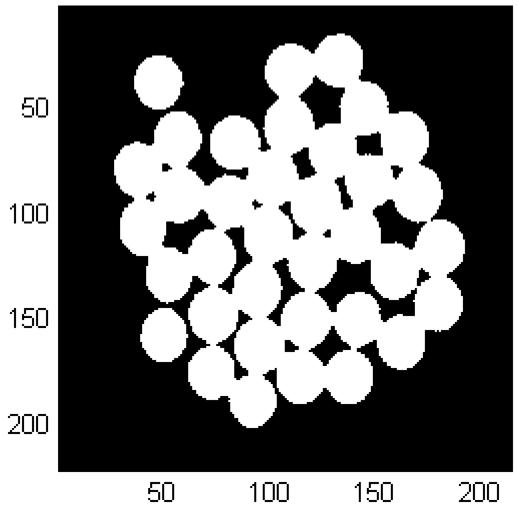


Fig. 5. Different DT effects with gray level lines.

4. Effects Comparisons for Different DTs

With DTs, a binary image can be converted to a gray level image, which is suitable for watershed segmentation. However, different DT functions produce different effects. We will compare the watershed segmentation effects with an example in this section.

Fig. 6 shows the watershed segmentation effects with different DTs for the binary image of a bunch of chocolate beans. We can see apparently that the chessboard achieves the best effect among the three segmentation result figures.



Original binary image of the chocolate beans

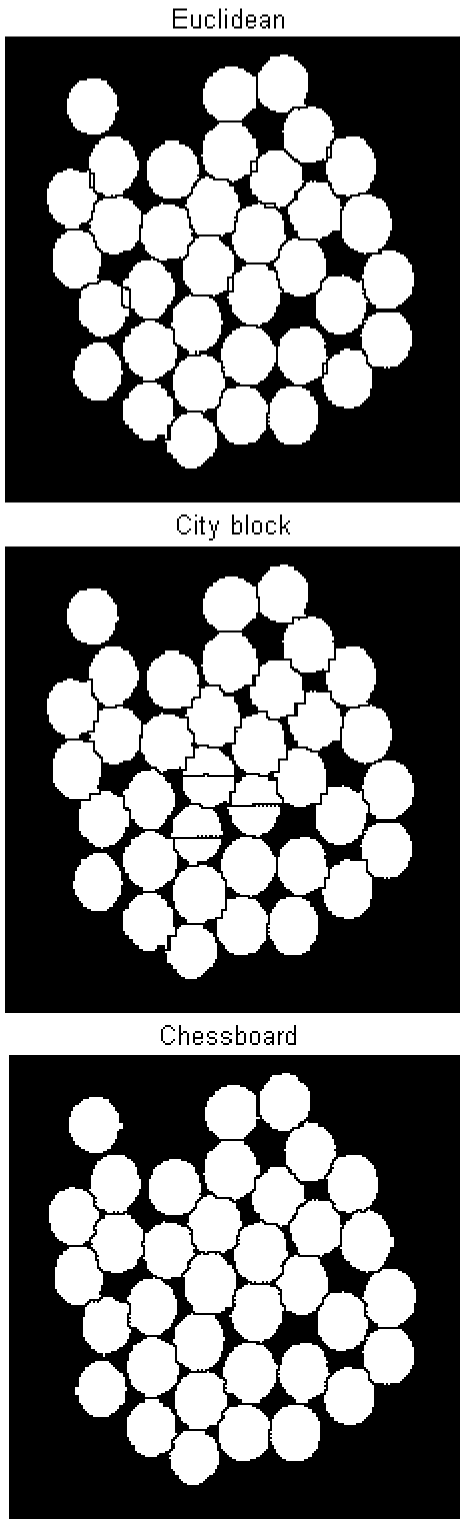


Fig. 6. The watershed segmentation results for the chocolate beans with different DTs.

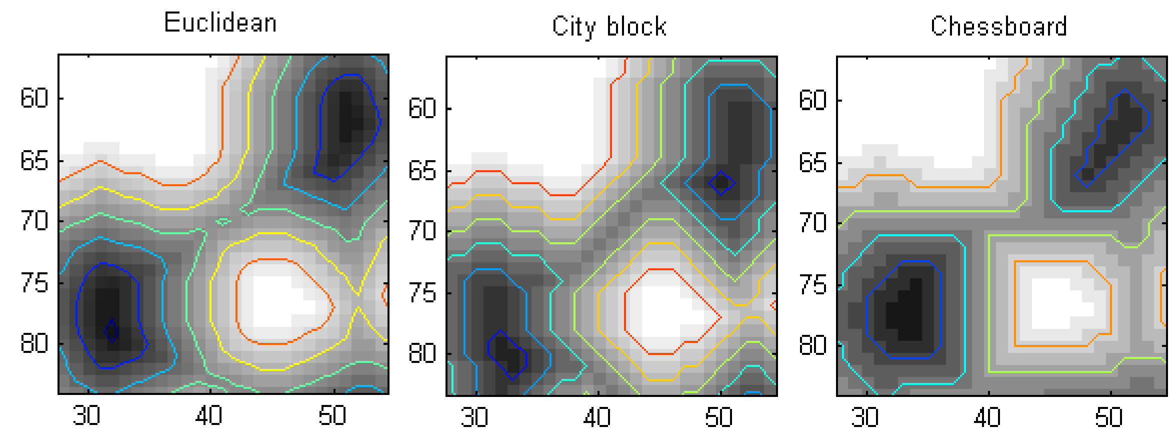


Fig. 7. “Salt and Pepper” over segmentation caused by Euclidean DT.

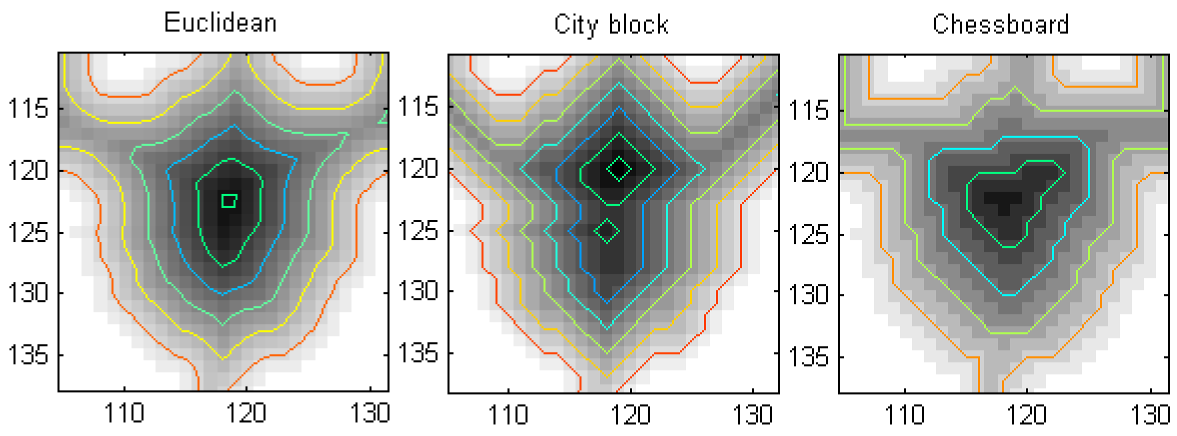


Fig. 8. Component over segmentation caused by City Block DT.

5. Analysis

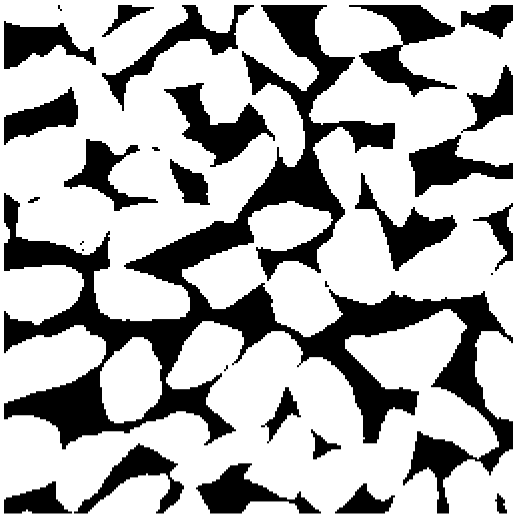
We will analyze the performances of these different DT algorithms in this section based on the example we showed in last section. Two more examples with for binary images with different component shapes will also be provided to give further proof for our analysis.

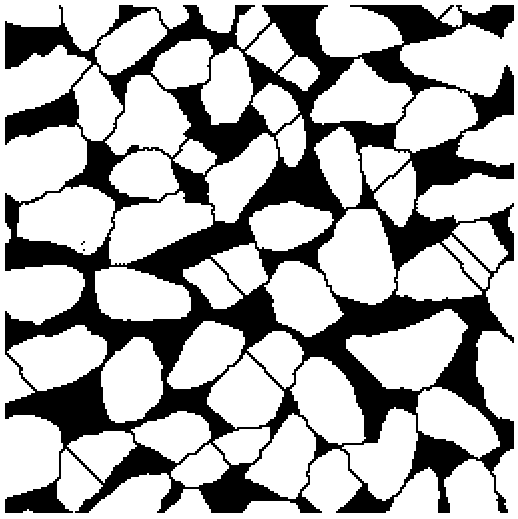
Euclidean DT has a higher possibility of “salt and pepper” over segmentation. The reason is that Euclidean propagates to the neighbor pixels in the shape close to a round circle, and it is very easy to form a small island made of a few pixels between different components. When watershed is implemented, the small island will be treated as a separate minimum and forms the “salt and pepper” in the image. Fig. 7 illustrates this scenario.

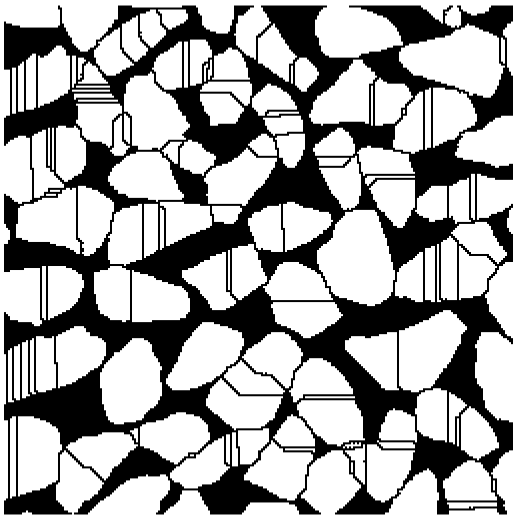
City Block DT has a higher possibility of over segmentation for the components in the image. The reason is that City Block DT propagates to the neighborhood in the shape of diamond, and it has a very strong tendency to form multiple minima at the center area of the component where the pixels have different gray levels. When the watershed is implemented, due to the multiple minima at the center, one component will be segmented into different parts. Fig. 8 showed this scenario.

Chessboard DT has a better pruning effect due to its square shape propagation. It can effectively remove the jaggedness formed in the Euclidean DT and avoid the components over segmentation caused by City Block DT. However, Chessboard DT has its own shortcoming. If two components have a high percentage of boundary overlap, Chessboard DT might be unable to separate the overlapped components.

Fig. 9 and 10 show another two examples of watershed segmentation effects with different DTs for the binary images of rocks and coffee beans. The shapes of the rocks are in irregular forms while the coffee beans are more close to ellipse. Table 1 summarizes the number of components and errors produced by the different DTs for the three examples. All examples show that the Chessboard DT achieves the best watershed segmentation result among the three different DTs.


Original binary image of the rocks


Chessboard


Euclidean

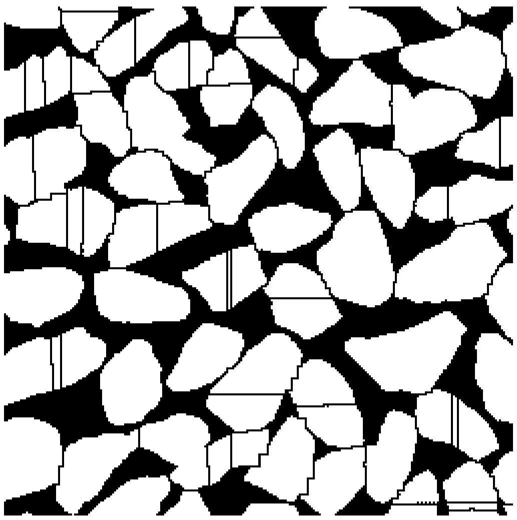
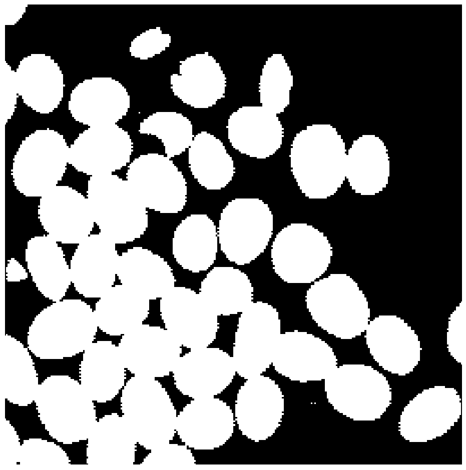
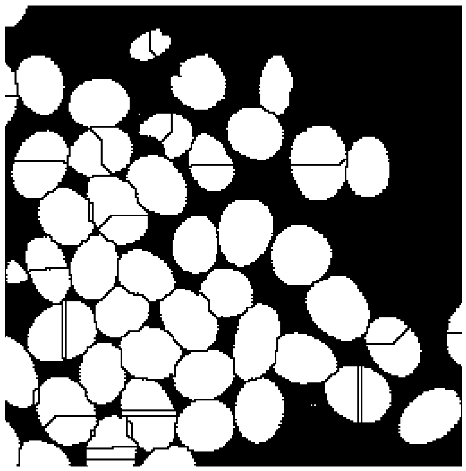
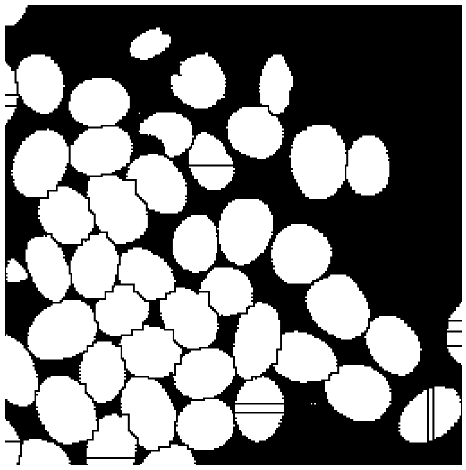

Cityblock

Fig. 9. The watershed segmentation results for the binary image of the rocks with different DTs.


Original binary image of the coffee beans


Euclidean


Cityblock

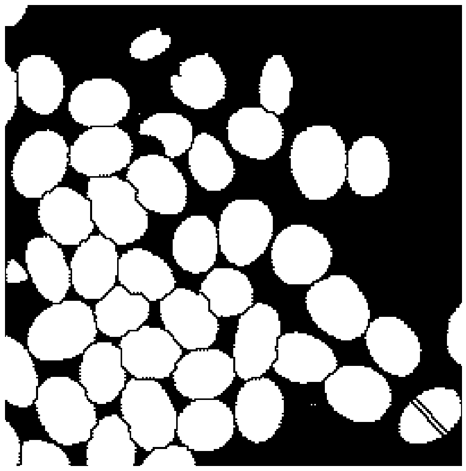

Chessboard

Fig. 10. The watershed segmentation results for the binary image of the coffee beans with different DTs.

6. Conclusions

Watershed segmentation is an effective method for gray level image segmentation. To apply watershed segmentation to binary images, we need to preprocess the binary images with distance transform to convert it to gray level images which are suitable for watershed segmentation.

The common DTs include Euclidean, City block and Chessboard. Different DTs produce very different watershed segmentation results for the binary images. For images containing components of different shapes, we find that the Chessboard DT can achieve better watershed segmentation results than Euclidean DT and City block DT.

Image ID	*NOC by manual count*	*NOC by euclidean DT*	*NOC by cityblock DT*	*NOC by chess board DT*
Chocolate beans	36	43 error: +19%	39 error: +8%	36 error: 0%
Rocks	60	162 error: +170%	88 error: +47%	72 error: +20%
Coffee beans	50	74 error: +48%	62 error: +24%	52 error: +4%

Table 1. Number of components (NOC) and errors produced by different DTs for the three images.

7. References

[1] Rafael C. Gonzalez and Richard E. Woods, *Digital Image Processing*, Prentice-Hall, 2002, pp. 617-620.

[2] Ramesh Jain, Rangachar Kasturi, and Brian G. Schunck, *Machine Vision*, McGraw-Hill, 1995, pp. 53 -54.

[3] Luc Vincent and Pierre Soille, "Watersheds in Digital Spaces: An Efficient Algorithm Based on Immersion Simulations," *IEEE Transactions of Pattern Analysis and Machine Intelligence*, Vol. 13, No. 6, June 1991, pp. 583-598.

[4] The Mathworks online documentation, Matlab Image Processing Toolbox, http://www.mathworks.com/access/helpdesk/help/toolbox/images/images.shtml.