Application of the Watershed Boundary Technique to Automatically Segment Surface Froth Images

By Jia Liu

Thesis presented for the Degree of Master in the Department of Electrical Engineering, University of Cape Town.
Cape Town, September 1995

Declaration

I declare that this dissertation is my own work. It is being submitted for the Degree of Master in Engineering at the University of Cape Town. It has not been submitted before for any degree or examination at this or any other university.

Signed by candidate

Jia Liu
(Signature of Candidate)

Acknowledgements

I would like to thank my Supervisor, Professor G. de Jager, firstly for proposing research into this exciting field of engineering, and secondly for his continued assistance throughout the period of development. I would also like to thank Robert Charles Crida for his assistance with the watershed segmentation software. Special thanks go to Professor G. Mclaren, his wife, and to Peter Golda for all their personal and technical help and advice. Finally, thanks go also to the members of the image processing group and others who assisted with the production of this thesis.

Abstract

The purpose of this dissertation is to investigate the suitability of using a recently proposed computer processing algorithm – the **watershed boundary technique** for applications in computer vision systems, where on-line segmentation of the surface froth images in commercial flotation cells is required.

In industrial flotation cells, the surface froth offers considerable visual information as to the grade and recovery in extraction and the concentration of minerals in raw ores. The individual bubbles that constitute the surface froth give rise to complex three-dimensional structures that are problematic to segment accurately and reliably by computer vision. Investigating an efficient technique to automatically and accurately extract these visual features in real time is therefore the main task of this research work.

Past research work into this field has resulted in a number of different techniques and algorithms for the purpose of segmentation. However, these algorithms are often iterative and therefore quite slow. The technique described here is non-iterative and therefore one with industrial real time processing implications.

The results show that the watershed boundary technique provides a reliable method for the segmentation of surface froth structures. Minor errors which occur do not significantly influence the statistical parameters which can be determined from the segmented images. This technique is not only very successful in segmentation, but may also be implemented for real-time operation. This clearly leads to a viable new segmentation method.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

With the current depth of development in the field of image processing research, the art of image analysis has reached the stage where a large number of potential computer vision applications are starting to become feasible. Whilst much work has been done to improve these computer vision algorithms, most applications are regarded as infeasible due to the lengthy processing times involved.

One such application is the determination of bubble size and shape distributions in a froth structure. In the mining industry, the froth structures on the surfaces of flotation tanks provide substantial visual information regarding the grade and recovery of the mineral being extracted. Consequently, the froth bubble size and shape distributions prove descriptive of the froth structure. The ability to accurately and reliably measure froth structure size and shape distributions using computer vision is of considerable interest to the industry and such a technique could have far reaching consequences in the optimisation of the control of industrial flotation cells.

This dissertation discusses the research work into the application of the **Watershed Segmentation Technique** for the analysis of surface froth structures described above, and the results thereof.

Chapter 2 provides some background on the flotation cell mineral extraction industry, which is the source of all froth images used here. Chapter 3 introduces the surface froth images, discusses some problems in the segmentation thereof and

investigates some solutions. Chapter 4 introduces the general concepts behind mathematical morphology, which forms the theoretical basis of the watershed segmentation technique. In Chapter 5, watershed methodology is discussed and the watershed algorithm relevant to this research work is explained. Chapters 6, 7 and 8 provide details of an investigation into the watershed processing algorithm. The consequent software design and implementation is then detailed. Chapter 9 presents the results of the application of the watershed segmentation technique to the surface froth images. The analysis of these results, together with the analysis methods perused is then given. Chapter 10 expands on the results and tests by providing seven more processed images with the appropriate analyses and distribution graphs. Chapter 11 proffers conclusions and recommendations.

Chapter 2

Froth Floatation and Bubbles

In this dissertation, all the bubble images for testing were taken by a development machine vision system which is capable of characterising the froth structures prevalent on the surfaces of industrial flotation cells. For a better understanding of the purpose of this research work and the relationships between bubble images and the mining mineral processing industry, a simple introduction to the industrial processes involved is given in this chapter.

2.1 Froth Floatation

It is well known that since most minerals exist as a mixture of extractable minerals and extraneous rocky material, extraction and concentration is therefore the main purpose of the minerals processing industry.

Minerals are separated into distinct products by the exploitation of differences in the chemical and/or physical properties of the various species. The valuable minerals are presented to the separating forces as individual, unlocked or liberated grains. Figure 2.1 [7] shows a typical, simplified flowsheet of a mineral processing operation.

Froth floatation is a primary separation technique based on the differences between the surface characteristics of mixed, liberated mineral particles. Basically,

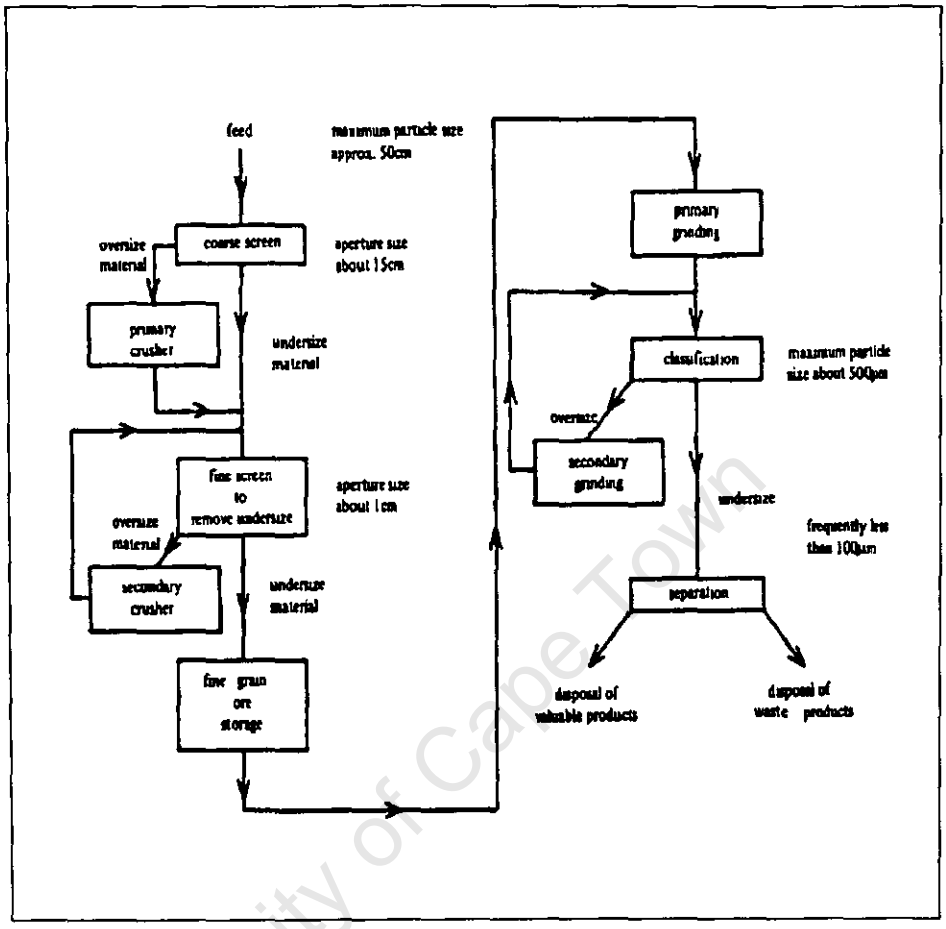


Figure 2.1: Simplified flowsheet of a mineral processing operation.

these small mineral particles are captured by bubbles and collected in the form of a froth. Thus, the froth floatation process is an enormously important technique in minerals processing, so much so that without floatation many familiar metals and inorganic raw materials would be exceedingly scarce and costly [7].

Nowadays in the minerals industry, research tends to search for new methods of high **recovery** at low **grade** [21] and more 'specific' chemical reagents which might show strong chemisorption on the particular minerals. This has greatly aided the floatation technique development. Note that recovery is a measure of how effectively the separation process has extracted the valuable mineral content

from the liberated ore. A suitable definition is given by

$$\text{recovery } (\%) = \frac{\text{precentage (or mass) of valuable in product stream}}{\text{percentage (or mass) of valuable in input stream}} \times 100$$

Further, grade is a measure of the quality of the concentrate of a separation process. A suitable definition is given by

$$\text{grade } (\%) = \frac{\text{mass of valuable in stream}}{\text{mass of valuable and waste in stream}} \times 100$$

Mineral floatation may be briefly described as follows: One begins by grinding the ore with water and a reagent, which results in what is referred to as an *imeral-water slurry*, down to a chosen average grain size to secure *liberation* of the discrete mineral particles. The imeral-water slurry is commonly passed through a hydrocylone to concentrate the *sands* and reject the *slimes* (the slimes being the finest size fractions). This stage is known as conditioning. Then, the conditioned pulp is run into a floatation cell, which is basically a container with a stirrer and thus a means of introducing air into the pulp [7]. Figure 2.2 shows this entire process diagrammatically. Further, the entire contents of the cell are beaten together vigorously, as shown in Figure 2.3 [7]. The following section discusses this last detail more thoroughly.

2.1.1 The Floatation Cell and Froths

Essentially, the cell is a mechanically agitated bath which uses a rotating impeller to thoroughly mix the pulp with the incoming air. The impeller breaks up the air supply resulting in fine air bubbles that disperse throughout the agitated suspension. Refer here again to Figure 2.3.

Adhesion and entrainment are two important mechanisms by which particle collection takes place in floatation. The floatation grains are caught by bubbles and carried to the top of the separation cell, forming a froth which overflows into a launder. Figure 2.4 [20] illustrates the material transport paths.

A bubble is a region in space occupied by a gas, and enclosed by a gas-liquid

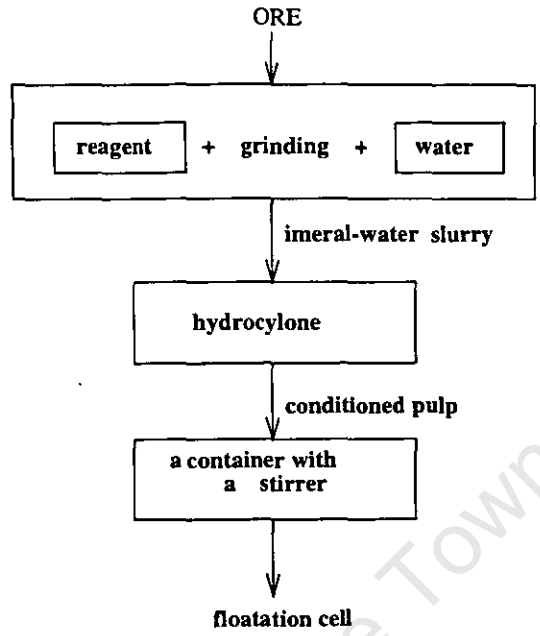


Figure 2.2: Simple flow sheet of mineral floatation processing.

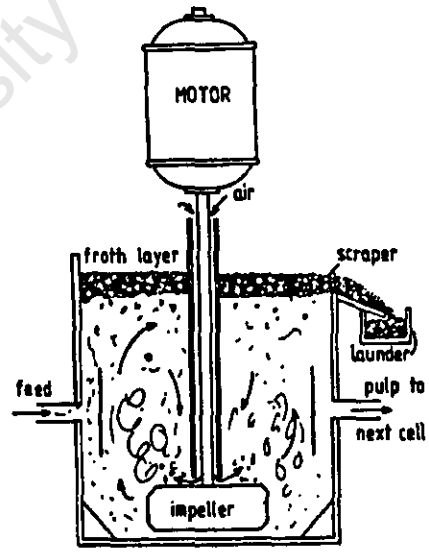


Figure 2.3: Schematic diagram of a dispersed air floatation machine as used for processing minerals.

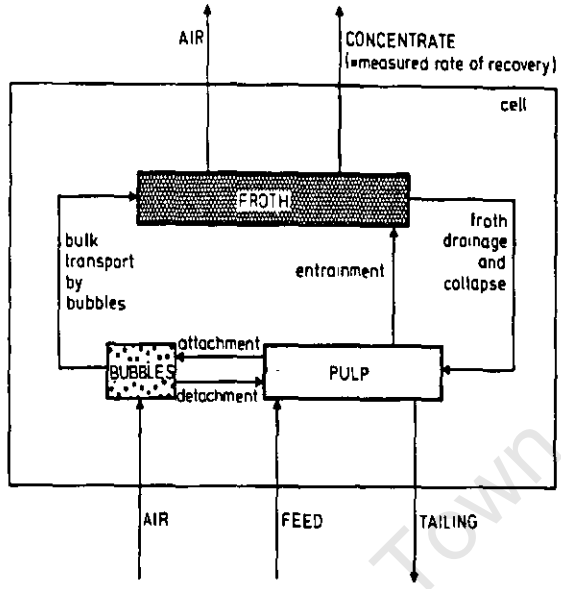


Figure 2.4: Transport paths of materials in floatation.

interface [7]. In the floatation process, bubbles must be able to adhere to the mineral surface so that the particle becomes floatable. These particles are carried into the froth by rising bubbles through the pulp phase. These bubbles with their load of particles thus accumulate at the pulp/froth interface.

On the surface of floatation cells, the froth structures provide visual information on cell efficiency, with respect to the grade and recovery of the mineral being extracted.

2.2 Froth Structures with Regard to Recovery and Grade

Practical floatation cells exhibit one of two predominant froth structures; known as wet (shallow) froths and dry (deep) froths [21].

The wet froths mainly consist of small and fast flowing bubbles, and due to the

low drainage, have a high water content. These charcateristics tend to indicate high recovery and low grade.

On the other hand, dry froths have a low water content due to the high drainage within the froth. This results in stiff and slow moving bubbles which, due to this, become elongated in the direction of flow. This is indicative of low recovery and high grade.

2.3 Froths with Regard to Bubble Size and Shape

The surface froth structure in floatation cells may be directly described by the size and shape of its constituent bubbles.

From the point of view of hydrodynamics, the equilibrium shape of the bubble is determined by stress balances at the interface. When a finite volume of gas is injected into the pulp, the bubbles take shape and rise to the interface under hydrostatic pressure. As the bubbles rise and the pulp flows, the shape of bubbles will continually change along with the variation of hydrostatic pressure.

The subject of the motion of bubbles in a large body of water has been studied by many researchers. Simply, the experimental results for air bubbles in pure water are given as follows:

1. Very small bubbles, diameter less than $1mm$, rise in straight lines with stable motion and are spherical in appearance; this spherical appearance being ensured by surface tension.
2. Medium size bubbles, diameter region 1 to $20mm$, rise with an unstable, or spiral motion and are ellipsoidal in shape. The surface tension maintains a balance of normal stresses at the interface.
3. Large bubbles, ranging beyond approximately $3cm^3$ in volume, rise relatively straight, but tend to break into smaller bubbles, and are clipped at the upper face of a sphere in appearance.

The behaviour of bubbles in viscous liquids, namely in pulp, is very similar to

that in water, in that the shapes of the terminal velocity curves are similar. Spherical, ellipsoidal and spherical capped bubbles are seen in all liquids [7].

From another viewpoint, froth structure varies greatly with froth height. Viewing from just above the pulp/froth interface, the bubbles are small in size, and closely packed spherical in shape. As the height of the froth increases, the bubbles tend to knit more loosely with each other to form polyhedral shapes and much of the water content drains away, leaving the bubbles less elastic. At the froth depth where the bubble structure starts to change from close-packed spheres to the polyhedral form, the beneficiation of raw mineral effectively stops. Therefore, an optimum froth height exists, where the liquid lamella reaches a critical value, which is related to the particle size and the bubble surface tension. A higher froth results in broken bubbles and loss of valuable material, hence, a maximum froth height is defined for a particular set of conditions, and it is an indication of maximum selectivity at a specified recovery [21].

Note also that the factor which affects the froth structure is not only froth height, but also froth drainage, froth stability and froth aeration. For a more detailed discussion of these parameters, see Symonds [21].

The following is a summary of the factors which influence the size of a bubble in the pulp phase [15]:

1. Size of the aperture from which it emerges;
2. Hydrostatic head against which it is compressed;
3. Surface tension of the interface formed with the pulp as it emerges;
4. Speed of emergence and the volume and pressure of gas behind it;
5. Turbulence of the surrounding pulp.

An understanding of the effects of the abovementioned factors and variables can help toward proper control of floatation cells, and thus optimum performance.

2.3.1 Bubble Size Measurement

As illustrated earlier, the size of the bubbles is an important factor in the efficiency of the flotation process. The measurement of bubble size and shape provides insight into the drainage occurring within the froth structure, thus indirectly providing a measure of the grade and/or recovery of the floatation cell in question. Therefore, the investigation of new measurement techniques can contribute greatly to floatation cell control.

Generally, these size and shape measurements are performed using images of the froth surface. But, searching for the methodology which is applicable for achieving accurate measurement results in real-time is the main purpose of researchers.

A number of methods have been described by O'Connor [15], D. W. Moolman *et al* [14] for determining the sizes of bubbles in two- and three-phase systems. These include, inter alia, photographic techniques, electroresistivity measurements, gas holdup and pressure measurements and calculations using empirical or semi-empirical correlations.

Today, the development and application of computer vision analysis to the characterisation of three-phase froth structures has gathered interest among researchers. It was originally investigated by Woodburn *et al* [3] in work pertaining to the demineralization of low-rank coals. Symonds [21], his research work revolving around the use of digital image processing for the analysis the surface froth structures in floatation cells, has shown that it is possible to automatically characterise surface froth structures using a machine vision system.

The performance of floatation cells may be monitored by a practical on-line machine vision system. That means the bubble images could be taken by video camera and recorded onto tape. In this way, the complex three dimensional surface is reduced to a simpler two dimensional surface. The bubble size distribution may be obtained at specified time intervals by using digital image pocessing technology. The results of the image processing may thus be used to implement automated control of these cells.

2.3.2 The Contribution of this Dissertation

The work presented in this dissertation has the same purpose as that of Symonds', that is, the accurate measurement and distribution of the sizes and shapes of the surface froth bubbles. The test images for research were taken from the same video material, however, this disseration provides the results of an investigation into using alternative digital image processing methdology to analyse and characterise the surface froth structures in the floatation cell.

2.3.2 The Contribution of this Dissertation

The work presented in this dissertation has the same purpose as that of Symonds', that is, the accurate measurement and distribution of the sizes and shapes of the surface froth bubbles. The test images for research were taken from the same video material, however, this disseration provides the results of an investigation into using alternative digital image processing methdology to analyse and characterise the surface froth structures in the floatation cell.

Chapter 3

Surface Froth Images and the Segmentation Thereof

For the video material obtained from **MINTEK**, Randburg, Johannesburg, South Africa, surface froth images were acquired by a colour camera and converted to black and white bubble images. For computer processing, this analogue video signal from the camera must be converted to a digital signal using an A/D transfer device. Refer to Symonds [21] for a detailed discussion.

In practice, the analogue video signal from the video tape recorder is fed into an MVP-AT frame grabber (MATROX Electronic Systems Limited, 1989) where the signal is digitized at real time video rates into discrete picture pixels and is captured in the form of a black and white image, 512×512 pixels in size.

The grey scale resolution of the digitized image is clearly dependent upon the analogue to digital converter that is used. Typically, monochrome images are digitized using 8 data bits, giving a total of $2^8 = 256$ grey levels, ranging from 0 (black) to 255 (white).

3.1 The Test Surface Froth Images

The image BUB1, as illustrated in Figure 3.1, is one of the test images[1] used in the processing.

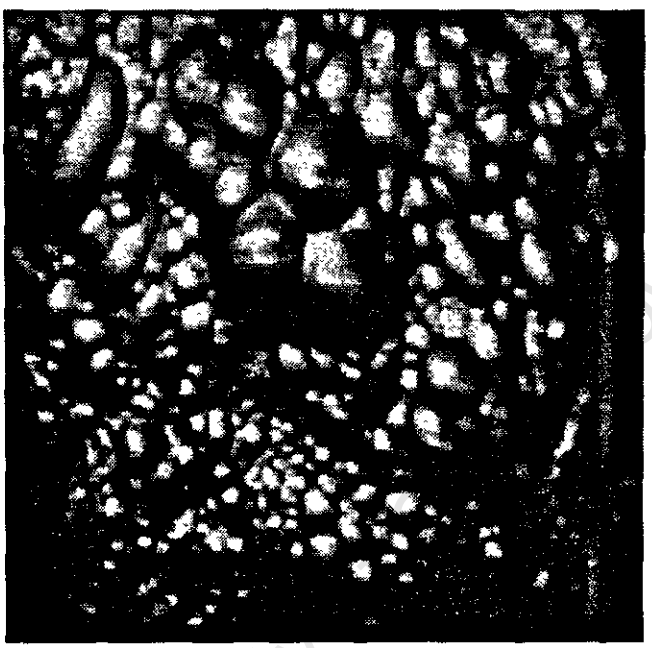


Figure 3.1: Test image - BUB1.

The first property to be noticed when viewing this surface froth image is that the bubbles are compact, that is, closely knit together, and that the top surfaces of the bubbles are light in colour, while the shadows in the image correspond to bubble boundaries. Thus it could be said that the highlights of bubbles and the bubble boundaries are basic conditions which help the human eye to distinguish bubble shape and size. In other words, these highlights and boundaries are two important visual features for the segmentation of the froth image. The human eye utilizes these features to quickly identify the individual bubbles.

[1]The test images that were studied in this disseration are the same as those used by Symonds' thesis [21] for the sake of easy comparison of processing results.

3.2 Image Segmentation

Segmentation is a major branch of image processing. It deals with image analysis or scene analysis. The input is still pictorial, but the desired output is a description of the given picture or scene.

For surface froth image processing, the fundamental processing is achieved by segmentation, which is taken to mean the process whereby the complete pixel image is separated into areas that represent individual bubbles. In other words, this segmentation comes down to the **extraction** of basic **features** of the bubble image. Thus, a binary image is produced from a grey level image and the final result of the image processing is that all the bubbles have been marked and isolated.

3.2.1 Visual Features that Characterise Froth

The bubble highlights and boundaries, as previously discussed, are two inherent visual features of bubble images.

Taking figure 3.1 as the reference image, the bubble **highlights** are the bright spots occurring on the surfaces of individual bubbles. For a high surface curvature, that is to say for a spherically shaped bubble, there will be a small highlight region on the top of the bubble; and similarly for low surface curvature, which is to say for an elongaged bubble, there is a larger highlight area on the top of the bubble. Outside of this region, since the incident light is no longer directly reflected from the bubble surface, the light intensity falls off towards the bubble boundary [21].

Therefore, at bubble **boundaries** the luminance is at a comparative minimum, and hence the gray levels of the boundary pixels should be at lower values as compared to their neigbours. In other words, the bubble boundary pixels correspond to valleys in a gray value graph along a scan row of the bubble image. Figure 3.2 shows such a scan line at row 200 of image BUB1 and Figure 3.3 shows the corresponding gray level graph. The bubble boundaries can be determined by the chain of boundary pixels of a connected bubble region which is exposed

at the froth surface.

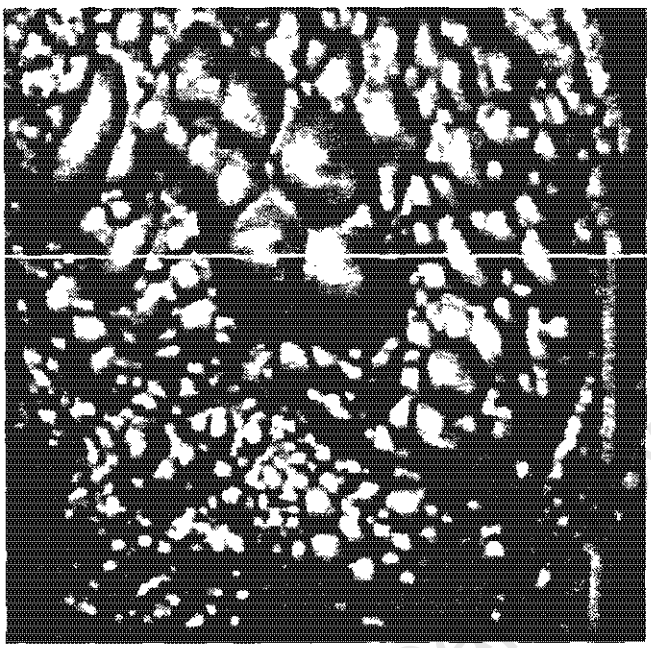


Figure 3.2: Scan line at row 200 of image BUB1.

3.2.2 The Segmentation Problem

In practice, the bubble boundaries that one tries to extract from images are not clearly defined, as mentioned previously. Variables such as complex varying illumination effects, the camera position relative to the bubbles, and the surface reflectance properties of the bubbles, pose several prominent problems for segmentation processing. More formally, these may be listed as follows:

- Bubble highlight saturation.
- Big size bubbles have two highlights because two lighting sources are used.
- The sharpness of some bubble boundaries are masked by a shadow region, since the angle of lighting is obtuse [21].

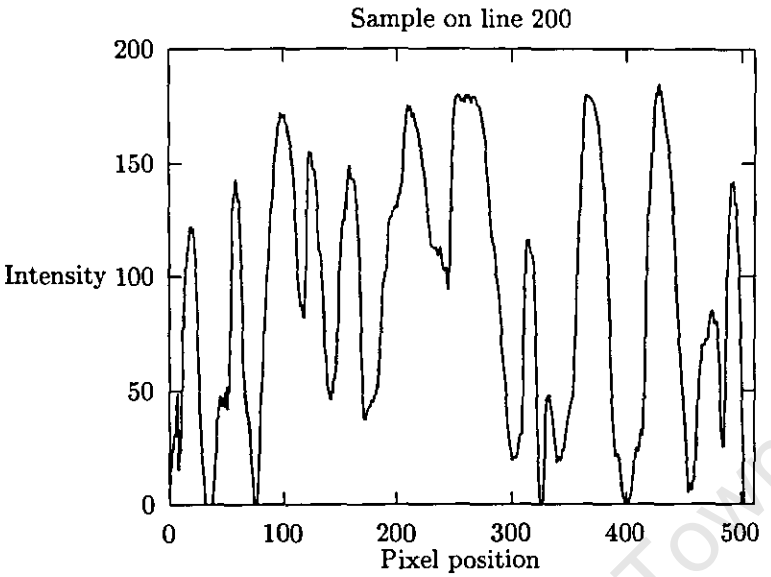


Figure 3.3: Grayvalue graph of image BUB1 at pixel row 200.

As a result of these above mentioned factors, segmentation may produce two types of error:

1. Extra regions may be generated.
2. Valid regions may missing from the segmented image.

These two errors may be combined in such a way that the shapes of segmented regions do not match accurately the object boundaries. Therefore, to achieve each bubble shape and size accurately, the segmentation process is considerably more complex and difficult.

3.2.3 Segmentation Techniques

It should be emphasized that there is no single standard approach to segmentation. Many different types of images or scene parts can serve as the segments on which descriptions are based, and there are many different ways in which one can attempt to extract these parts from the picture. Successful segmentation must

be judged by the utility of the description that is obtained using the resulting objects [18].

For successful separation or reliable segmentation of a digitised froth image into the individual bubbles, the result should show that all these individual bubbles have been separated as such. The accuracy of the segmentation is "judged" by using a measure of efficiency of the process. For segmenting bubble images, many techniques have been attempted. Paul Symonds [21] has investigated conventional segmentation techniques and a morphological segmentation technique which uses a rolling ball algorithm. These are mentioned briefly in the next two paragraphs.

The automatic threshold selection technique for bubble highlight extraction and case studies of various edge detection techniques for bubble boundary extraction were investigated, and his results have shown that these techniques failed to give correct segmentation.

Further, the morphological processing of surface froth images using a discrete spherical structuring element (DSSE), or namely rolling ball algorithm, was presented. The results have shown that it could be a viable but not error-free segmentation technique. However, the program was iterative and therefore slow for any real time application. Therefore, it is not particularly suitable as a segmentation technique to be applied in practice.

Anoter segmentation method that must be mentioned here is called Computer Aided Bubble Segmenting. It is very similar to the computer vision approach, except that the stage of segmenting the image into individual bubbles is performed by a human. In this method, the human is presented with the bubble image on a computer screen, and must trace around each bubble using a drawing software package. This effectively results in the segmentation of the image into individual bubbles as perceived by this software operator. This method, however, takes far too long to be successful in real time applications. It must be stressed nevertheless, that at this point in time, where computer vision is still far from reaching the accuracy and ability of the human eye, this method should clearly provide the best possible results in this case. Therefore, in this research work, the accuracy of the segmentaion results was "judged" against and compared with

the results of human segmentation. As an example, the image BUB1, segmented manually is shown in Figure 3.4.

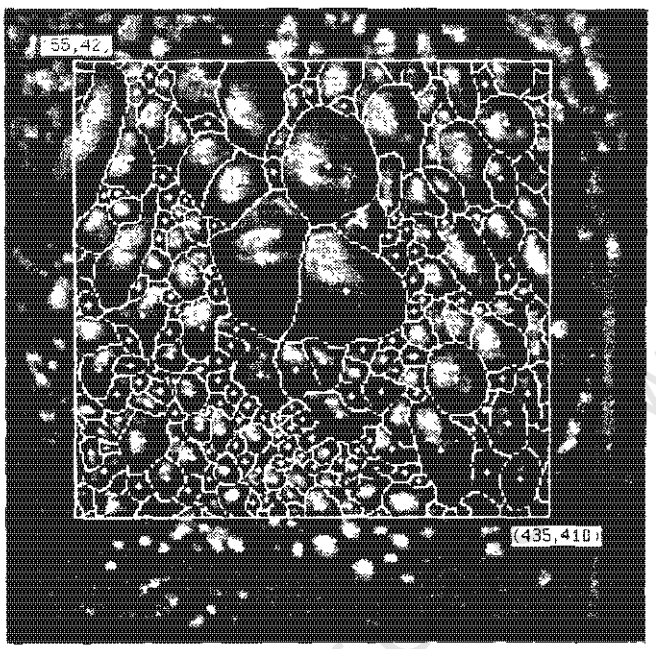


Figure 3.4: Manually segmented area of interest of the image BUB1 overlaid on the original image.

Chapter 4

Mathematical Morphology

In biology, the term **morphology** refers to the study of form and structure in both plants and animals; similarly, in imaging, **Mathematical morphology** refers to a branch of nonlinear image processing concerned with the analysis of shape [2]. This involves the extraction of image shape features and concentrates on the analysis of shapes of geometric structures such as edges, fillets, holes, corners, wedges, and cracks in the image. This is achieved by operating, or processing the image with various shape–structuring elements.

There are many areas where morphological methods have been successfully applied to image analysis. These include edge detection, texture analysis, particle analysis, feature extraction, shape representation and description, size distributions, and fractals. Mathematical morphology has been widely used for biomedical and electron microscopy image analysis, and it has been a valuable tool in many computer vision applications, especially in the area of automated visual inspection.

4.1 Morphological Processing

The general idea behind morphological processing is to probe or examine an image A with a **structuring element** B, which itself is a small image, and to quantify the manner in which the structuring element fits, or does not fit,

within the image. The two fundamental morphological operations on an image are defined as **erosion**, denoted $\ominus$, and **dilation**, denoted $\oplus$. These may be combined in sequences to produce other operations, such as **opening**, denoted $\circ$, and **closing**, denoted $\bullet$. The effects of these operations on a particular example image are illustrated in Figure 4.1 [12].

4.1.1 Morphological Erosion and Dilation

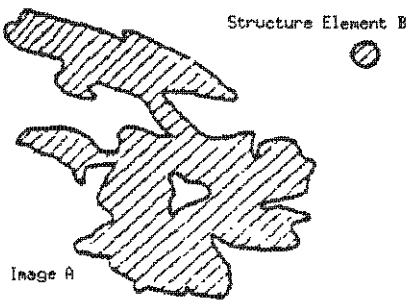
Erosion represents the act of probing an image to find out whether some primitive shape fits inside the image, and mathematical morphology depends on this notion. Dilation is the complementary operation to erosion, and is defined in terms of it, with the primitive shape fitting on the outside of the image perimeter [2].

Sets in mathematical morphology represent the shapes that are expressed on binary or gray scale images [11]. In the following discussion a set is understood to refer to a binary image. Geometrically, the erosion of A by B is the set of all points z such that the translate B_z is contained in the original set A. Also, the dilation of A by B is defined as the set of all points z such that B_z intersects A.

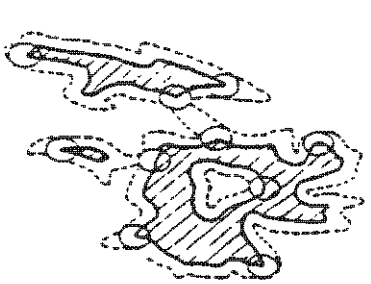
4.1.2 Binary Morphology

As mentioned before, binary images may be represent by sets. The set of all the black pixels in a black and white image (a binary image) constitutes a complete a description of this binary image. Sets in Euclidean 2–space denote foreground regions in such binary images — refer to Figure 2.1(a), where white (0) represents the background region and (1) represents the shaded foreground. The set of all white pixels and the set of all black pixels in a binary image may often be obtained by thresholding a gray–level image.

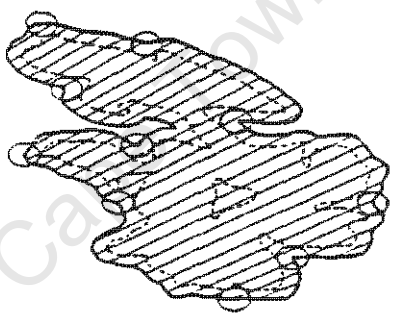
Figure 4.1 shows that erosion shrinks the set A, whereas dilation expands A as would be expected. Further, opening suppresses the sharp capes and cuts the narrow isthmuses of A, whereas closing fills in the thin gulfs and small lobes. Thus, if the structuring element B has a regular shape, both opening and closing can be thought of as nonlinear filters which smooth the contours of the input



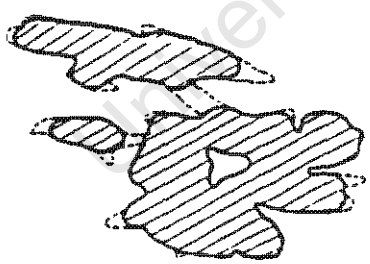
(a) Origin



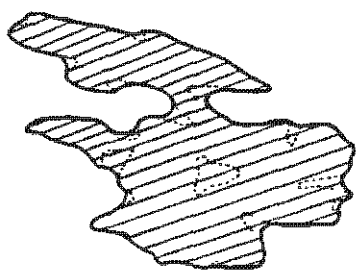
(b) Erosion



(c) Dilation



(d) Opening



(e) Closing

Figure 4.1: Erosion, dilation, opening and closing of A (binary image of geographical island) by a disk B centered at the origin. The shaded areas correspond to the interior of the sets, the dark solid curve to the boundary of the transformed sets, and the dashed curve to the bounday of the original set A (From reference [12]).

signal.

According to morphological theory, **binary erosion** is the morphological transformation that combines two sets by using containment as its basis set.

If A and B are sets in Euclidean N–space, then the erosion of A by B is the set of all elements z for which $z + b \in A$ for every $b \in B$. Refer to Maragos and Schafer[12] and Ming-Hua, Ping-Fan [1] for a more detailed treatment of this subject.

Definition 1: Let A be a subset in the space E^N and $B \in E^N$. The **erosion** of A by B is denoted by $A \ominus B$ and is defined as

$$A \ominus B = \{z \in E^N \mid z + b \in A, \text{ for every } b \in B\} \tag{4.1}$$

Erosion is an invariant translation and also an increasing operation. It is generally used for the shrinking or reducing of the original image. Erosion may be represented as an intersection of the negative translates:

$$A \ominus B = \bigcap_{b \in B} A_{-b} \tag{4.2}$$

Binary dilation was first used by Minkowski [6], and in mathematical literature it is referred to as the *Minkowski addition.* Dilation is a morphological transformation that combines two sets by vector addition of the elements of the sets [6].

Definition 2: Let A be a subset in the space E^N and $B \in E^N$ with elements $a = (a_1, \ldots, a_N)$ and $b = (b_1, \ldots, b_N)$ being N–tuples of element coordinates. The dilation of A by B is denoted by $A \oplus B$ and is defined as

$$A \oplus B = \{z \in E^N \mid z = a + b, \text{ for some } a \in A \text{ and } b \in B\} \tag{4.3}$$

Dilation is a commutative, associative, invariant translation, and also an increasing operation. It may also be used for noise removal, since it is analogous to the concept of low pass filtering. Dilation may be represented as a union of translates

of the structuring element:

$$A \oplus B = \bigcup_{a \in A} B_a \tag{4.4}$$

This is the dual operation to erosion.

From the above, it may be seen that the set resulting from the opening of A by B is in fact a set resulting from erosion of A by B followed by a Minkowski sum with B. Similarly, closing results from a Minkowski sum followed by erosion.

Definition 3: The opening of A by B is denoted by $A \circ B$ and is defined by

$$A \circ B = (A \ominus B) \oplus B \tag{4.5}$$

Opening is translation invariant and an increasing operation. It is the dual operation to closing. Opening an image with a disk–shaped structuring element smooths the contours, breaks narrow isthmuses, and eliminates islands and capes smaller in size or width than the disk structuring element.

Definition 4: The closing of A by B is denoted by $A \bullet B$ and is defined by

$$A \bullet B = (A \oplus B) \ominus B \tag{4.6}$$

Closing is an increasing, extensive, and idempotent operation. It is the dual operation to opening. Closing an image with a disk–shaped structuring element smooths the contours, fuses narrow breaks and long thin gulfs, eliminates holes smaller in size than the disk structuring element, and fills gaps on the contours.

4.1.3 Gray Scale Morphology

Gray scale morphology indicates that the morphological operators act on real–valued functions defined on N–dimensional Euclidean space, where, for signals $N = 1$, and for images $N = 2$ [2]. A gray scale image is an image in which each pixel has a specific value in a range larger than just 0 or 1. The binary morphological operations are naturally extended to gray scale imagery by the use

of a *min* or *max* operation [6]. From this it follows that the local *min* operator is equivalent to erosion and the local *max* operator is equivalent to dilation, subject to the condition that the relevant operator operates on the umbra of an image which is defined later, and in a local region.

The definitions and equations below show how the extensions of erosion and dilation to gray–scale morphology are made. Refer to Sternberg [19], Maragos and Schafer [12] for a more detailed discussion. The next paragraphs present a intuitive definition of a *top surface* of a set and its *umbra*. This is followed by a formal mathematical discussion.

Suppose we have a set A in Euclidian 3–space, where the first 2 coordinates of the 3–tuples of A make up the spatial domain of A, while the 3^{rd} makes up the surface. By definition, the *top surface* of A is the projection of A onto its spatial domain. Hence, the top surface at some point x is the highest value y subject to the condition that (x, y) is still an element of A (see Figure 4.2). One may thus define the *top surface* of A as $T[A](x) = \max\{y \mid (x, y) \in A\}$.

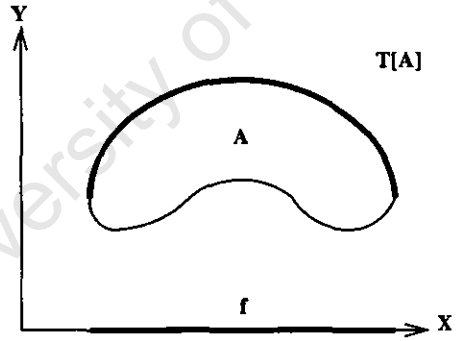


Figure 4.2: Concept of top or top surface of a set.

Further, the concept of an *umbra* may now be defined, in terms of $T[A]$, as being the set of all points lying *below* the top surface $T[A]$ (see Figure 4.3). Mathematically, the umbra $U[T[A]] = \{(x, a) \mid a \leq T[A](x)\}$.

A more formal approach is now presented, which is based on a given set A in Euclidian N–space [6]. Thus, let $A \subseteq E^N$ and let $B = \{x \in E^{N-1} \mid \text{for some } y \in$

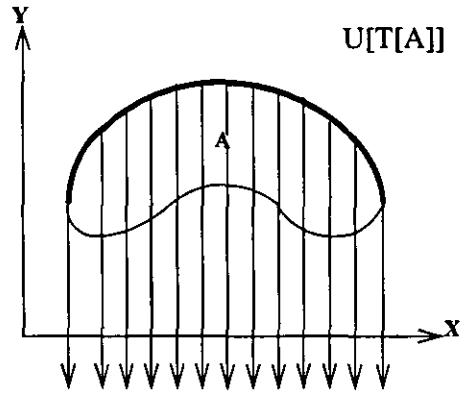


Figure 4.3: Umbra of the top surface of a set.

$E, (x, y) \in A\}$. The *top surface* of A, denoted $T[A] : B \to E$, is now defined by

$$T[A](x) = \max\{y \mid (x, y) \in A\} \tag{4.7}$$

Further, a set $C \subseteq E^{N-1} \times E$ is an *umbra* if $(x, y) \in C$ results in the fact that $(x, a) \in C$ for every $z \leq y$. Thus, for any function f defined on some subset F of Euclidian $(N-1)$ space, the umbra of f is a set consisting of f and every point below f. Hence, let $F \subseteq E^{N-1}$and $f : F \to E$. The *umbra* of f, now denoted by $U[f]$, $U[f] \subseteq F \times E$ is redefined to be

$$U[f] = \{(x, y) \in F \times E \mid y \leq f(x)\} \tag{4.8}$$

Note that y extends to $y = -\infty$, and that the top surface may be reconstructed from its umbra since

$$T[A](x) = \max\{y \mid (x, y) \in U[T[A]]\}, \ \forall x \tag{4.9}$$

Having defined the top surface and the umbra of a surface, **gray–scale dilation** of two surfaces may now be defined as the top surface of the dilation of their umbras. Let $F, G \subseteq E^{N-1}$ and let $f : F \to E$ and $g : G \to E$. The dilation of f

by g is denoted $f \oplus g$, where $f \oplus g : F \oplus G \rightarrow E$, and defined by

$$f \oplus g = T\{U[f] \oplus U[g]\} \tag{4.10}$$

The above conceptual definition is however not suitable for a hardware implementation. The following shows that gray scale dilation may in fact be calculated by finding the maximum of a set of sums. Taking f and g as per the above definition, $f \oplus g : F \oplus G \rightarrow E$ may be found by

$$(f \oplus g)(x) = \max\{f(x - z) + g(z) \mid z \in G, x - z \in F\} \tag{4.11}$$

The above equation is similar to convolution. However, it must be noted that in convolution we perform a summation of products, while here we take the maximums of sums.

In a similar manner to gray scale dilation, we now define **gray–scale erosion** of one surface by another as the top surface of the binary erosions of the umbra of one by the umbra of the other. Keeping the definitions of f and g, the erosion of f by g is denoted by $f \ominus g$, where $f \ominus g : F \ominus G \rightarrow E$, and defined by

$$f \ominus g = T\{U[f] \ominus U[g]\} \tag{4.12}$$

Similarly to gray scale erosion, the above is not suitable for a hardware implementation. Instead we take the minimum of a set of differences:

$$(f \ominus g)(x) = \min_{z \in G}\{f(x + z) - g(z)\} \tag{4.13}$$

Unlike correlation, we have replaced the summation of correlation by the minimum operation and the product of correlation by the subtraction operation.

As with the binary operators erosion and dilation, we could easily extend the above definitions to obtain the gray–scale morphology forms of opening and closing. Figure 4.4 [12] shows the top surface and the umbra of a 1-D signal. The effects of erosion, dilation, opening and closing on the signal are also shown.

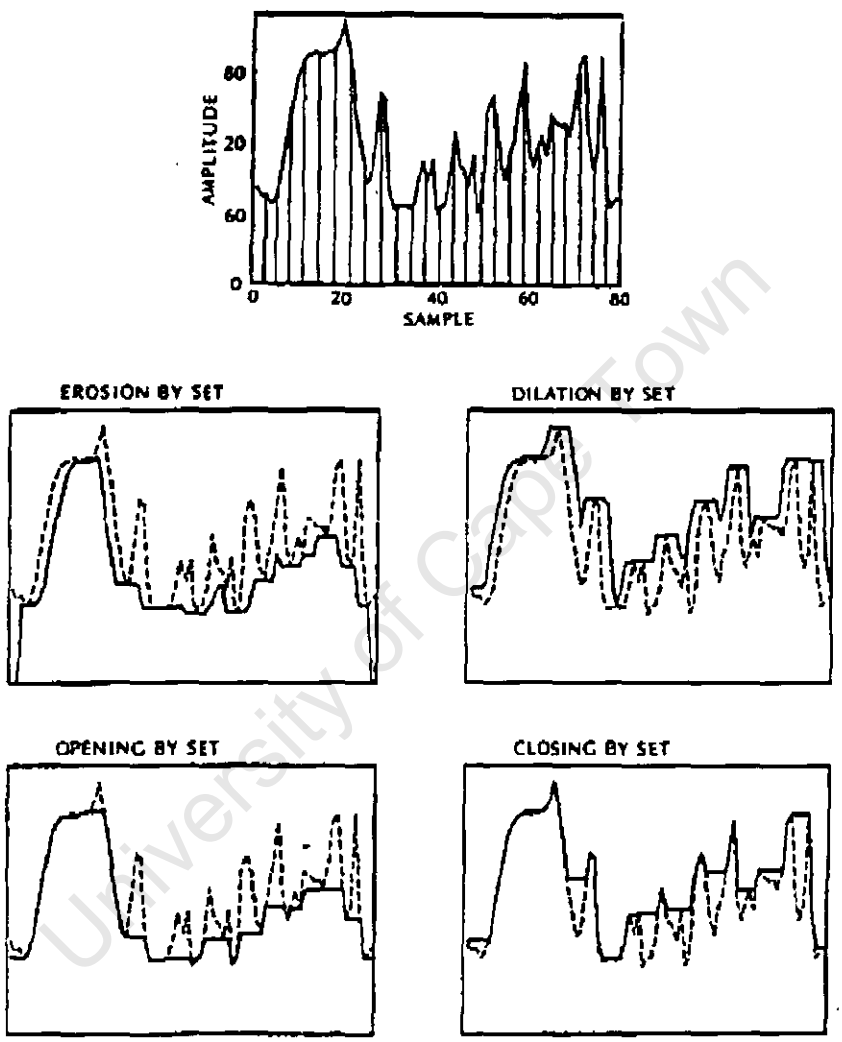


Figure 4.4: An illustration of the concept of morphological operation, using a 1-D signal. The shaded region is the umbra of the input signal, and the dashed curves in the lower four plots represent the input signal (From reference [12]).

Chapter 5

Watershed Methodology

In the field of morphological image processing, the act of segmentation by watersheds is one of the jewels. This general concept is defined in geography in terms of drainage patterns of rainfall. The same concept has been applied to the gray-scale segmentation in image processing. Take an image of a topographic surface of a catchment terrain, and suppose a drop of water is placed upon it (see Figure 5.1[2]). If placed on the right side of the vertical line, the drop will fall to the centre of the larger disk. Alternatively, if placed on the left side of the line, it will fall to the centre of the smaller disk.

For a given point in a catchment terrain, the points at which a drop of water would fall to that given point is called the **watershed** of that point. The lines

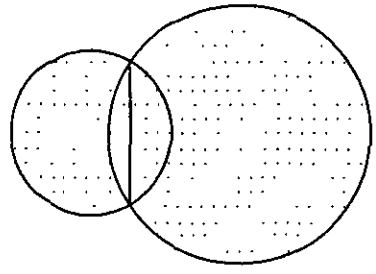


Figure 5.1: Segmentation of overlapping disks by watershed dividing line.

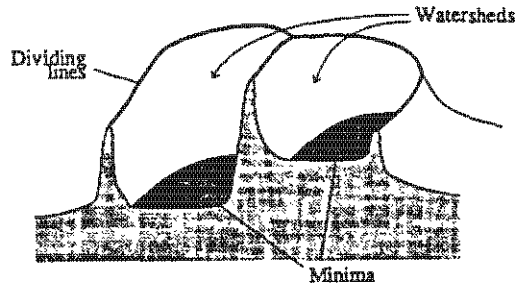


Figure 5.2: Watersheds

separating the different watersheds are called **dividing lines**. That implies that a watershed region only has one minimum area which is the bottom of the catchment terrain. These concepts are illustrated in Figure 5.2 [2]. The dividing lines of an image's morphological gradient provide the desired thin edge image for the original image. Both the **watershed regions** and **watershed boundaries** are important parameters in bubble image processing, and are discussed below.

5.1 Watershed Regions

A watershed region represents the catchment area of a minimum point. The set of points that make up a watershed region will be a subset of the boundary points of watersheds. That is, if a point does not belong to the watershed region set, it should at least belong to the watershed boundary set. Watershed regions of images can be calculated by using the local gradient and the intensity extrema of the image. The gradient information is then used to follow the image down to some intensity minimum.

5.2 Watershed Boundaries

Watershed boundaries are simply the lines separting the watershed regions. It depends on the multiresolution behaviour of the intensity extrema which define these regions, combined with other morphological tools.

5.3 The Watershed Algorithm

The methodology of watersheds has a number of variants (see [23]; [5]; [4]; [22]). The watershed boundary segmentation technique which was investigated here was introduced by John M. Gauch and Stephen M. Pizer [4] in 1992.

5.3.1 Identifying the Extrema of an Image

The algorithm begins by identifying the local intensity minima of an image which define the bottoms of watersheds. The image gradient could supply the direction information of the flow at any point. Those points in the image that tend toward the same minima are identified as belonging to the same watershed region of the image.

Since the characteristics of an integer–valued images are discrete, the surface of the image is rather uneven. Thus the input image needs to be converted to floating point and very slightly blurred. The purpose is to eliminate those sharp points in the image and simplify the process of identifying maxima and minima. To distinguish between these critical points, the gray value of each pixel is compared with its eight neighbours. If all neighbours have a value less than the central pixel, then that central pixel is identified as an intensity maximum. An intensity minimum is identified in a similar manner.

This above process is followed by calculating the image gradient. This is done so as to identify the drainage directions for each pixel in the image and thus finding all the intensity maxima (or minima) in the image. The identifier of this extremum is then labelled with a unique integer value. The eight neighbours

of each point are searched to determine the maximum and minimum intensity directions (which may or may not be in opposite directions due to discreteness). There are nine possibilities for each of these directions since the central pixel could be an extremum.

5.3.2 Identifying the Watershed Boundaries

By marking the locations of intensity minima with region identifiers in an output image, the partition of the input image into watershed regions is started. The marking is done by labelling the points with unique incrementing labels. For each of the remaining points in the image, follow the image downhill gradient to some intensity minimum. The identifier of this extremum is then recorded in the output pixel corresponding to this starting point. All of the other points along this gradient descent path are also labelled with the same intensity minimum. Once all pixels in the image have been associated with their respective minima, the output image will contain the watershed regions of the image. Similarly, following the image uphill, the regions for watershed duals can be calculated using a similar algorithm starting with the intensity maxima.

The position of watershed boundaries is finally determined by scanning the labelled image from left to right and then from top to bottom, detecting where region numbers change. These locations are then recorded in an output image.

5.3.3 Watershed Boundary Hierarchy

It is necessary to pay special attention to the watershed boundary hierarchy. This is the crux for achieving correct boundary positions. In all cases, it should be recognized that a particular methodology will often require pre–processing of the image, application of the algorithm, and post–processing to provide an acceptable output image.

For multi–scalar analysis, the structure of an image is simplified by smoothing with a series Gaussian convolutions. As blurring proceeds and thus some extrema are destroyed each time, store all of the positions of the intensity extrema for the

current scale and link them from one smoothing level to the next by following the image gradient descent. Then, the paths of linking intensity extrema of the image are set up. This information is then used to impose a hierarchy on the watershed regions.

Since the extrema of previous scales will annihilate into next new extrema positions, the watershed regions associated with the annihilated intensity extrema are said to be subregions of watershed regions that directly flow down from the annihilation points. It could be said that the relationship between these two watershed regions is that of parent and child.

The parent–child relationship can be described using a tree structure. The region associated with the final intensity extrema in the image is the root of this tree. The branches of the tree at each level can have an arbitrary number. The number of the parent extrema is used to determine the importance of each of the boundary curve segments.

5.3.4 Associating Scale with Watershed Boundaries

By continuing this process for all intensity extrema that define the bottom watershed regions, a hierarchy of watershed regions is defined. After that, the algorithm is used to determine the corresponding scale of watershed boundaries.

For example, take watershed boundaries as water barriers that disappear when adjacent watersheds annihilate into each other (see Figure 5.3 [4]).

The majority of the watershed boundary points in the image will be labelled by scale. But if two adjacent regions do not annihilate directly into each other, the hierarchy is searched to find the lowest scale watershed region that is a parent of both of these regions. The scale of the boundary between these regions is then determined to be the highest scale required for these two regions to annihilate into the parent region.

> "If region A annililates into region C at scale 5 and region B annihilates into C at scale 3, the scale of the boundary between A and

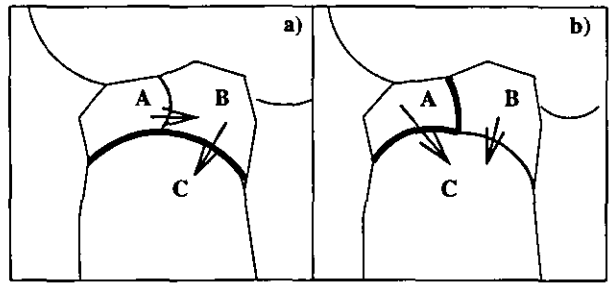


Figure 5.3: Associating scale with watershed boundaries. In case (a), region A annihilates into B before region B annihilates into C . In case (b), region B annihilates into C before region A annihilates into C.

> B is equal to 5. This corresponds to the lowest scale at which water originally in region A will mix with water from region B."

By continuing this process, the scale of all watershed boundary curves is calculated. Then, a representation of the parent–child relationships between these structures is built up.

5.3.5 Imposing a Resolution Hierarchy on Watershed Boundaries

According to definition, watershed boundaries are those curves that separate two regions. Also, each watershed boundary curve has two parents. Both endpoints of the boundary are connected to two curve segments that make up part of the boundary of adjacent watershed regions. When a curve segment annihilates, these pairs of adjacent watershed boundary curves merge, decreasing the number of curve segments making up the watershed boundaries of the image. There are four cases to consider, as can be seen in Figure 5.4 [4]:

- "In the simple case, when boundary segment C annihilates, the two curve segments A and B join to form a longer segment AB. At the same time, curves D and E are joined to form DE. Therefore, the number of local curves goes from 5 to 2. The curve

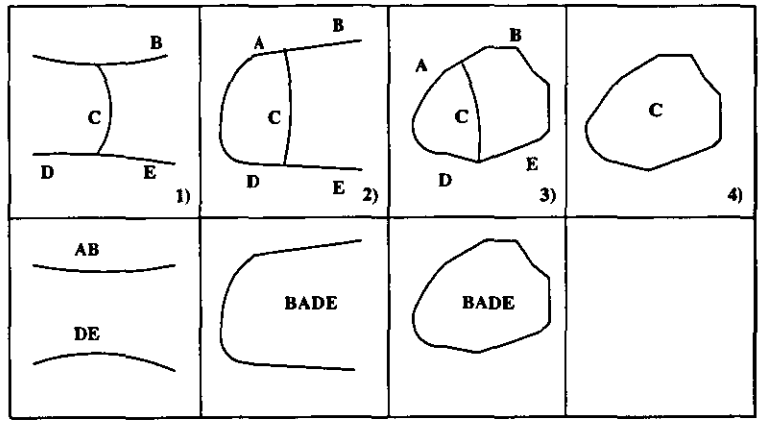


Figure 5.4: Four cases of watershed boundary annihilation.

segment C is defined to be a child of both boundary segments AB and DE.

- If curve A loops around and becomes curve D, then the removal of curve C will result in a single curve segment $BADE$. Similarly, if B loops around to become E, we obtain a single curve $ABED$ when C is removed. Here, the number of local boundary curves goes from 4 to 1. The curve segment C is defined to be a child of either the new boundary segment $BADE$ or $ABED$.
- If curve A loops around and becomes curve D and curve B loops around and becomes curve E, then when curve C is annihilated, the four neighboring curves join to form a single curve $ABED$. In this, the number of local curve segments goes from 3 to 1. Again, curve C is identified as a child of this single boundary segment.
- If the curve that annihilates is a closed loop that is not connected to any other watershed boundary curves, then we have no parent segment to associate with the child. In this case, the number of local curve segments goes from 1 to 0. Although this situation may sound unlikely, it will occur quite frequently in practice. Every time we have a small pit on the side of a ridge, the watershed boundary of the pit will be a closed curve." [4]

To sum up, an invalid watershed boundary is a curve that results from two curve segments that are connected such that the same watershed region is divided by this curve. Thus, such curves are ignored when the hierarchy of watershed boundary segments is built. A graph, with descriptions of curve segments in the nodes, is used to record the relationships between the watershed boundary curve segments. The parent–child relationships between the segments are reflected by edges between the nodes, such that operations which follow the graph "downwards" identify the descendants of a curve segment, and the converse, as can be expected from a hierarchial structure. All the boundary curves are recorded on an output image which can later be superimposed on the original image to display the result.

Chapter 6

Preprocessing for Watershed Segmentation

In order to achieve good results from the watershed segmentation technique, some preprocessing work has to be done on the images. This chapter discusses the preprocessing work involved and the reasons for it.

6.1 Inversion of the Image

As mentioned in Chapter 5, a basic condition of watershed segmentation theory is that each watershed region has its own intensity minimum area into which water would normally flow. In other words, each region is a *catchment basin* with an intensity minimum area as its bottom.

Taking Figure 3.1 as the reference image once again, it may be seen that the black areas of the image generally constitute bubble boundaries while the white areas represent the tops of the bubbles. In order to show the image in a form that corresponds to the theory of watershed methodology, the original image needs to be inverted. This results in the bubble boundaries now being white, and the bubble surfaces tending to black. The corresponding visual effect is that the bubble boundaries now stand out and are synonymous with watersheds, while

the bubble surfaces may be thought of as being catchment basins, as required by theory.

Therefore, the inverted form of Figure 3.1, as shown in Figure 6.1, was used as the direct test image for segmentation using the watershed methodology.

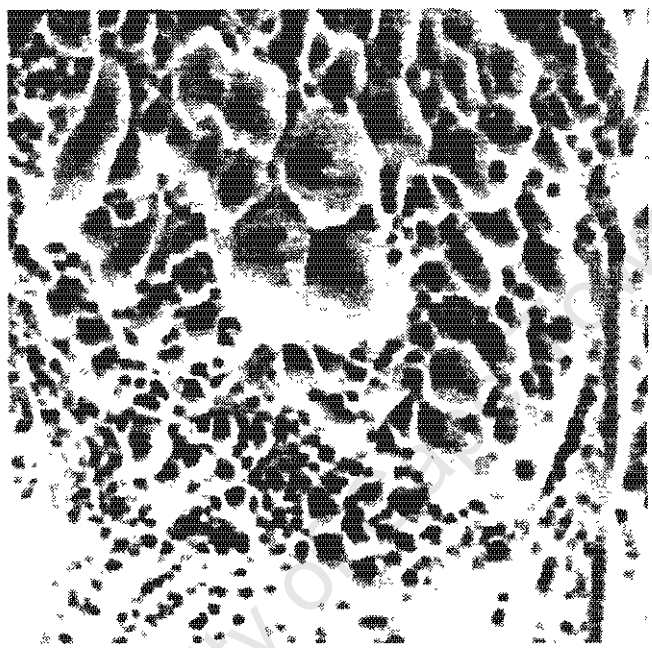


Figure 6.1: Inverted image of BUB1.

6.2 Further Preprocessing

Once image inversion has been performed, further stages of preprocessing may take place. The first stage is to smooth the image. This is because with an unsmoothed surface froth image, the result of watershed segmentation will show redundant segmentation lines, as is illustrated in Figure 6.2.

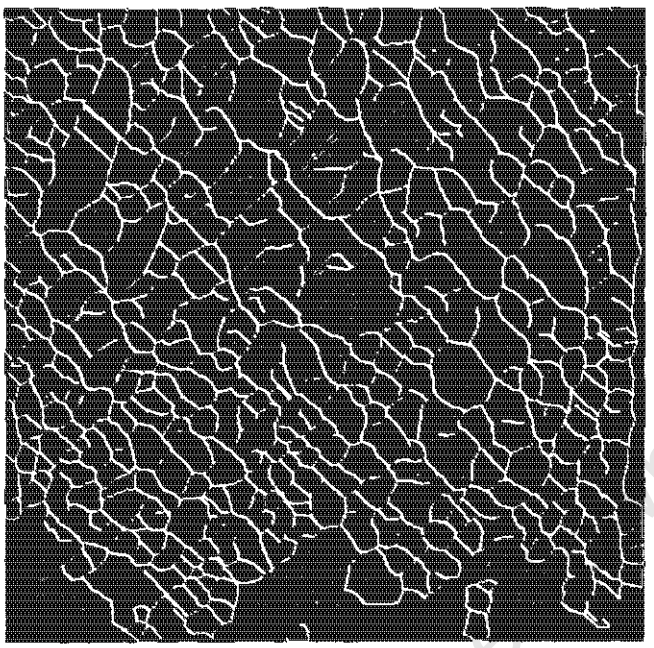


Figure 6.2: The watershed lines of the test image with no smoothing.

6.2.1 Smoothing of the Test Image

To blur an image, one should convolve it with a smoothing filter kernel. In mathematics, convolving an image I with a kernel k having support over domain K produces a convolved image, denoted by $I * k$, that is defined by [6]

$$(I * k)(x, y) = \sum_{(i,j) \in K} I(x - i, y - j)k(i, j) \tag{6.1}$$

Smoothing filters are designed to reduce the noise and detail in an image. Clearly, there are many filters that may be used to smooth an image, but the basic problem of smoothing filters is how to perform this without blurring out features of interest.

One of the advantages of this requirement to smooth the image before watershed processing is that it allows an interested party to easily investigate the effects of using all the different filtering techniques. As a first test, the smoothing of the

test image was performed by convolution with a 5×5 pixel square kernel. The result of watershed segmentation on the now smoothed image showed that this square kernel failed to achieve the desired effect since too much useful information was lost, as may be seen in Figure 6.3.

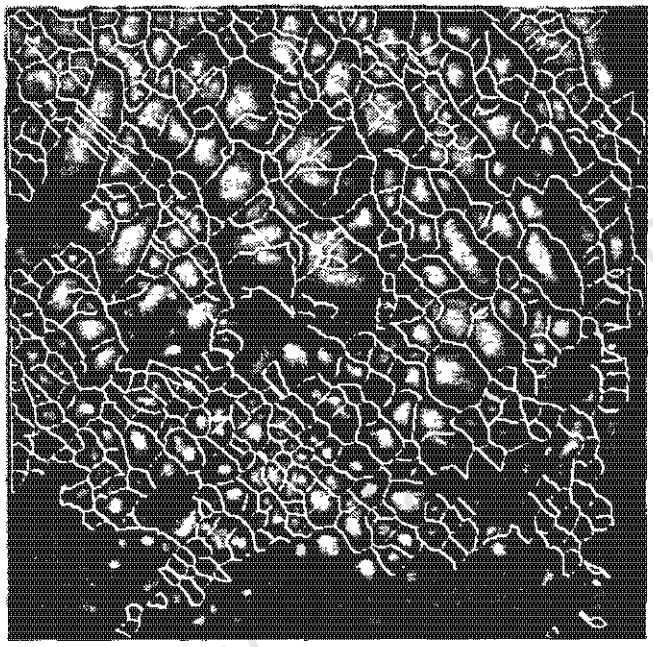


Figure 6.3: The original smoothed image (5×5 square kernel) overlaid with the watershed segmentation results.

After testing with numerous other filter types, it was decided that a Gaussian low pass filter gave the most pleasing results. The Gaussian filter and how it was used is now discussed.

A Gaussian filter is a linear spatial smoothing filter whose kernel is given by the two-dimensional Gaussian function $G_\sigma(x, y)$ of standard deviation σ as

$$G_\sigma(x, y) = \frac{1}{2\pi} e^{-\frac{1}{2}(\frac{x^2}{\sigma_x^2} + \frac{y^2}{\sigma_y^2})} \tag{6.2}$$

Figure 6.4 shows watershed segmentation results after blurring the test image with a Gaussian filter.

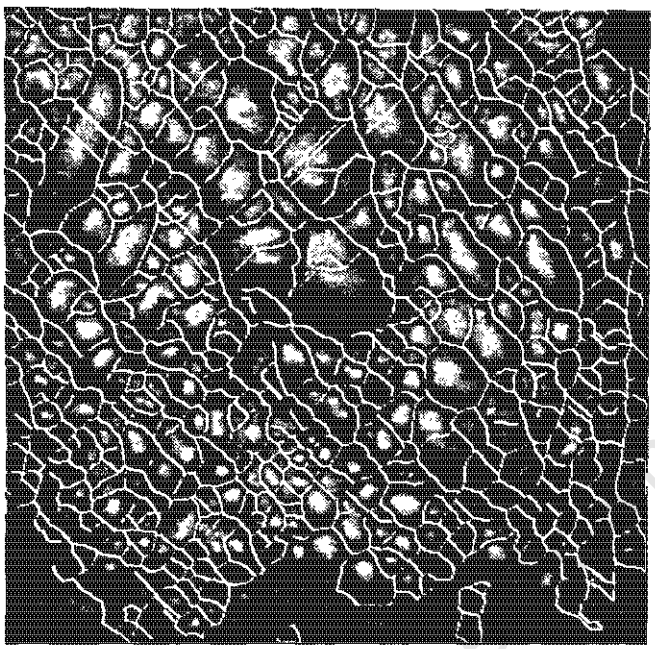


Figure 6.4: The results of watershed segmentation overlaid on the original image after smoothing with a Gaussian kernel.

However, one problem with Gaussian filtering of coordinate functions is that it results in a shrinking of the size of closed curves. The larger the curvature or the degree of smoothing, the greater is this amount of shrinkage. The source of this shrinkage arises from the fact that each point on a curve is being averaged with its neighbours, which in both directions curve towards the local centre of curvature [10]. When applying the watershed technique on the image smoothed by a Gaussian filter, the segmentation boundary lines were shifted slightly from the watershed boundary position. As a simple solution to this problem, a morphological dilation process was used before smoothing. In other words, the inverted image was dilated by a three dimensional spherical structuring element, which countered the shrinkage caused by Gaussian blurring.

Once the smoothing process has been completed, the watershed segmentation technique can be applied to the inverted surface froth image.

Chapter 7

Watershed Segmentation: Algorithm Design and Implementation

The basic theory of Watershed Methodology has been introduced in the early chapters of this dissertation. A detailed description of the application of the watershed theory to froth image processing is now given. The segmentation results are discussed in Chapter 8.

For the segmentation of surface froth images into separate bubbles, the watershed boundary technique was implemented by using the Khoros Image Processing Package running on a SUN workstation as the base platform. The relevant program modules were then linked together to perform the processing. A simple introduction to the Khoros Image Processing Package and the working software environment will be given at the end of this chapter.

7.1 Algorithm Design

First, the algorithm is described in general terms in point form. This gives an idea of the overall functionality of the segmentation method. After this, the

actual C functions are designed and explained in more detail. Finally, a flowchart illustrating the algorithm is presented.

7.1.1 General Algorithm Description

The algorithm may be described in the following generic terms:

1. Label the image minima - at the end of this, each unique minimum point or plateau area on the image will be labelled with a unique number. Note that all the points in a plateau will be labelled with the same value.

2. Store minima points - go through the labelled image and store the x and y positions of all the minima that were found. This information will be used later to determine which minima annihilate into other minima when the image is smoothed.

3. Label each point - for each point in the image, starting at that particular point, trace down along the gradient until a minimum is reached. Once such a minimum is reached, label all points in the down-gradient path with the same label as that at the found minimum. This will have the effect of labelling every point in the image with the label of the watershed region, or minimum, that it belongs to.

4. Now repeat the following steps for the number of hierarchy layers required by the user:

5. Smooth the image using a Gaussian filter. This will have the effect of annihilating some minima into other minima.

6. Label the image minima for this new smoothed image in the same way as in step 1.

7. Store minima points in the same way as in step 2.

8. Find parents - use the stored minima from the previous iteration and the current iteration to calculate which minima have been annihilated. The

parents of the annihilated minima will be the label values of the minima areas into which the previous minima annihilated.

9. If the number of iterations is equal to the number of hierarchy levels required by the user, then continue to step 10, else return to step 5.

10. Go through the originally labelled image horizontally. If the label of one point does not change to the next point, then there is no boundary and the value of the output image at that point must be zero.

11. If however there is a change in the labels, then this indicates the presence of a boundary. The gray-scale value of the boundary at this point in the output image must be related to the **strength** of the minimum that this point belongs to. That is, if the minimum that the point belongs to was annihilated early in the hierarchy levels, then the gray-scale value must be low, if it was annihilated late in the hierarchy tree, then the gray-scale value must be higher, and so on.

12. Go back to step 10 and repeat until the entire labelled image has been processed horizontally.

13. Now repeat steps 10, 11 and 12 but in the vertical direction.

7.1.2 Function Specifications

To implement the above algorithm, seven functions were provided. These are specified in basic terms below. Also refer to Figure 7.1 which shows the flowchart for the operation of the algorithm using these functions.

7.1.3 Function `lremodify()`

This is the main Khoros library function which coordinates the watershed segmentation process of the algorithm described before. It performs the following functions:

1. Check if the correct input image has been provided and whether an allowable number of hierarchy levels has been requested by the user.

2. Allocate memory for all required image arrays.

3. Label all the minima by calling the `Label_Extrema()` function.

4. Store the minima positions by calling the `Store_Points()` function.

5. Loop through every point in the labelled image. If the current point is not labelled, then call the `Label_Point()` function to perform the down-gradient tracing and labelling.

6. Iterate the following steps for the number of hierarchy layers requested by the user:

7. Smooth the image with a Gaussian filter.

8. Label the minima on this smoothed image by calling the `Label_Extrema()` function.

9. Store the minima positions by calling the `Store_Points()` function.

10. Determine which minima have annihilated into other minima by calling the `Find_Parents()` function.

11. If the number of iterations as requested by the user have been completed then continue to the next step, else return to step 7.

12. Loop through every point in the labelled image in the vertical direction. If the current point and the next point have the same labels then the value of this point on the output image is zero. Otherwise, if the labels change, thus indicating a boundary, the value of this point on the output image is related to how far in the hierarchy tree the minimum to which this point is connected is annihilated. The calculation of the actual gray-scale value is performed by calling the `Merge_Levels()` function.

13. Repeat the above step for the horizontal direction.

14. Free all dynamically allocated memory and exit.

7.1.4 Function `Label_Extrema()`

The purpose of this function is to label each minimum and plateau in the provided image with a unique number. This function may be summarised as follows:

1. Clear the image to which the labels must be written.
2. Loop through each point in the original image. If any of the eight points around the current point have a lower gray-scale value, then clearly this point is not a minimum. On the other hand, if all eight surrounding points' gray-scale value is greater than or equal to that of the current point then this point is a minimum or a plateau. This point must then be labelled with a unique number. Recall that we are processing an inverted image.
3. Clearly the above process will allocate different labels to points which belong to the same plateau. This must be changed so that all points which are part of one plateau have the same label. Loop through each point in the labels image. For each point call the `Relabel()` function which will label unique connected minima and plateaus with the same label value.
4. After the above process, the labels will be out of numerical succession order. Thus, loop again through all the points in the labels image and re-label them with the numerical values increasing from left to right and from top to bottom using the `Relabel()` function. This is useful for later error checking.

7.1.5 Function `Relabel()`

This function labels plateaus and connected minima with the same label value. Note that the value to be used for labelling must be passed to this function.

1. Set the current point label value to the provided value. A first glance shows that this has no useful function since the point already has that value. However, this is used for recursion purposes, as be seen from the description of step 2.

2. Loop through the eight possible directions. If the adjacent point has a different label, then that point's label must be changed to the label value of the current point. This adjacent point becomes the current point and recursion back to step 1 occurs.

3. On the other hand, if the adjacent point has no label but is of the same gray-scale value as the current point, then call the `Label_Point()` function to check for connected minima and then recurse back to step 1 to label the adjacent point the same as the current point.

7.1.6 Function `Store_Points()`

This function takes the labelled image and stores one single x,y position of a minimum or plateau.

1. Allocate memory for the `points` array where all these minima points are to be stored.

2. Loop though the labels image. When a new label occurs, store the x and y position where this change occurs.

7.1.7 Function `Label_Point()`

This function loops through the entire image looking for points that have not been labelled. If such a point is found, tracing down-gradient begins until a labelled minimum is reached. The point and the traced path is labelled with the label of the reached minimum.

1. If current point has a label, then return that label, else -

2. Calculate the gradient to the eight points around the current point.

3. Find the steepest downward gradient of the eight gradients found.

4. If there exists such a downward gradient then recurse back to step 1.

5. Otherwise, a labelled point must have been found since a mininum has been reached. Thus, return its value so that the function may reverse the recursion process, thus labelling each point along the traced path with the value of the minimum or plateau that was found.

7.1.8 Function `Find_Parents()`

This function takes the stored minima points for the current hierarchy level and the previous level as well as the newly labelled image and calculates which minima points in the previous level have annihilated into which minima in the current level.

1. Go through each stored minimum point in the previous minima points array. Call the `Label_Point()` function for each stored minimum point so that this point may be traced to the correct minimum in the new labelled image. The parent of this minimum point is then the label value of the minimum to which this point was traced in the new labelled image.

7.1.9 Function `Merge_Levels()`

Given the different labels of two adjacent points on the original labelled image this function finds at which level of the hierarchy tree the watershed regions represented by these two points merge. Thus, the farther down the tree the regions merge, the higher the gray-scale value given to the first point in the output image.

1. Begin at the bottom of the tree and count upwards.
2. If the two points have annihilated, then return the level at which this occurred.
3. Otherwise, the label values of the two points become the label values of their parents and execution recurses to step 1.

Finally, Figure 7.1 shows the basic flowchart of the operation of the `lremodify()` main program.

7.2 Algorithm Implementation

The above discussion of the algorithm and the basic specification of the functions involved is sufficient documentation for the understanding of the actual code, especially since the code does not contain any special features or difficult programming tricks.

However, there are three more points that must be explained in more detail.

7.2.1 Direction Definitions

Any point which is deemed to be the current point is thought of as being at the origin of the direction axes with the x,y coordinate of (0,0). The source code definition of the eight directions around this point is as follows:

```
int dx[8] = {  0,  1,  1,  1,  0, -1, -1, -1 };
int dy[8] = { -1, -1,  0,  1,  1,  1,  0, -1 };
```

This is shown graphically in Figure 7.2

7.2.2 Software Development Environment

The watershed segmentation program is written in the C language using the Khoros Image Processing Package running on a SUN system as mentioned in the beginning of this chapter. It should be helpful to give a simple introduction to the Khoros System.

Khoros is a software environment for research in the use of visual programming as a tool for software development for scientific visualisation. The Khoros System runs under XWindows (X11R4). The algorithm library of Khoros has been

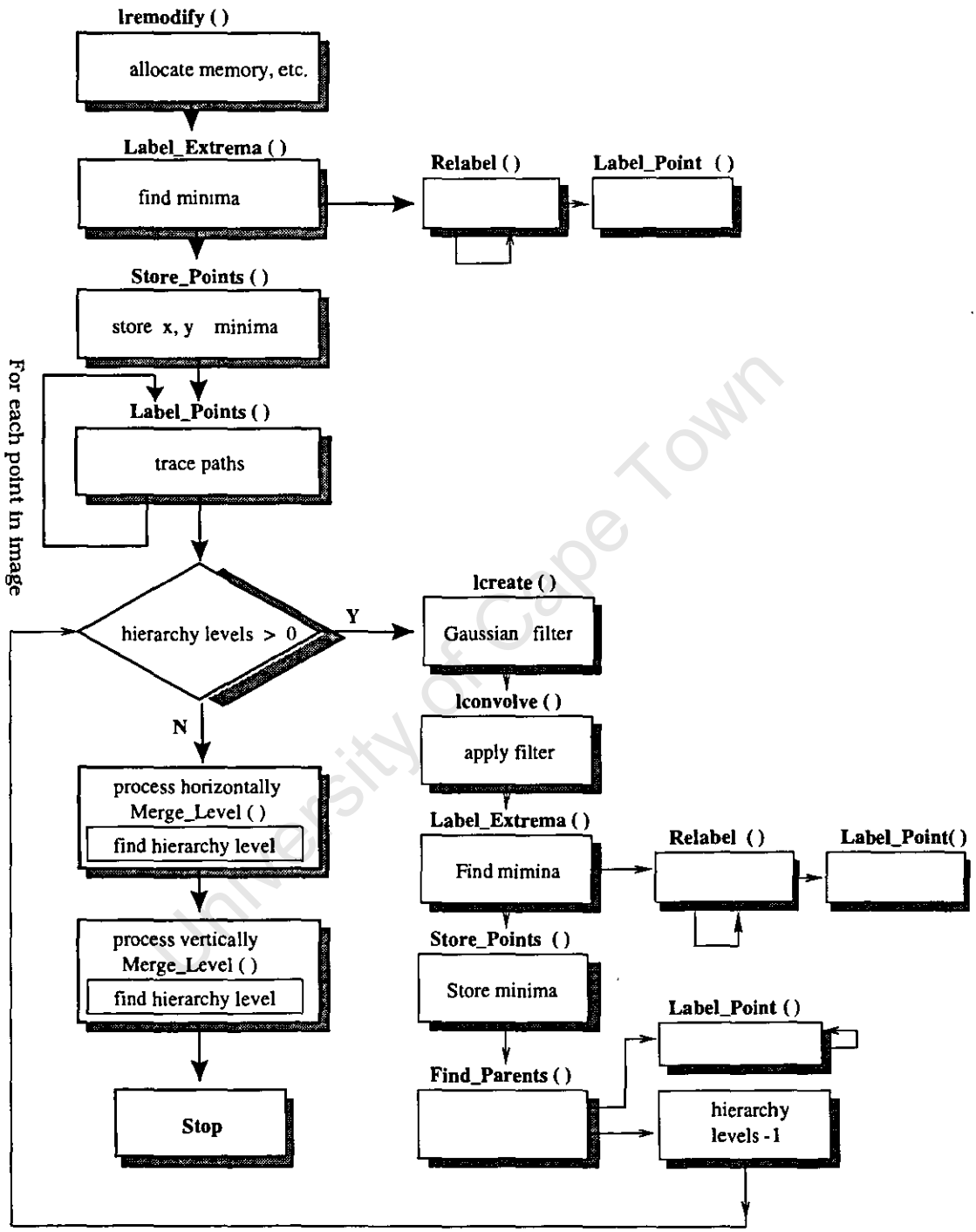


Figure 7.1: The basic flowchart of the operation of the `lremodify()`main function.

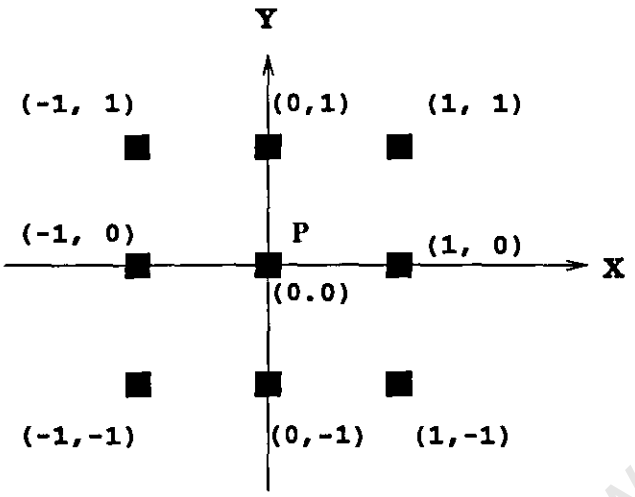


Figure 7.2: The Point P with its 8-neighbour pixel positions definitions.

developed to facilitate research in image processing, signal processing, pattern recognition, remote sensing, machine vision, and geographic information systems [8]. Khoros library software was written in the C language.

In a **cantata** workspace, as illustrated in Appendix A, the user selects the desired program blocks, places the corresponding glyph icons, and interconnects these elements to indicate the flow of processing from routine to routine. In other words, on **cantata**, a program is described as a path over which data tokens flow. For more detail about **cantata** please refer to Khoros Manual Volume I, Chapter 2 [8].

A user can interactively create and maintain a data processing (vroutine) program in the Khoros environment by using **composer** as a tool. The resulting programs will be compliant with the Khoros command line user interface as well as be compatible with the **cantata** visual language. All a user needs to do is create a new **pane** file and reference it from within the **cantata** user-interface system (UIS) form. The new **pane** file can be read into **cantata** via the workspace options. For more detail about **composer**, please refer to the Khoros Manual Volume II, Chapter 4 [8].

For the watershed segmentation program of this research work, the first task was to create a watershed **pane** file, recorded as `remodify.pane`, by using the **composer** tool. Then, according to the basic watershed principle and algorithm design which was introduced in this chapter, fill out the `remodify.c` and `lremodify.c` files which were created by using **Composer Programming Tools**.

The above discussion about Khoros may be difficult to understand since Khoros is an extremely powerful and correspondingly complex development tool. Unfortunately there is no place here for a more full and detatiled discussion of it. Such a discussion would also not be relevant to the purpose of this dissertation.

Chapter 8

Image Post-processing

The watershed segmentation result of a surface froth image, after application of the program described in Chapter 7, is shown in Figure 8.1 (recorded as IM1). A brief visual analysis of the image illustrates that the a majority of the bubbles have been correctly segmented. However, many over-segmented lines appear at the centres of some contours. Clearly those redundant lines must be eliminated. Thus, post-processing of the segmented image is necessary. Consequently, this chapter deals with the post-processing methods which perform these eliminations.

8.1 Deletion of Over-Segmented Lines

As mentioned, IM1 shows the image on which post-processing must be performed. Since the extra lines that may be seen were caused by different reasons and placed at different positions, the post-processing work must be separated into several steps:

- Bubble boundary contour lines have been labelled in different gray level numbers after watershed processing. The number of different gray levels depends upon the number of hierarchy levels, as defined in Chapter 7. Therefore, methods of "threshold" and logic "and" combinations were investigated, so as to exploit these differences in gray level for our benefit.

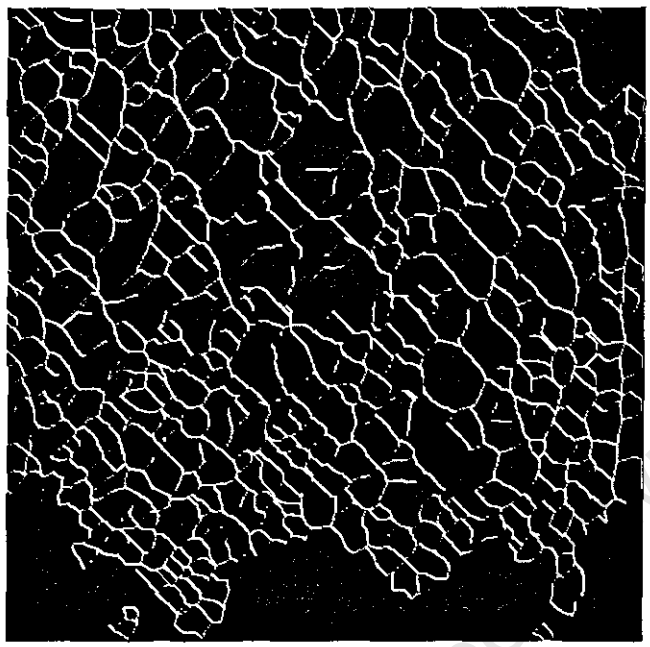


Figure 8.1: Bubble contours of BUB1 after watershed segmentation.

- Some of the extra lines cross the minimum areas of the image and may be deleted by using "threshold" and logic "and" methods as well. This is explained in more detail later in this chapter.
- For those extra lines that cannot be deleted by the above methods, the "Variance" method was investigated.

8.1.1 Threshold and Logical "And" Processing

The first step is to threshold the image IM1 (Figure 8.1) until the over-segmented lines disappear while the desired lines still remain. The thresholding was performed by using the inbuilt **cantata** function.

The act of thresholding removes most of the over-segmented lines but introduces another problem. When the over-segmented lines disappear, disconnections appear at the junctions of desired lines and the just deleted lines. Since these disconnections consist of one pixel "holes", the desired boundary lines are easily

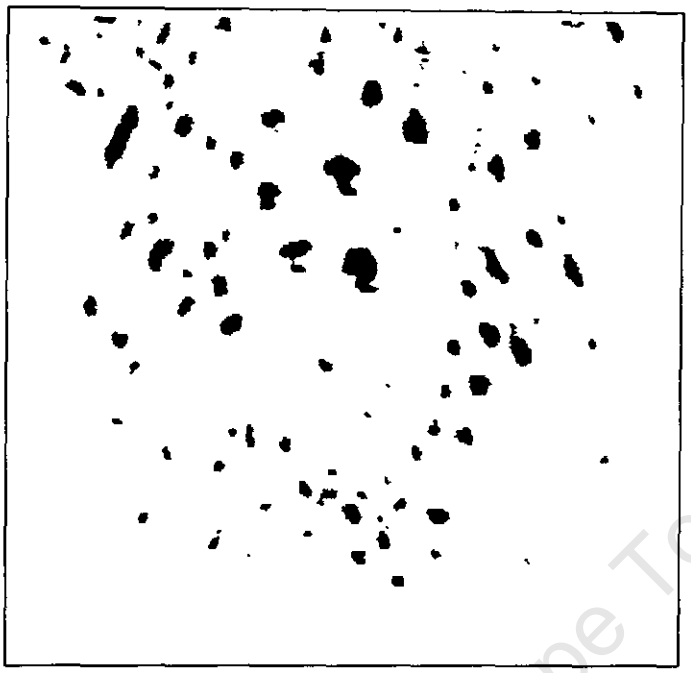


Figure 8.2: Minimum areas of the inverted image BUB1.

connected again by using morphological dilation. Therefore, dilation is the next step in this process. Then, the resulting dilated image is logically "and"ed with the image IM1. The output image is referred to as IM2.

For further reducing of undesirable lines of the image IM2, threshold and logical "and" methods were applied again. Refer to Figure 6.1 which is the inverted image of BUB1. To obtain only the minimum areas of this image, a suitable threshold value must be chosen. The resulting output image is shown in Figure 8.2 (referred as IM3). This is a binary image where pixels representing minimums have a value of 0, while the rest if the image has a value of 1. Since these black minimums occur in the bubble centres, undesirable lines which cross bubble centres on image IM2 may be removed by logically "and"ing IM2 and IM3. The resulting image is shown in Figure 8.3, and is referred to as IM4.

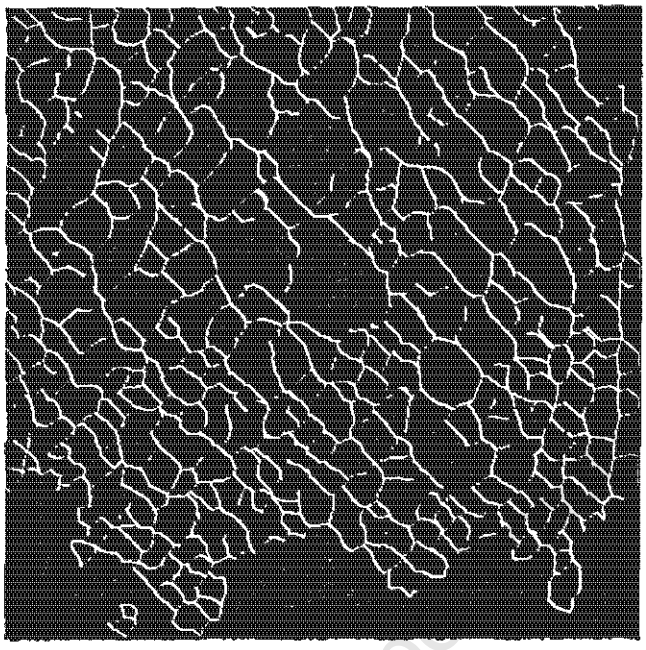
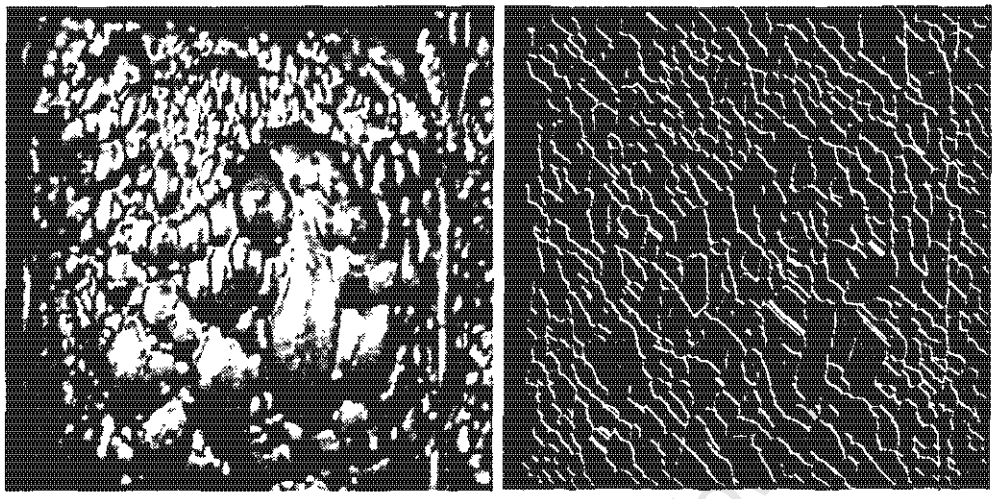


Figure 8.3: The segmented image of BUB1, after post-processings.

8.1.2 Variance Processing for the Reduction of Redundant Segmentation Lines

As a further effort to attempt to reduce redundant segmentation lines, "variance" methods were investigated. Unfortunately, the results illustrated that this method can only have a beneficial effect when processing big bubble regions.

In all of the above sections, the post-processing was limited to the processing of single pixels. That is, the above algorithms and methods were performed on a pixel by pixel basis. Now, the *variance* processing method is discussed as alternative post-processing way. This method differs from previous methods mentioned mainly in that it is not used to process the image on a pixel by pixel basis, but rather on a region by region basis.



(a) The original image Bubb1.

(b) Image Bubb1 after watershed segmentation.

Figure 8.4: The segmentation result of image Bubb1.

Variance Algorithm Design

The idea of using the *variance* algorithm came from the "Bubb" series of images (which was another set of images looked at, but not discussed in this thesis). In one of the images, Bubb1, after applying the watershed segmentation technique, there appeared many over-segmented lines at shadow regions of the large bubbles, as shown in Figure 8.4(b).

In trying to eliminate those extra lines, one characterisation of the image that could be used, is shadow regions. Because the bubbles, which show extra segmented lines in them are large, and the elongatedness value is high, the shadow region is big enough to be processed by using the *variance* algorithm.

The aim of the *variance* method is to readjust the gray scale distribution of the bubble images. That is, to change the gray level of the smooth (shadow) region near boundaries. After processing by *variance*, that region should be at a lower gray value; and the boundary area gray value should be higher before. Thereafter, combine "threshold" and logic "and" processing to eliminate those

extra lines which occur in the smooth regions.

Take Figure 8.4 (a) as the reference image. The image shows that in the large bubble contours there exist comparatively smooth regions. Hence, for the *variance* algorithm, choose a region which should be smaller than the detected existing smooth region first, then calculate the variance of the image. Thereafter, process the image by variance.

The variance, s^2, mathematical formula is given as follows:

$$s^2 = \sum_{i=1}^{k^2} \frac{(x_i - \overline{x})^2}{k^2} \tag{8.1}$$

where

k – the value of the chosen region;

x_i – the gray-value of pixel i;

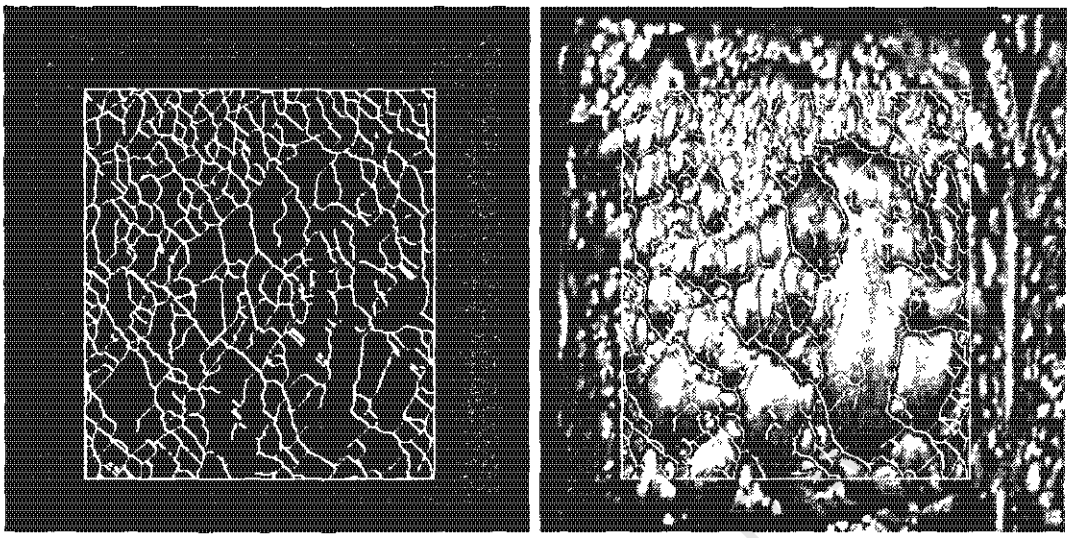
$\overline{x}$ – the mean gray-value of the k region, which in turn is given by

$$\overline{x} = \sum_{i=1}^{k^2} \frac{x_{ij}}{k^2} \tag{8.2}$$

where i and j are the x and y direction pixels respectively.

A summary of the algorithm processing steps according to the above variance equation is given below:

1. Set-up the initial image gray value as 255;
2. Choose the k region. For different images, there is a different k value, according to the shadow regions;
3. Loop along the x and y directions to calculate the sum of the k region pixel gray values. Then, calculate the mean gray value of the k region, that is $\overline{x}$;
4. Calculate the sum of the squares of deviations from the mean, that is $\sum (x_i - \overline{x})^2$;



(a) The watershed image of Bubb1 as processed by the variance algorithm.

(b) The watershed image of Bubb1 of part (a) overlaid on the original image.

Figure 8.5: The segmentation result of image Bubb1, after application of the variance method.

5. Calculate the *standard deviation*, that is the square root of their *variance*;
6. Trace through the whole image, removing those pixels which have lower value than *standard deviation*;
7. Readjust the image gray value distribution, that is, use the standard deviation of the image times the image, which is the standard variance image times initial gray value 255, then divided by the image which is the output of step 6.

After *variance* processing, the output image was thresholded, and then logic "and"ed with the watershed image. Figure 8.5 shown the results of variance processing on image Bubb1.

The results illustrate that the *variance* algorithm can give a good redundant line elimination effect when the image shows a big shadow region in one bubble, such as in Bubb1. In all other circumstances however, this method is not very effective, and was thus not pursued any further.

8.2 Thinning and Cleaning

The image of Figure 8.3 is almost completely segmented. However, it is still not ready for accurate bubble size analysis. This is due to some broken lines which appear in the image and also because the lines are not all of the same thickness. In this case, if one were to compute the bubble size and shape, the result would give erroneous data statistics. So, further post-processing is required. The required removal of broken lines and standardising of line thicknesses was performed by first thinning and then morphological closing.

8.2.1 Thinning

The thinning algorithm implemented here is discussed by Azriel Rosenfeld and Avinash C. Kak [18] (pg 232-240). The program was written in C under the Khoros Software development environment.

According to this algorithm, the deletion (thinning) processing does not locally disconnect boundary lines of segmented surface froth images. It is a specialised shrinking process which guarantees that the connectedness properties of the line do not change, even if all such points are deleted simultaneously. To prevent an already thin arc from shrinking at its ends, a further stipulate is that points having only one neighbour in an image are not deleted [18].

The thinning algorithm can be stated as follows: Delete all border points from a given side of a line, provided they are "simple" and not end points. Here, the border point P of line is called "simple" if the set of 8-neighbours of the P that lie in line has exactly one component that is adjacent to P [18].

For example, delete from north, was written in C as follows:

```
...
if (dir == N && d[N] == 0 )        /* North */
 if ( !(
        (   d[W]  && (!d[S])  &&   d[E]  ) ||
        ( (!d[W]) &&   d[NW]             ) ||
```

```
        (   d[NE] && (!d[E])              ) ||
        ( (!d[E]) &&   d[SE]  && (!d[S]) ) ||
        ( (!d[S]) &&   d[SW]  && (!d[W]) ) ||
       )
    )
    thin[x+y*width]=0
...
```

Where the N, W, S, E, NE, SE, SW, NW are constants that define the 8-neighbour directions of one point. Thus, checking from the north side of the line, if the point does not satisfy the five logical conditions in the 'if' statement, the point must be deleted. Do this successively from the north, south, east, west, north, sides of the line until no further changes take place from one iteration to another. Therefore, the result of thinning the image is a network of lines only one pixel wide without change of the image connectedness properties.

8.2.2 Deletion of Broken Lines by Morphological Closing

After the thinning process, the region labelling program is applied. (Region labelling is disscussed in Chapter 9.) On such a labelled image, the broken lines are easily deleted by using a morphological closing operator.

According to the morphological closing theory, which was described in Chapter 4, closing the labelled image with a structuring element which is one pixel in size and flat square in shape, will remove end points of line endings. After such multiple dilation and erosion processing, the broken lines in the labelled image are deleted successively. The results of cleaning (thinning and closing) of image IM4 are shown in Figure 8.6.

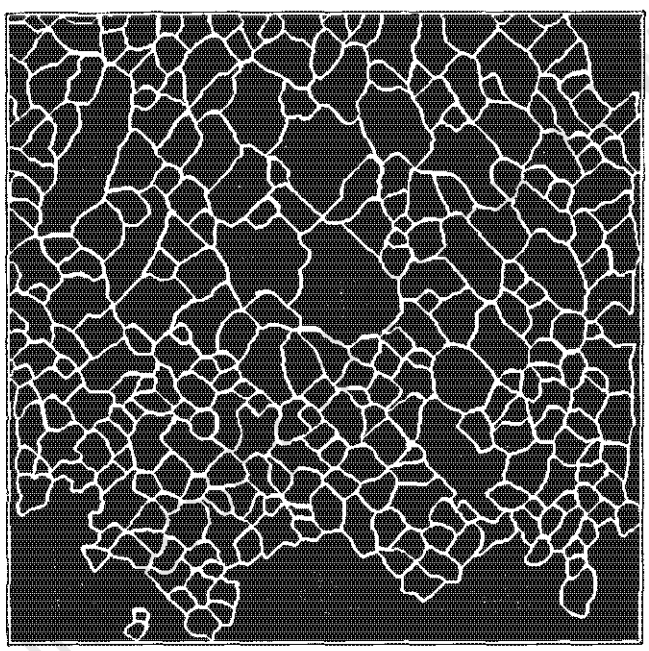


Figure 8.6: The watershed image of BUB1 after thinning and cleaning.

Chapter 9

The Results and Analysis of the Watershed Segmentation of Surface Froth Images

Clearly, the aim of watershed segmentation is to extract bubble boundaries from surface froth images. Once the isolated bubble regions are determined, the research work moves onto the analysis of the statistical distributions of the resulting bubble sizes and shapes. This is necessary since the bubble size and shape information can be used to quantitatively characterise the surface froth, and when combined with other chemical froth analysis results, the grade and recovery statistics of the froth drainage can then be controlled and optimised according to this information.

9.1 Preparation of the Segmented Images for Analysis

Before the measurement of any surface froth image properties can take place, each isolated bubble first needs to be uniquely labelled. This is required by the VSHAPE program, which is discussed later, so that it is able to keep track of

which bubbles it has analysed.

The labelling is performed by using the VLABEL program, which is integrated into the **Khoros Library**. The algorithm used by the program to label each bubble region individually may be summarised as follows:

"A pixel receives the same label as its neighbour if the likelihood distance between the two pixels is acceptable. The label process is propagated for a given region number until it is no longer possible to find a candidate [8]."

The likelihood distance in question may be computed using *Euclidean distance* or *city block distance*. In the case of labelling segmented surface froth images, the distance is computed using the former. The Euclidean distance between two points $p = (x, y)$ and $q = (u, v)$ is defined by

$$d_e(p, q) = \sqrt{(x - u)^2 + (y - v)^2} \tag{9.1}$$

Initially, all the pixels in the image are assumed to have a null label - that is, they do not belong to any region. The segmented surface froth image, $I(i, j)$, is scanned, starting with the top left-hand corner, and each pixel is examined in order. The image is thus scanned four times and the following functions are performed for each scan [8]:

- Scan 1 - The distance between the pixel to the next pixel is calculated and is stored in a new image called "*distance*".
- Scan 2 - The distance between two neighbours is analysed so as to fix the *threshold* which is used by the labelling routine to link pixels in the same regions together.
- Scan 3 - Each neighbour to the pixel is checked and linked to the pixel if the distance between them is less than the threshold.
- Scan 4 - All the uniform regions are labelled with a unique value.

Once the labelled image is produced, one may proceed onto the bubble size and shape statistical distribution analysis.

According to Azriel Rosenfeld and Avinash C. Kak [18], the **size** properties of a connected set S are determined by area and perimeter, while the **shape** properties are determined by complexity, elongatedness and circularity.

Since each region boundary corresponds directly to a bubble boundary, the area of the bubble in question is obtained by simply counting all pixels with the same label, i.e. in that region. Similarly, the perimeter is obtained by counting all the pixels, labelled with the same number, which constitute the region boundary. These calculations are automatically performed by the VSHAPE routine from the **Khoros Library**.

9.1.1 The VSHAPE Program

The VSHAPE program performs shape analysis on a labelled image (as discussed above) or a grey level image and extracts features like moments, areas, centroids, and the like [8]. These results are printed to an ASCII file by the program. Below is a sample layout of this resulting file, in the case of a labelled image:

The features listed below are computed for each region (i.e. bubble):

- Centroid on x and y axes.
- Variance on x and y axes.
- Area or weighted area.
- Orientation of the principal axis.
- Eccentricity of the shape.
- Bounded box co-ordinates.

The following features are optional:

- Standard moments.
- Central moments and normalised central moments.

- Invariant moments.

In addition to the ASCII file, an outline image can be generated - it contains the outlines of all the regions contained within the labelled image. For these images there is a final possibility, which is to use pseudo-random colour to assign a different colour to each region. This aids in the visualisation of the regions, and yields an artistic view of the image. Note that the principles behind the algorithm which is used by the VSHAPE program are taken from "Vision in Man and Machine" by Martin D. Levine [9].

9.1.2 Bubble Size and Shape Determination

Area

As mentioned before, to determine the area, A, of each region, the number of pixels in that region is counted. Consequently, the area is expressed in numbers of pixels.

Perimeter

Similarly, to determine the perimeter, P, for each region, that region's boundary is followed simply by distinguishing between two different label values. Thus, a count of all the pixels having a neighbour with a different label value yields the perimeter.

The bubble perimeters can be computed by running the VSHAPE program again on the outline image which was discussed earlier, since the outline image contains the outlines of every region in the image, and these outlines still retain the label numbers of the region whence they originate. Thus, the perimeter of each region is simply the area of the outline.

Circularity Measure

A measure of circularity, C, may be defined as follows [9]:

$$C = \frac{P^2}{4\pi A} \tag{9.2}$$

where P and A are the respective perimeter and area of a closed region.

As may be seen, C is a minimum when P and A are obtained from an ideal circle. Any distortion to the circle, assuming A is kept constant, will clearly increase P and thus increase C. Thus, the larger the value of C, the less circular a shape is.

However, under certain circumstances, a wedged circle, such as in Figure 9.1, might have an unduly large value of C. Thus, one must keep in mind that Equation 9.2 is not ideal, and thus this method alone is not sufficient to determine a shape's circularity to any degree of precision.

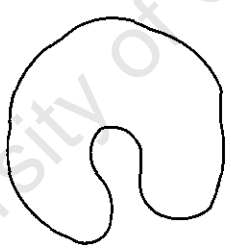


Figure 9.1: Example of a wedged circle.

Eccentricity Measure

Another, unrelated method for the quantification of the circularity or eccentricity of a shape is to use a methodology based on moments. Namely, a subset of *Scalar Transform Techniques* [9] will be used here.

The two-dimensional $(p+q)$th order moments are defined as

$$m_{pq} = \sum_i \sum_j i^p j^q I(i,j) \qquad p,q = 0,1,2,... \tag{9.3}$$

The physical interpretation of the moments can be given as follows [18]: If a grey level image I is regarded as being composed of point masses located at the points (i,j), then m_{00} is the total mass of I or, in other words, is the area A of I if the per unit mass is assumed to be unity. Further m_{02} and m_{20} are the moments of inertia of I around the i and j axes respectively.

The centre of gravity, or mean, of the pattern is then given by [9]

$$\bar{i} = \frac{m_{10}}{m_{00}} \qquad \bar{j} = \frac{m_{01}}{m_{00}} \tag{9.4}$$

The ($\bar{i}$, $\bar{j}$) point can be employed as a standard reference location of an object within an image. If the co-ordinate system is shifted so that its origin coincides with ($\bar{x}$, $\bar{y}$) then this translation normalisation results in a set of central moments μ_{pq} as follows,

$$\mu_{pq} = \sum_i \sum_j (i-\bar{i})^p (j-\bar{j})^q I(i,j) \tag{9.5}$$

Alternatively, it may be defined in terms of ordinary moments by means of the following equation [9]:

$$\mu_{pq} = \sum_{r=0}^{p} \sum_{s=0}^{q} C_r^p C_s^q (-\bar{i})^r (-\bar{j})^s m_{p-r,q-s} \tag{9.6}$$

where

$$C_r^p = \frac{p!}{r!(p-r)!} \tag{9.7}$$

Thus,

$$\mu_{00} = m_{00} = \mu \tag{9.8}$$

$$\mu_{10} = \mu_{01} = 0 \tag{9.9}$$

$$\mu_{20} = m_{20} - \mu\bar{i}^2 \tag{9.10}$$

$$\mu_{11} = m_{11} - \mu\bar{i}\bar{j} \tag{9.11}$$

$$\mu_{02} = m_{02} - \mu\bar{j}^2 \tag{9.12}$$

$$\mu_{30} = m_{30} - 3m_{20}\bar{i} + 2\mu\bar{i}^3 \tag{9.13}$$

$$\mu_{21} = m_{21} - m_{20}\bar{j} - 2m_{11}\bar{i} + 2\mu\bar{i}^2\bar{j} \tag{9.14}$$

$$\mu_{12} = m_{12} - m_{02}\bar{i} - 2m_{11}\bar{j} + 2\mu\bar{i}\bar{j}^2 \tag{9.15}$$

$$\mu_{03} = m_{03} - 3m_{02}\bar{j} + 2\mu\bar{j}^3 \tag{9.16}$$

From these central moments, the shape property attributes may be determined as is explained below.

The ellipticity or eccentricity, E, is an important global shape descriptor. It can be obtained by computing the ratio of one axis of the shape and the corresponding perpendicular axis, namely major axis I_a and minor axis I_b of an equivalent ellipse. This method was applied in the program VSHAPE for the measurement of eccentricity and is defined as follows [9]:

$$E = \frac{|I_a - I_b|}{I_a} \tag{9.17}$$

where

$$I_a = \frac{1}{2}(\mu_{20} + \mu_{02}) + \sqrt{\mu_{11}^2 + (\frac{\mu_{20} - \mu_{02}}{2})^2} \tag{9.18}$$

$$I_b = \frac{1}{2}(\mu_{20} + \mu_{02}) - \sqrt{\mu_{11}^2 + (\frac{\mu_{20} - \mu_{02}}{2})^2} \tag{9.19}$$

Generally, if the shape is circular or square-like, this ratio value tends to zero. One the other hand, if the E value is near unity, it represents a filamentary shape.

By combining measures of both circularity and eccentricity, the shape properties of an object may be quantified more reliably.

To summarise, Tables 9.1 and 9.2 list the parameters that have been discussed in this section. The first table lists the parameters that are useful for the quantising of bubble sizes, while the second table applies to bubble shapes.

Size	*Symbol*
area	A
perimeter	P

Table 9.1: Parameters related to bubble size.

Shape	*Symbol*
circularity	C
eccentricity	E

Table 9.2: Parameters related to bubble shape.

9.2 Area of Interest in Surface Froth Images

Before results and analyses on the parameters described in the previous section are given, it should be noted that the test bubble images which were processed during this research work and the manually segmented images which were used as standard reference images were chosen from the same sources as those of Paul J. Symonds [21]. Also, the areas of interest (AOI) processed are the same as those of Symonds. The reasons for this are listed below:

1. Since the purpose this research work and that of Symonds are the same, choosing the same set of test images is preferable so that processing and analysis results may easily be compared.

2. Even though human visual characteristics are different, for normal people such differences should be negligible. Thus the manually segmented images which were segmented by Symonds have also been used in this research work as the reference images. Of course, Symonds' images are not the standard, but may be taken as such for the purposes of this dissertation.

3. The same areas of interest of the images have been chosen so that comparisons of results with those of Symonds are viable, as mentioned in point one above. Further, bubbles which appear towards the edges of images are distorted by the conveyor belt on which the froth travels and thus do not represent true characteristics. Also, bubbles at image edges suffer from poor illumination, thus making analysis of gradients almost impossible. For these reasons, all bubbles on image edges have been ignored. The areas of interest concentrate on central regions of these images.

9.3 The Statistical Characterisation of Surface Froths

The most common form of graphical representation of experimental data distributions is the **histogram**. A histogram entails the breaking up of the data into

"classes" or categories, the determining of the number of observations in each class and the construction of a graph to display these frequencies. It depicts the frequency distribution using bars constructed so that the area of each bar is proportional to the number of observations in the respective category.

The results of measurements and calculations of bubble areas, perimeters, circularities and eccentricities are presented in histogram format first. Then, the histogram data will be fitted to appropriate statistical distributions. Further, a visual comparison of the expected (i.e. standard) and observed distributions will be presented and analysed.

9.3.1 The Plotting of Histograms

Since the purpose of the histograms here is to display the deviations from standard distributions, the data values should be divided into as many bins (i.e. classes) as is statistically meaningful. This is best explained by an example. If the bin sizes are too large, so that most of the data values are lumped into one bin, clearly no meaningful information can be extracted. On the other hand, if the bin sizes are too small, the data is divided into too many groups so that the overall effect is lost.

In practice, the size and shape distribution of surface froth bubbles is skewed, that is asymmetrical, with the right-hand tail being much longer - this is termed 'positively skewed'. In this case, the optimum bin width is an important factor.

Using test image BUB1 as a reference image, the bin widths were selected as follows:

Area The range of this parameter is, in our case, from about 1 pixel to about 7000 pixels. Fifty bins were chosen and thus each bin width is equal to 140 pixels. This encompasses a pixel area range from 0 to 7000.

Perimeter The range of this parameter is from about 1 to 400 pixels. Choosing 40 bins results in each bin width being equal to 10 pixels.

Circularity These values range from about 1 to 5. Thirty bins were chosen

giving bin widths of 0.167.

Eccentricity These values range from about 1 to 7. Twenty five bins were chosen giving bin widths of 0.28.

In order to find the above mentioned bin numbers and their sizes, the relevant data was plotted using **Mathcad v5.0 Plus**. The bins were adjusted until a satisfactory result was obtained. Comparisons were also made with the work of Symonds [21].

The measure of the asymmetry of a distribution, that is the *skew*, is given by [21]

$$Skew = \frac{1}{N}\sum_{i=1}^{N}[\frac{x_i - \overline{x}}{\sigma}]^3 \tag{9.20}$$

where $\overline{x}$ is the mean histogram bar height, σ is the corresponding standard deviation, N is the number of bins in the histogram and x_i is the i-th bin height.

The standard deviation, σ is given by

$$\sigma = \sqrt{\frac{\sum_{i=1}^{N}(x_i - \overline{x})^2}{N-1}} \tag{9.21}$$

and the mean value, $\overline{x}$, is given by

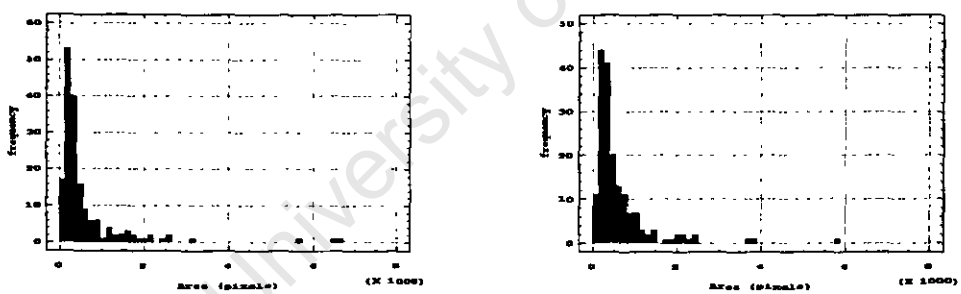
$$\overline{x} = \frac{\sum_{i=1}^{N} x_i}{N} \tag{9.22}$$

These parameters, that is *mean, standard deviation* and *skewness*, are those most frequently used to characterise a histogram distribution [21].

Finally, note that the x-axis value at which the highest bar in the histogram occurs is termed the *mode*. In other words, it is the value of x for which a discrete distribution $f(x)$ is maximum.

9.3.2 Histograms of Bubble Shape and Size Distributions

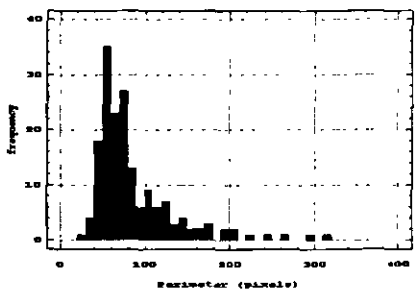
The information produced by the VSHAPE program was further processed using **Mathcad V5.0 Plus** and **Statgraphics 6.0**. The area, perimeter, circularity and eccentricity histogram distribution graphs are given in Figures 9.2, 9.3, 9.4 and 9.5 respectively. The legend for each graph gives the mean ($\overline{x}$), standard deviation (σ) and skewness values. Further, Figures 9.6, 9.7, 9.8 and 9.9 give graphs, in the same order as above, firstly where the expected and the observed distributions are overlaid and secondly where the expected and the observed cumulative distributions are overlaid.



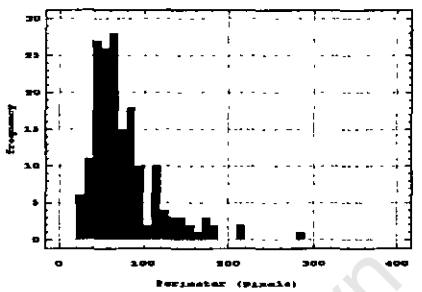
(a) Expected distribution with $\overline{x}$ = 625.16, σ = 934.23, skewness = 4.37.

(b) Observed distribution with $\overline{x}$ = 618.09 σ = 713.15, skewness = 3.71.

Figure 9.2: Bubble area distributions for the test image BUB1.

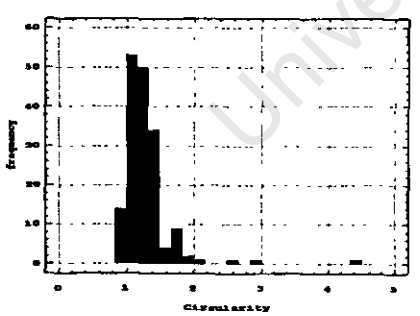


(a) Expected distribution with $\overline{x} = 85$, $\sigma = 59.55$, skewness = 2.29.

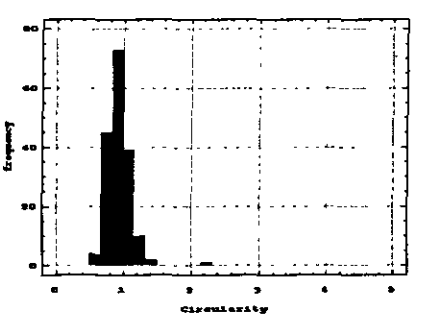


(b) Observed distribution with $\overline{x} = 86$, $\sigma = 47.00$, skewness = 1.41.

Figure 9.3: Bubble perimeter distributions for the test image BUB1.

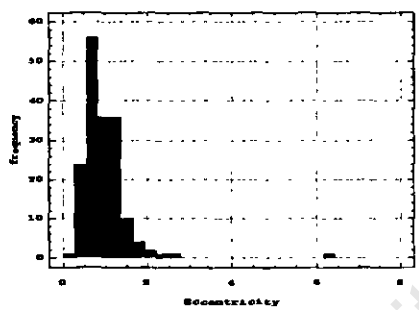


(a) Expected distribution with $\overline{x} = 1.32$, $\sigma = 0.57$, skewness = 5.36.

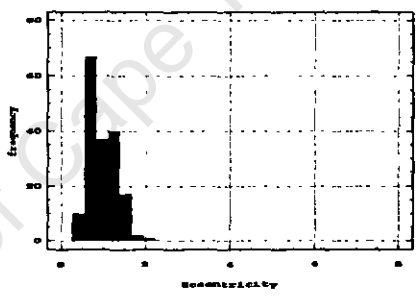


(b) Observed distribution with $\overline{x} = 0.93$, $\sigma = 0.18$, skewness = 2.60.

Figure 9.4: Bubble circularity distributions for the test image BUB1.

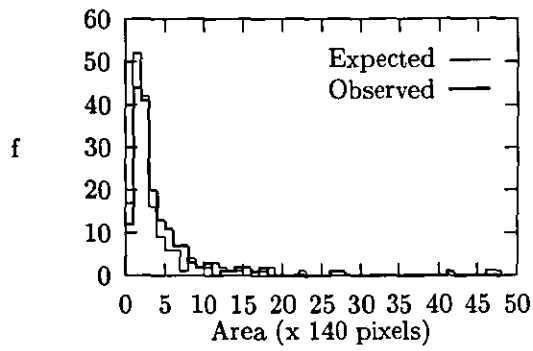


(a) Expected distribution with $\overline{x} = 0.83$, $\sigma = 0.37$, skewness $= 0.97$.

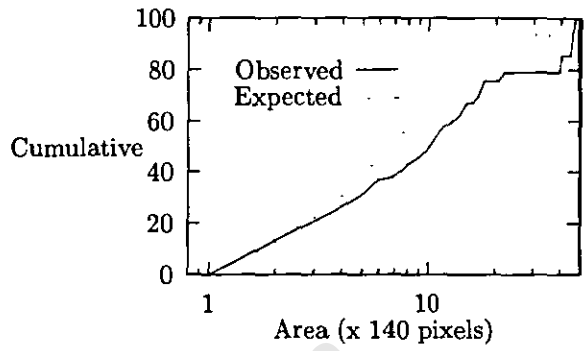


(b) Observed distribution with $\overline{x} = 0.93$, $\sigma = 0.49$, skewness $= 5.41$.

Figure 9.5: Bubble eccentricity distributions for the test image BUB1.

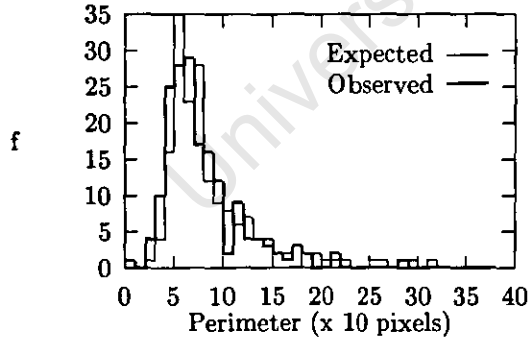


(a) Overlaid graphs of observed and expected area distributions.

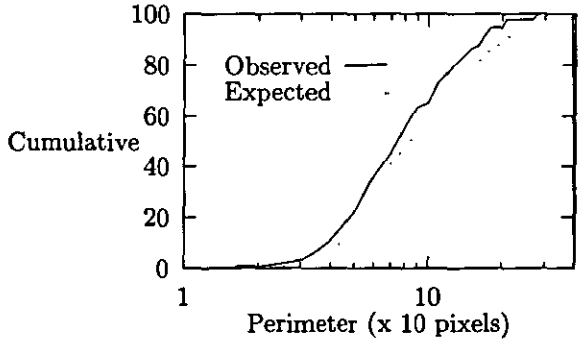


(b) Overlaid graphs of observed and expected cumulative area distributions.

Figure 9.6: Overlaid distributions and cumulative distributions of area for test image BUB1.

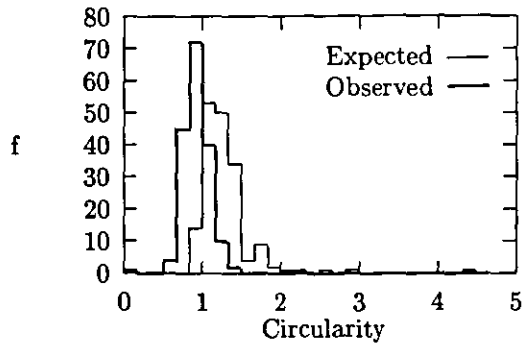


(a) Overlaid graphs of observed and expected perimeter distributions.

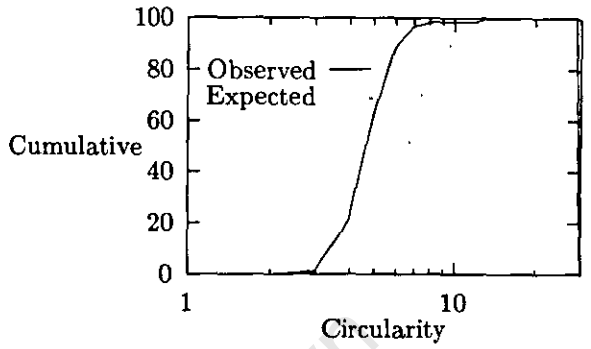


(b) Overlaid graphs of observed and expected cumulative perimeter distributions.

Figure 9.7: Overlaid distributions and cumulative distributions of perimeter for test image BUB1.

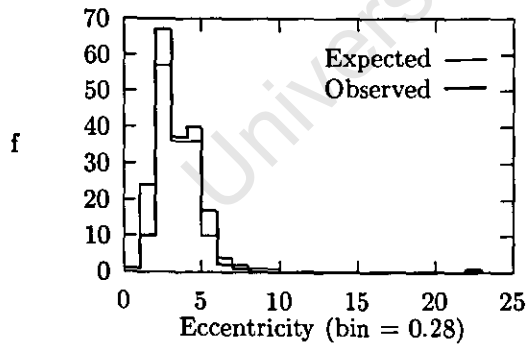


(a) Overlaid graphs of observed and expected circularity distributions.

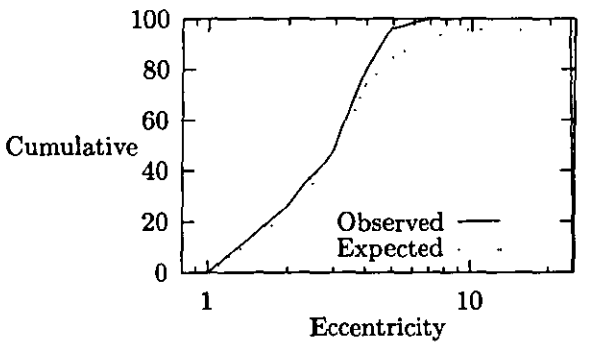


(b) Overlaid graphs of observed and expected cumulative circularity distributions.

Figure 9.8: Overlaid distributions and cumulative distributions of circularity for test image BUB1.



(a) Overlaid graphs of observed and expected eccentricity distributions.



(b) Overlaid graphs of observed and expected cumulative eccentricity distributions.

Figure 9.9: Overlaid distributions and cumulative distributions of eccentricity for test image BUB1.

9.3.3 Analysis of the Histograms

The histograms in the previous section offer an excellent visual representation of the image processing results. However it is not enough to obtain quantitative statistical measures and thus further analysis is required. That is, one must examine how well the observed distributions fit the expected distributions. For this purpose, the *chi-square (χ^2) goodness of fit test* will be used. This method is used extensively in the field of applied statistics and is defined as follows [13]:

$$\chi^2 = \sum_{i=1}^{k} \frac{(O_i - E_i)^2}{E_i} = \sum_{i=1}^{k} \frac{[(\text{observed freq.}) - (\text{expected freq.})]^2}{\text{expected freq.}} \tag{9.23}$$

with $k - 1$ degrees of freedom,

where k is the number of categories (i.e. bins). Milton [13] suggests that k be larger or equal to at least five for meaningful results. In other words, E_i and O_i should consist of five or more categories, or bins, each.

For judging of the results, the concept of **hypothesis** is given below:

H_0 : Null hypothesis - this represents the fact that the observed distribution roughly fits the expected distribution. In other words, that the O_i and E_i distributions are similar to each other.

H_1 : Alternate hypothesis - this represents the fact that the two distributions are *far* different from each other - that is, the observed distribution does not fit expected distribution to any reasonable degree.

This naturally brings up a question with regard to the definition placed upon the word *far* - "By how *far* must the two distributions differ to cause the H_0 to be rejected, and how those differences can be measured ?"

From the chi-squared equation (Eq. 9.23), it may be seen that if H_1 is taken to be true, then χ^2 will tend to take on large values. Therefore, for H_0 to be rejected, a critical value of χ^2 should be determined and used. That is, H_0 is to be rejected when

$$\chi^2 \geq \text{critical value} \tag{9.24}$$

This critical value is usually obtained from the "upper-tail probabilities for the chi-squared (χ^2) distribution table" where the significance level (α) is taken as 0.05 (or 5%) and the appropriate number of degrees of freedom is (k-1) [17].

Applying the Chi-Squared Test to the Bubble Size and Shape Distributions

For segmented surface froth structures, the expected frequency, E_i, is obtained from the area of interest (AOI) of the manually segmented image and the observed frequency, O_i, is obtained from the AOI of the computer segmented image. Note that obviously the total number of segmented regions will not be the same for the manually segmented image and the computer segmented image, and thus one cannot apply the chi-square test directly. The expected E_i values must thus be modified. These modifications are given by Symonds [21] as

$$E_i = (\frac{N_O}{N_E}) \cdot e_i \tag{9.25}$$

where N_O is the number of bubble regions segmented by computer; N_E is the number of bubble regions segmented manually; and e_i is the number of bubble regions falling into bin i.

This modified E_i value will now be used in the χ^2 test. Note that in the test image BUB1, $N_O = 174$ and $N_E = 172$. The χ^2 values for the area, perimeter, circularity and eccentricity distributions are shown in Tables 9.3, 9.4, 9.5 and 9.6 respectively. These results are analysed thereafter.

Interval	O_i	E_i	o_i	χ^2
1 - 140	11	17	10.78	2.27
140 - 280	44	52	43.47	1.71
280 - 420	41	40	40.18	0
420 - 560	20	16	19.6	0.81
560 - 700	13	9	12.74	1.55
700 - 840	11	6	10.78	3.80
840 - 980	7	6	6.86	0.12
980 - 7000	27	25	26.46	0
Total	174	172	172	10.26

Table 9.3: χ^2 values for the **Area** distribution.

Interval	O_i	e_i	E_i	χ^2
1 - 50	10	15	11.46	0.19
51 - 60	11	18	13.75	0.55
61 - 70	21	18	13.75	3.82
71 - 80	14	23	17.57	0.73
81 - 90	10	12	9.17	0.08
91 - 100	12	13	9.80	0.49
101 - 120	11	22	16.59	1.88
121 - 140	17	14	10.56	3.93
141 - 400	17	26	19.60	0.34
Total	123	161	122.25	12.01

Table 9.4: χ^2 values for the **Perimeter** distribution.

Interval	O_i	E_i	o_i	χ^2
0 - 0.625	20	0	19.6	/
0.626 - 0.75	53	0	51.94	/
0.76 - 0.875	49	14	48.02	1.6
0.876 - 1.000	34	41	33.32	23.66
1.001 - 1.125	11	33	10.78	1.02
1.126 - 1.250	4	33	3.92	0
1.251 - 1.375	2	22	1.96	7.15
1.501 - 5.000	1	21	0.98	29.14
Total	174	172	170.92	62.57

Table 9.5: χ^2 values for the **Circularity** distribution.

Interval	O_i	E_i	o_i	χ^2
0 - 0.28	10	25	9.88	9.14
0.29 - 0.56	67	56	66.19	1.85
0.57 - 0.84	37	36	36.55	0.0
0.85 - 1.12	40	36	39.22	0.28
1.13 - 1.40	17	10	16.79	4.16
1.41 - 1.68	2	4	1.96	1.04
1.69 - 7.00	1	5	0.98	3.23
Total	174	172	171.58	20.15

Table 9.6: χ^2 values for the **Eccentricity** distribution.

Analyses of χ^2 Results

Area: Refer here to Table 9.3. The calculated total χ^2 value is 10.26 for 7 degrees of freedom (i.e. eight bins). The critical value $\chi^2_{7,0.05}$ (the 7 refers to the degrees of freedom while the 0.05 refers to the chosen significance level) is 14.06. Thus, H_0 is accepted.

Even though H_0 is accepted, the expected and observed area distributions are not the same. This may be explained with the aid of Figure 9.6(a) which shows that the main differences between the two distribution graphs occur in regions with small areas with values less than 280 pixels. These differences are caused by the pre-processing work before watershed segmentation, as described in Chapter 7. In order to avoid extraneous segmentation lines, the images were smoothed by a Gaussian filter. After such smoothing, clearly some of the small area regions were lost, thus causing the differences between the two distributions. This loss can be controlled by adjusting the Gaussian filter's σ value, depending on the general size of the bubbles in which one is most interested.

Note that the cumulative distribution graph of Figure 9.6(b) contradicts the above discussion by showing main differences in corresponding to large bubbles. However, this is caused by the actual data representation used, where the differences in small area regions only affect the graph later, in large area regions.

Perimeter: Refer here to Table 9.4. The calculated χ^2_8 value is 12.01. The critical value $\chi^2_{8,0.05}$ is 15.51. H_0 is thus accepted.

Circularity: Figures 9.8(a) and (b), show an interesting result in that the distributions look similar but the observed distribution is displaced along the x-axis. This shift may be explained by the fact that the value for circularity was defined in terms of the area and perimeter of the segmented bubbles. The results show that the observed circularity values tend to be lower than 1 while the expected circularity values tend to be greater than 1. Clearly, the act of smoothing the image by the Gaussian filter will in effect smooth many edges and thus result in a lower perimeter value for an almost unchanged area which in turn results in a lower value for circularity. On the other hand, the manually segmented bubble boundaries show more "detail", or "noise", which results in a longer perimeter value, therefore increasing circularity values. It is also suspected that a human being, with his/her three dimensional vision and intelligence is able to successfully differentiate shadow from bubble border even on a two dimensional image. The computer algorithms however, have no such abilities and will treat a dark shadow and a dark bubble border in the same manner. Shadows will often

emphasise the circularity of an object and this might be one of the reasons why the circularity values were so low.

Because of the abovementioned shift phenomenon, it is meaningless to apply the χ^2 test directly. To remedy this, the O_i value was shifted down two bins to fit the E_i value more closely, as shown in Table 9.5. The resulting total χ^2_6 value is 62.57. The critical value $\chi^2_{6,0.05}$ is 12.59. H_0 is thus rejected.

Eccentricity: Refer here to Table 9.6. The calculated χ^2_8 value is 20.15. The critical value $\chi^2_{6,0.05}$ is 12.59. H_0 is thus rejected.

This measure of bubble circularity indicates that in fact the observed bubbles are less circular than the expected bubbles. This contradicts the discussion under subsection "circularity" above. One may summarise that both eccentricity and circularity measures are required to form some kind of meaningful conclusion. It must also be pointed out that Symonds [21] obtained similar results to these.

9.3.4 Statistical Distribution Fitting to the Bubble Size and Shape Distributions

By studying the shapes of the bubble size histogram distribution in Figures 9.1(a) and (b), it is clear that it might be possible to describe this distribution in terms of the log normal distribution. It may be seen from Figure 9.10, which shows an arbitrary example log-normal distribution, that this is a viable assumption.

In terms of a definition, the log normal distribution occurs in practice whenever a random variable is encountered, which is such that its logarithm has a normal distribution. Its probability density, $p(x)$ is given by [16]

$$p(x) = \frac{1}{X_m \ln \sigma_g \sqrt{2\pi}} \exp[-\frac{\ln^2 \sigma_g}{2}] \cdot \exp[-\frac{(\ln X - \ln X_m)^2}{2 \ln^2 \sigma_g}] \tag{9.26}$$

where X_m is the mode of the distribution and σ_g is a characteristic constant. The mean, $(\overline{x})$, and variance, (σ^2), are given by

$$\overline{x} = \exp(\theta + \frac{1}{2}s^2) \tag{9.27}$$

$$\sigma^2 = W(W-1)\exp(2\theta) \tag{9.28}$$

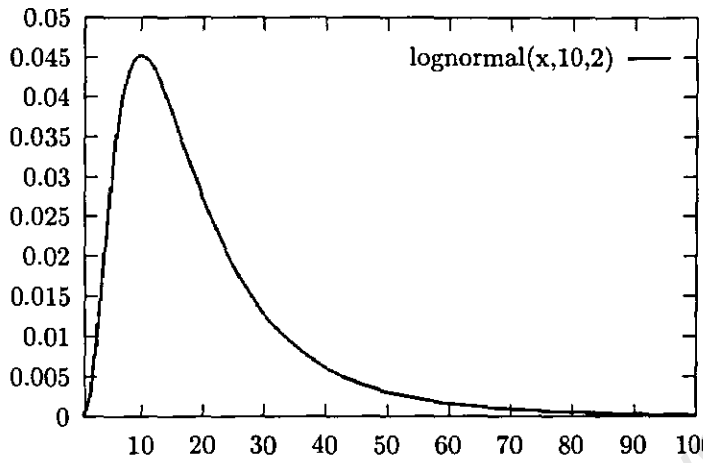


Figure 9.10: Graph of log-normal distribution, with $X_m = 10$ and $\sigma_g = 2$.

where $s = \ln \sigma_g$, $\theta = \ln X_m + \ln^2 \sigma_g$ and $W = \exp(s^2)$.

The expected and observed area and perimeter distributions, with fitted log-normal plots are shown in Figures 9.11 (area, expected), 9.12 (area, observed), 9.13 (perimeter, expected) and 9.14 (perimeter, observed). Note that the fits and plots were obtained by using Statgraphics V6.0. Further, Tables 9.7, 9.8, 9.9 and 9.10 provide the χ^2 test calculation results, in the same logical order as the figures. The analysis of these results follows thereafter.

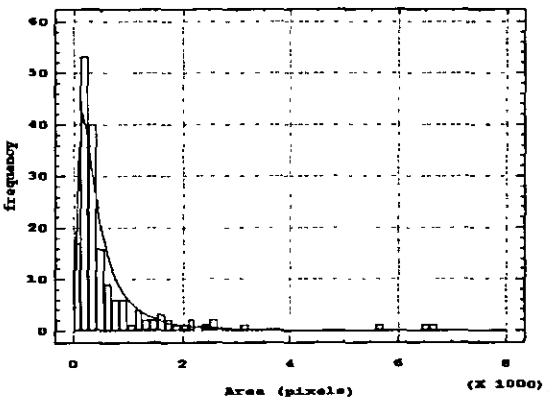


Figure 9.11: Log normal fit of the expected area distribution, with best fit parameters $\overline{x} = 572.75$ and $\sigma = 655.33$.

Interval	O_i	E_i	χ^2
1 -140	17	24	2.03
141 - 280	53	40	4.15
281 - 420	40	30	3.32
421 - 560	16	20.7	1.06
561 - 700	9	14.3	1.98
701 - 840	6	10.1	1.67
841 - 980	6	7.3	0.22
981 - 1120	1	5.4	3.55
1121 - 1400	6	7.1	0.16
1401 - 1820	7	5.7	0.29
1821 - 7000	11	7.3	1.84
Total	172	172	20.31

Table 9.7: χ^2 test for the expected area distribution versus log normal fit.

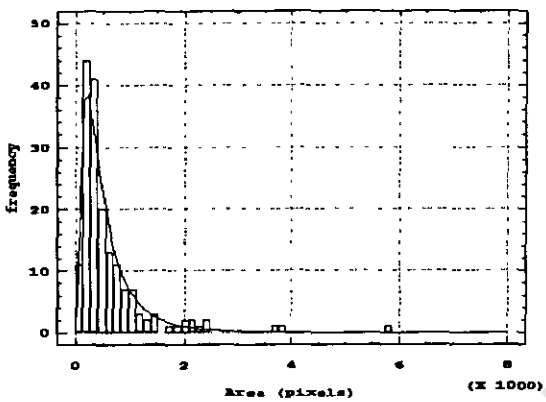


Figure 9.12: Log normal fit of the observed area distribution, with best fit parameters $\overline{x} = 600.56$ and $\sigma = 614.67$.

Interval	O_i	E_i	χ^2
1 - 140	11	16.9	2.07
141 - 280	44	38.1	0.91
281 - 420	41	32.0	2.51
421 - 560	20	23.1	0.42
561 - 700	13	16.3	0.67
701 - 840	11	11.6	0.03
841 - 980	7	8.3	0.21
981 - 1120	7	6.1	0.13
1121 - 1400	5	8.0	1.10
1401 - 1820	4	6.2	0.79
1821 - 7000	11	7.2	1.96
Total	174	174.1	10.86

Table 9.8: χ^2 test for the observed area distribution versus log normal fit.

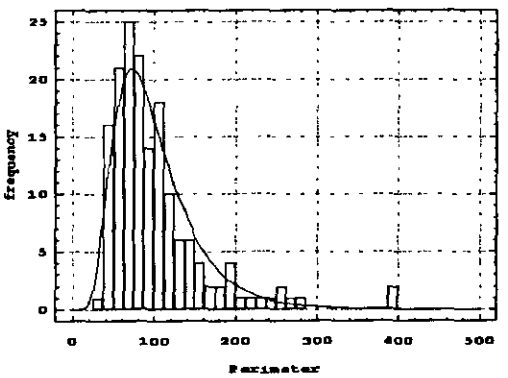


Figure 9.13: Log normal fit of the expected perimeter distribution, with best fit parameters $\overline{x} = 100.83$ and $\sigma = 51.13$.

Interval	O_i	E_i	χ^2
1 - 37.5	1	5.4	3.61
37.5 - 50	16	12.3	1.14
50 - 62.5	21	18.3	0.40
62.5 - 75	25	20.7	0.88
75 - 87.5	22	20.1	0.17
87.5 - 100	14	17.8	0.82
100 - 112.5	18	14.9	0.66
112.5 - 125	10	12.0	0.32
125 - 137.5	6	9.4	1.21
137.5 - 150	6	7.2	0.21
150 - 162.5	4	5.5	0.42
162.5 - 187.5	4	7.4	1.54
187.5 - 500	14	10.0	1.57
Total	161	161	12.98

Table 9.9: χ^2 test for the expected perimeter distribution versus log normal fit.

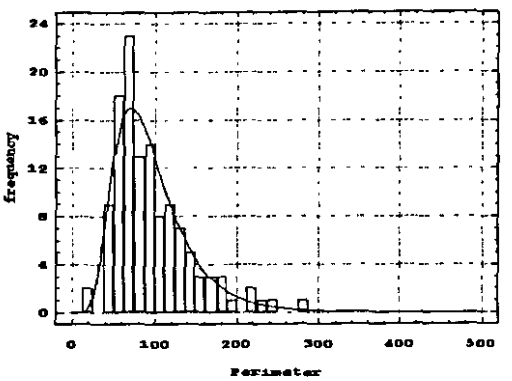


Figure 9.14: Log normal fit of the observed perimeter distribution, with best fit parameters $\overline{x} = 96.78$ and $\sigma = 47.02$.

Interval	O_i	E_i	χ^2
1 - 50	11	14.0	0.66
50 - 62.5	18	15.0	0.62
62.5 - 75	23	16.9	2.21
75 - 87.5	13	16.2	0.61
87.5 - 100	14	14.0	0.00
100 - 112.5	8	11.4	1.02
112.5 - 125	9	9.0	0.00
125 - 137.5	7	6.8	0.00
137.5 - 150	5	5.1	0.00
150 - 175	6	6.6	0.06
175 - 500	9	8.0	0.14
Total	123	123	5.32

Table 9.10: χ^2 test for the observed perimeter distribution versus log normal fit.

Analysis of Results: Referring to Table 9.7, the χ^2 calculated value is 20.31, compared with the critical value 18.30, the conclusion is **Reject** H_0. Referring to Table 9.8, the χ^2 calculated value is 10.86, compared with the critical value 18.30, the conclusion is **Accept** H_0. Referring to Table 9.9, the χ^2 calculated value is 12.98, compared with the critical value 21.02, the conclusion is **Accept** H_0. Referring to Table 9.10, the χ^2 calculated value is 5.32, compared with the critical value 18.30, the conclusion is **Accept** H_0.

In summary, it may be seen that overall, the expected and observed distributions are highly similar, thus proving that the watershed segmentation algorithm is able to segment a surface froth image successfully and accurately.

Chapter 10

Further Segmentation of Various Surface Froth Images

For a useful practical implementation, the automatic image analysis system should be able to handle a wide range of different images. Thus, to demonstrate the success of automatical segmentation of surface froth images by using the watershed segmentation algorithm, in addition to the test image BUB1, this chapter presents seven more results of segmentation of various surface froth images, namely BUB2 to BUB8.

Images BUB1 to BUB4 and images BUB5 to BUB8 are two groups of images and represent two different froth structures. Images BUB1 to BUB4 were chosen to represent low highlight conditions, while images BUB5 to BUB8 were chosen to represent multiple highlight conditions. Further, images BUB1 to BUB4 form a continuous process, meaning that they are of the same surface froth, representing one time frame after another. The same holds for images BUB5 to BUB8.

The proposed segmentation algorithm is reliant on a number of parameters which control the segmentation process. If the images are visually similar and under similar lighting conditions, the parameters will only require optimisation for one image. Thereafter, all these images can be processed with the same parameter setup.

For segmentation of the two image groups, the parameters are almost the same except for the area of interest (AOI) and the post-processing stage. The images BUB5 to BUB8 have multiple highlight conditions and have comparatively big size bubbles, so that *Variance* processing is applied at the post-processing stage, as introduced in Chapter 8.

Also, in images BUB5 to BUB8 the total number of manually segmented bubbles

in the AOI region is different to that of Symonds' [21], because even though the same images were used, a number of small area regions were lost due to filtering. Note here that the aim of the segmentation in this case concentrates on the middle size and large size bubbles. The loss of small area bubbles in the computer segmented image is easily offset in the manually segmented image by thresholding the VLABEL program. The small area regions whose value is less than 0.01% of the total AOI image area value are given a label of zero, and are then ignored.

The statistical data analyses of the segmentation results provide a set of expected (manual segmentation) and observed (automatical segmentation) *Area, Circularity* and *Eccentricity* distributions. The graphs modelling the *Size* distributions using a best fit log normal distribution are shown as well.

10.1 Processed Images

In this section, the results of segmentation of images BUB2 to BUB8 are all given in the same format, as follows:

1. Original image.
2. Manually segmented AOI of image, overlaid on the original image.
3. Automatically segmented AOI of image, overlaid on the original image.
4. Area distribution for (a) the manually segmented image, and (b) the automatically segmented image.
5. Log normal test fits for (a) the area distribution determined from the manually segmented image, and (b) the area distribution determined from the automatically segmented image.
6. Overlaid area distributions of manually and automatically segmented images.
7. Cumulative distributions of manually and automatically segmented images.
8. Eccentricity distributions for (a) the manually segmented image, and (b) the automatically segmented image.
9. Circularity distributions for (a) the manually segmented image, and (b) the automatically segmented image.

The values of the mean, standard deviation, skewness and mode are given with each distribution. For size distributions, the unit is pixels. For circularity and eccentricity, since measurements are dimensionless quantities, the corresponding statistics have no units.

10.1.1 Image BUB2

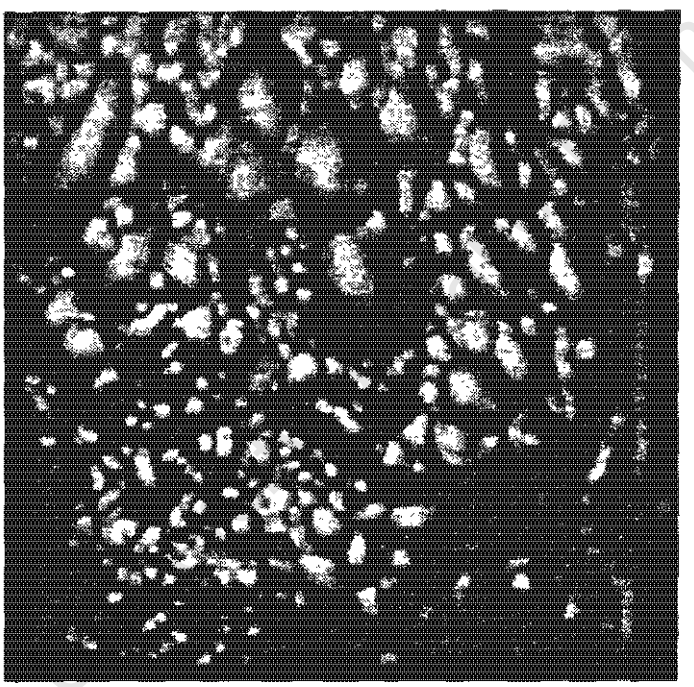
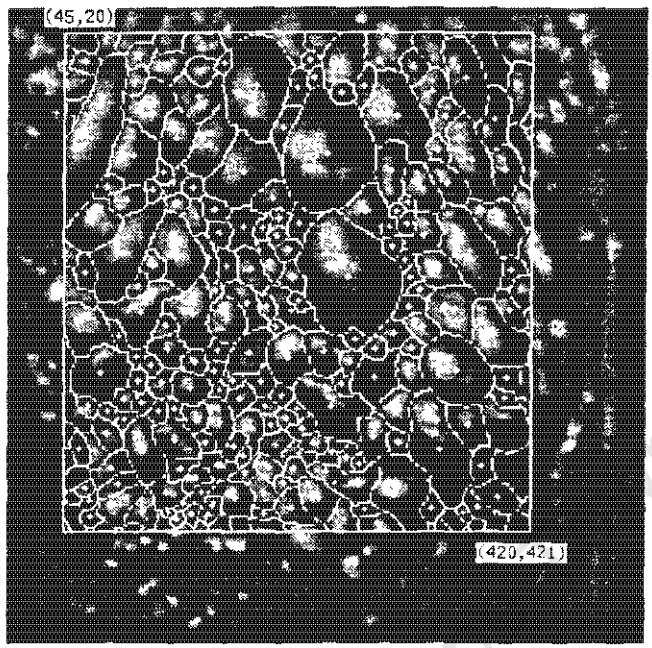
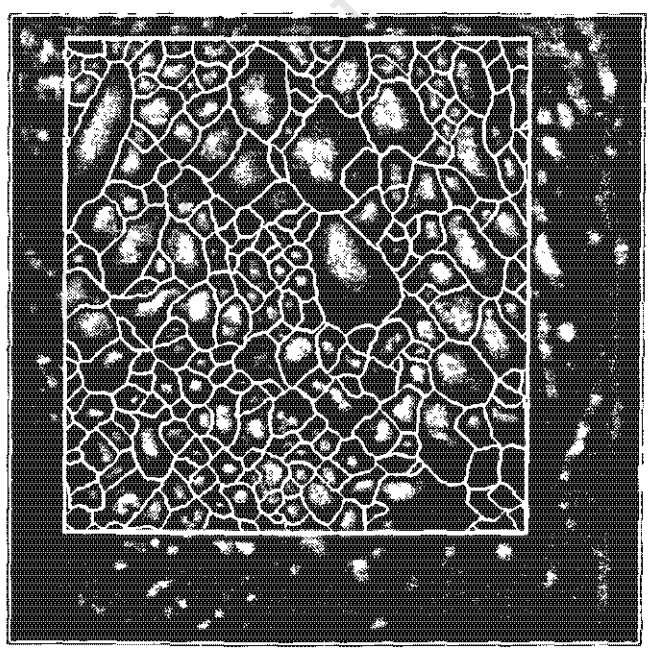


Figure 10.1: Original Image BUB2.

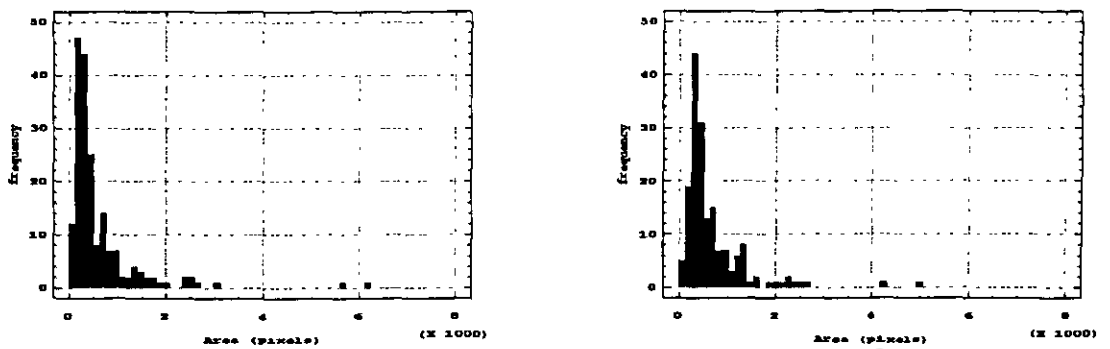


(a) Manually segmented AOI of BUB2.



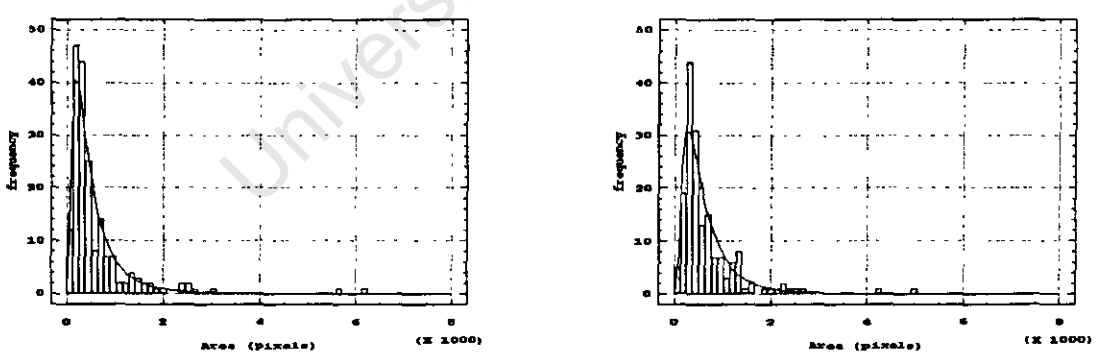
(b) Automatically segmented AOI of BUB2.

Figure 10.2: Segmented AOI's of BUB2.



(a) (b)

Figure 10.3: (a) Expected *Area* distribution with $\overline{x} = 609.73$, $\sigma = 768.21$, mode = 211 and skewness = 4.14. (b) Observed *Area* distribution with $\overline{x} = 680.67$, σ = 666.76, mode = 423 and skewness = 3.21.



(a) (b)

Figure 10.4: (a) Log normal fit to the expected *Area* distribution with the best fit parameters $\overline{x} = 581.21$ and $\sigma = 611.53$. (b) Log normal fit to the observed *Area* distribution, with the best fit parameters $\overline{x} = 666.09$ and $\sigma = 585.97$.

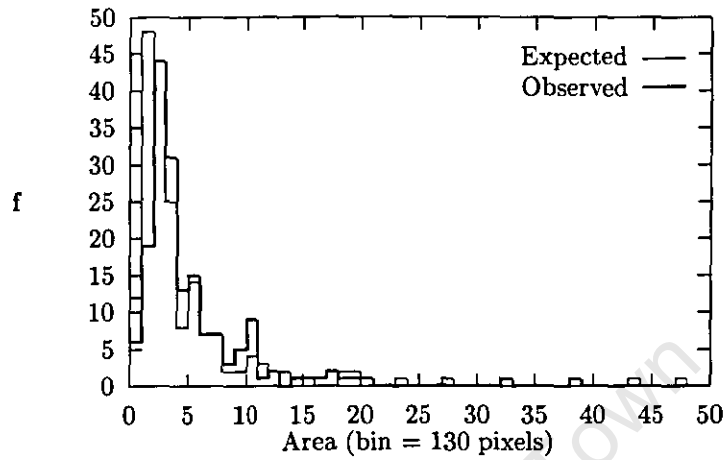


Figure 10.5: Overlaid graphs of BUB2 area distributions.

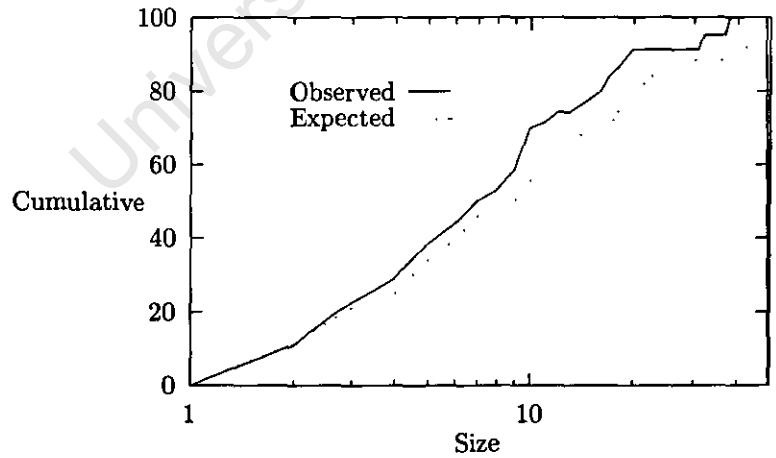
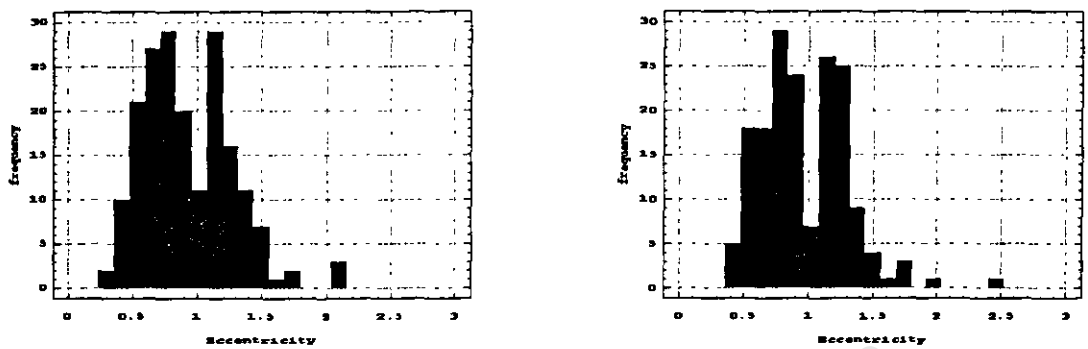
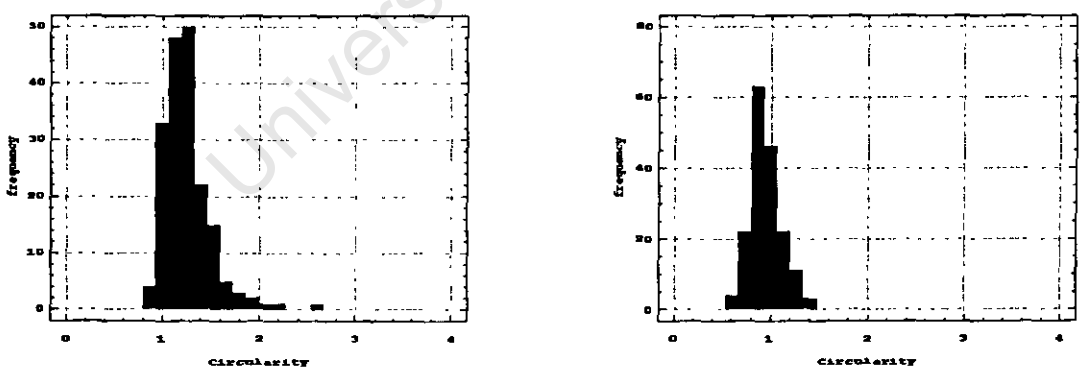


Figure 10.6: Cumlative size distributions of BUB2.



(a) (b)

Figure 10.7: (a) Expected *Eccentricity* distribution with $\overline{x} = 0.93$, $\sigma = 0.34$, mode = 0.87 and skewness = 0.68. (b) Observed *Eccentricity* distribution with $\overline{x} = 0.89$, $\sigma = 0.33$, mode = 0.89 and skewness = 0.79.



(a) (b)

Figure 10.8: (a) Expected *Circularity* distribution with $\overline{x} = 1.32$, $\sigma = 0.54$, mode = 1.10 and skewness = 4.91. (b) Observed *Circularity* distribution with $\overline{x} = 0.95$, $\sigma = 0.15$, mode = 0.95 and skewness = 0.70.

10.1.2 Image BUB3

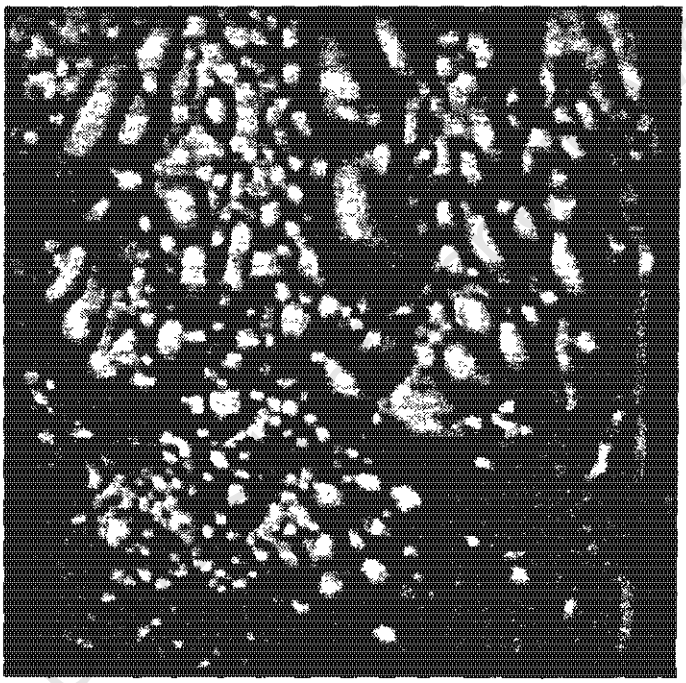
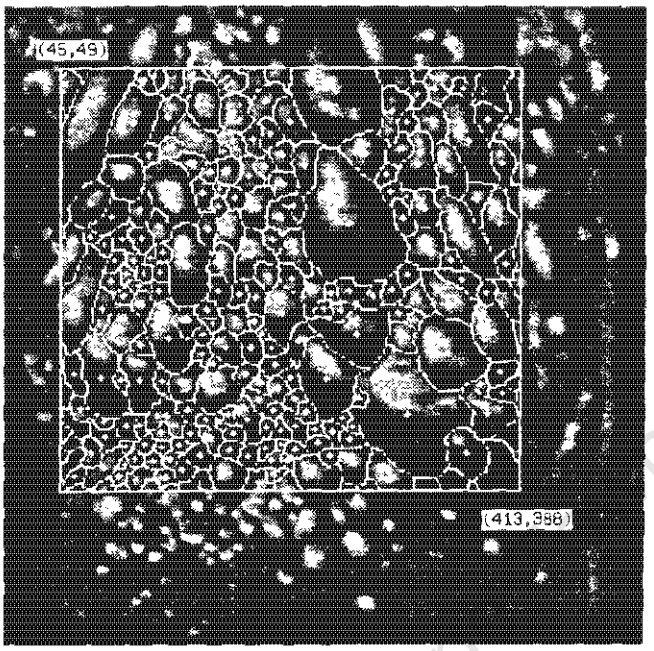
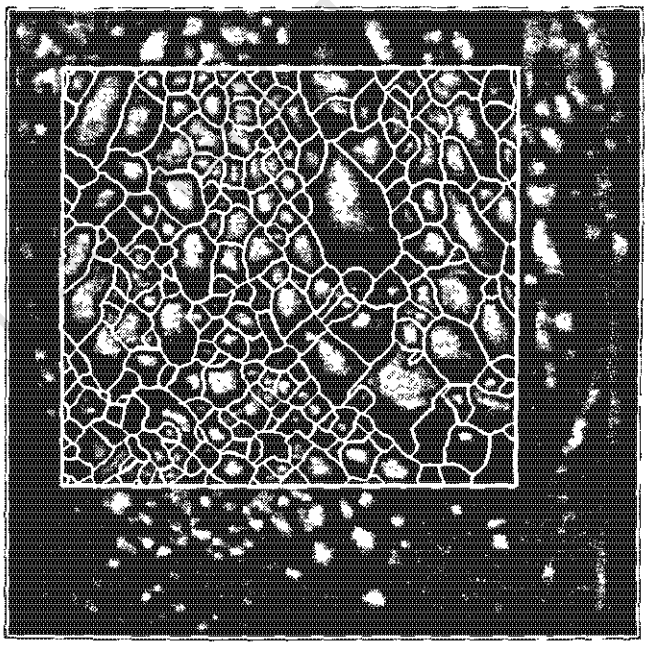


Figure 10.9: Original Image BUB3.

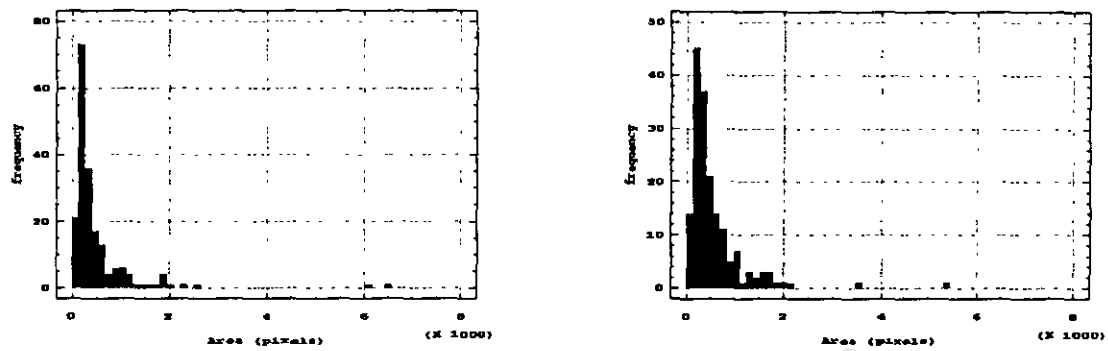


(a) Manually segmented AOI of BUB3.



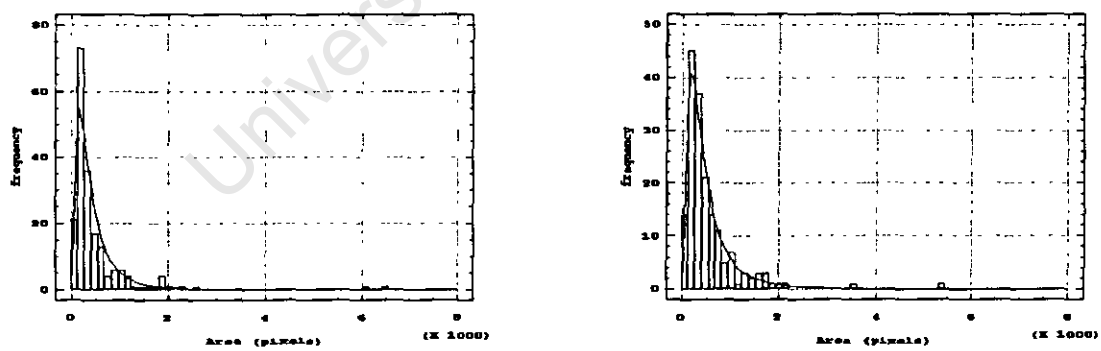
(b) Automatically segmented AOI of BUB3.

Figure 10.10: Segmented AOI's BUB3.



(a) (b)

Figure 10.11: (a) Expected *Area* distribution with $\overline{x} = 510.23$, $\sigma = 740.54$, mode = 188 and skewness = 5.51. (b) Observed *Area* distribution with $\overline{x} = 559.76$, σ = 613.89, mode = 204 and skewness = 4.12.



(a) (b)

Figure 10.12: (a) Log normal fit to the expected *Area* distribution with the best fit parameters $\overline{x} = 470.55$ and $\sigma = 469.07$. (b) Log normal fit to the observed *Area* distribution, with the best fit parameters $\overline{x} = 557.88$ and $\sigma = 579.00$.

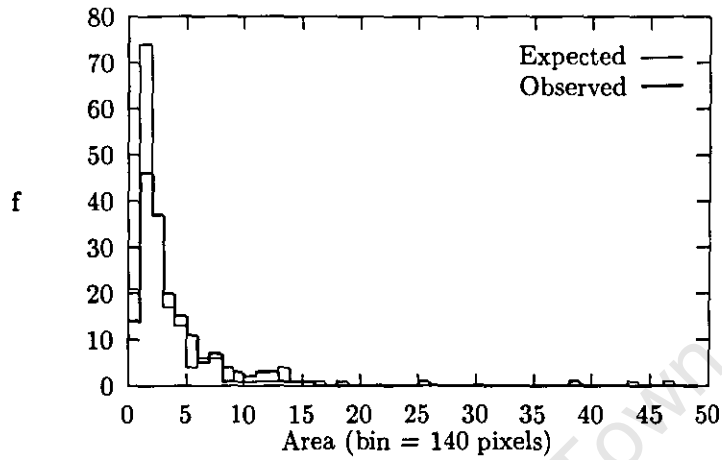


Figure 10.13: Overlaid graphs of BUB3 area distributions.

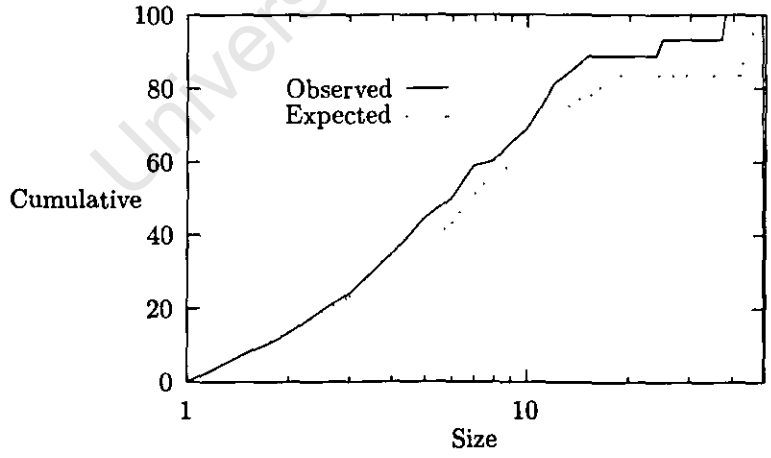
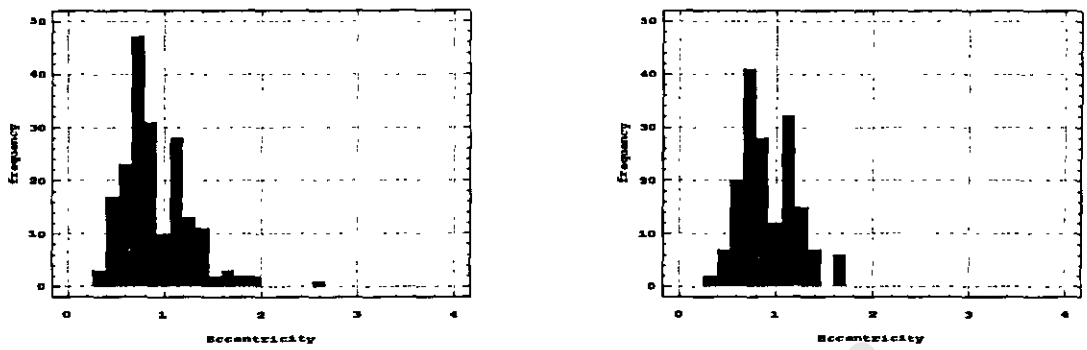
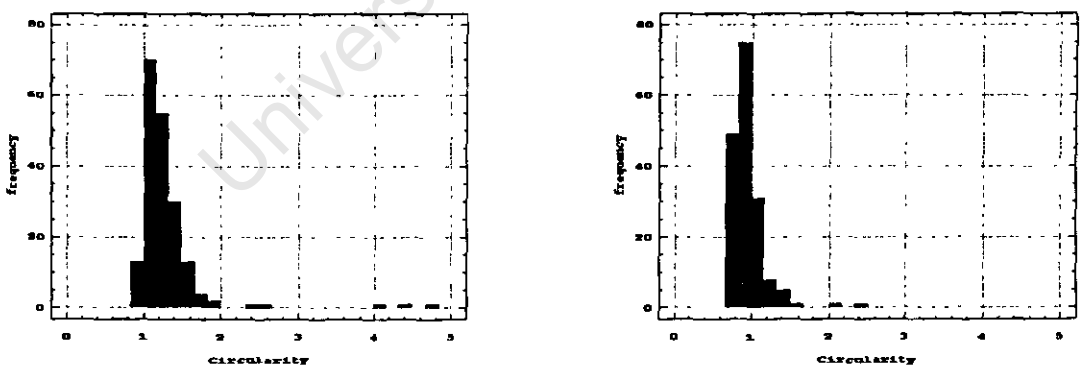


Figure 10.14: Cumlative size distributions of BUB3.



(a) (b)

Figure 10.15: (a) Expected *Eccentricity* distribution with $\overline{x} = 0.91$, $\sigma = 0.34$, mode = 0.80 and skewness = 1.16. (b) Observed *Eccentricity* distribution with $\overline{x} = 1.03$, $\sigma = 1.42$, mode = 0.86 and skewness = 12.28.



(a) (b)

Figure 10.16: (a) Expected *Circularity* distribution with $\overline{x} = 1.31$, $\sigma = 0.52$, mode = 1.07 and skewness = 5.14. (b) Observed *Circularity* distribution with $\overline{x}$ = 0.94, $\sigma = 0.22$, mode = 1.05 and skewness = 3.35.

10.1.3 Image BUB4

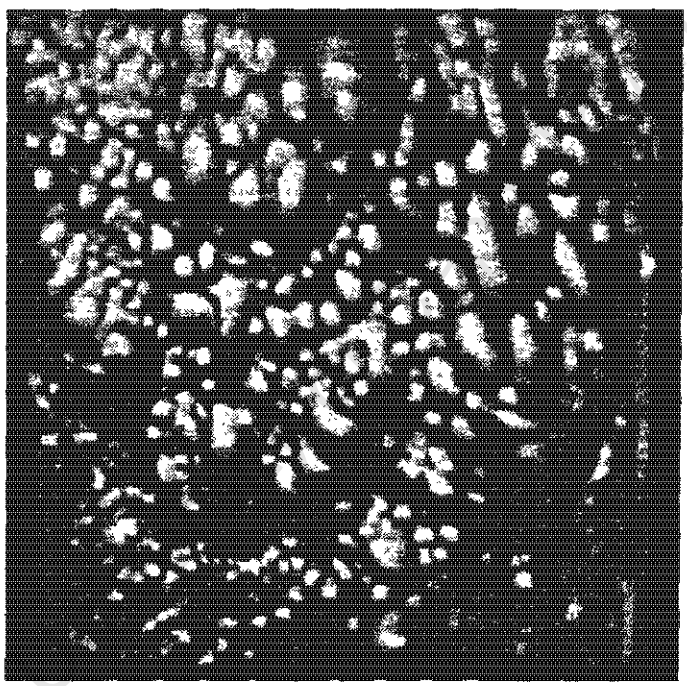
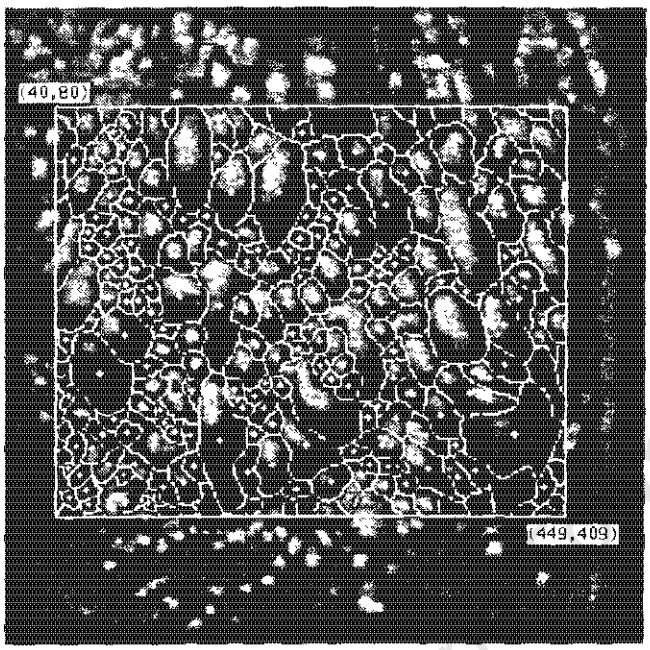
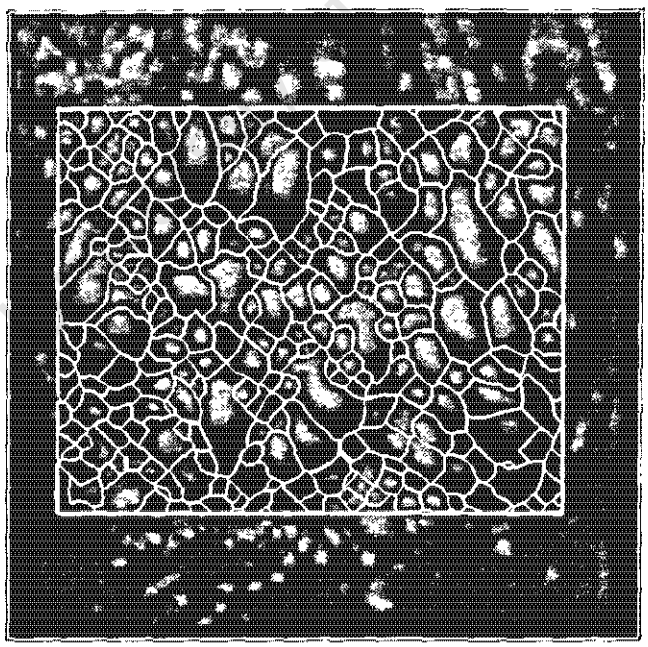


Figure 10.17: Original Image BUB4.

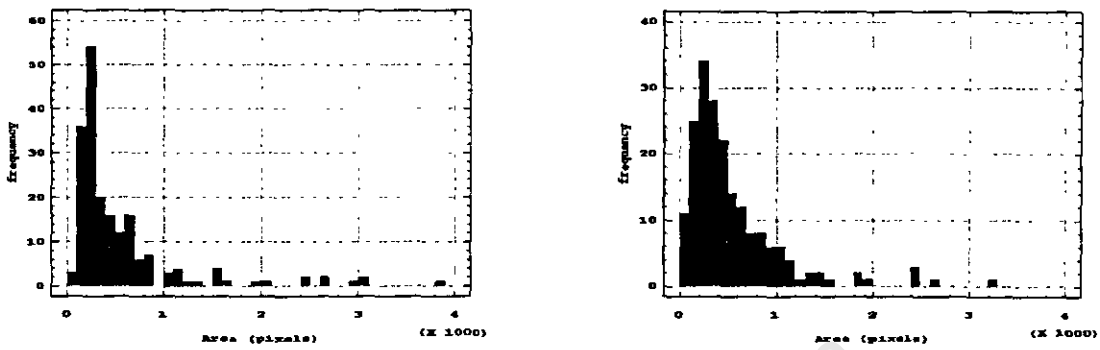


(a) Manually segmented AOI of BUB4.



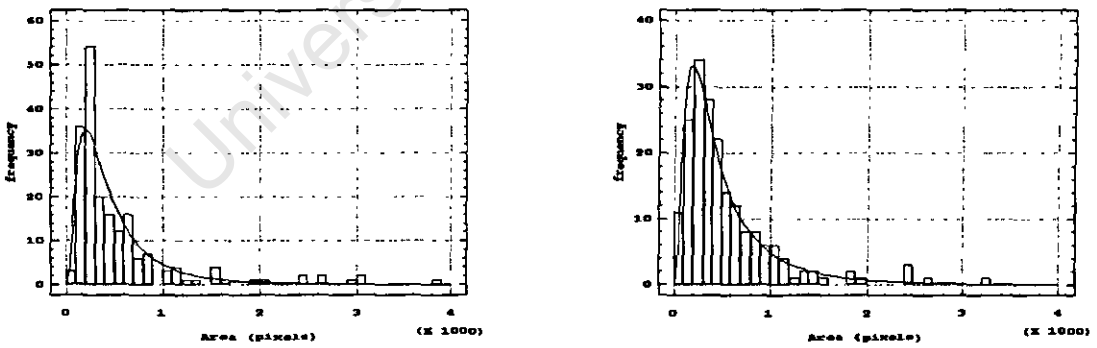
(b) Automatically segmented AOI of BUB4.

Figure 10.18: Segmented AOI's BUB4.



(a) (b)

Figure 10.19: (a) Expected *Area* distribution with $\overline{x} = 545.11$, $\sigma = 608.52$, mode = 144 and skewness = 2.88. (b) Observed *Area* distribution with $\overline{x} = 543.91$, σ = 500.54, mode = 233 and skewness = 2.47.



(a) (b)

Figure 10.20: (a) Log normal fit to the expected *Area* distribution with the best fit parameters $\overline{x} = 517.04$ and $\sigma = 485.55$. (b) Log normal fit to the observed *Area* distribution, with the best fit parameters $\overline{x} = 560.77$ and $\sigma = 588.43$.

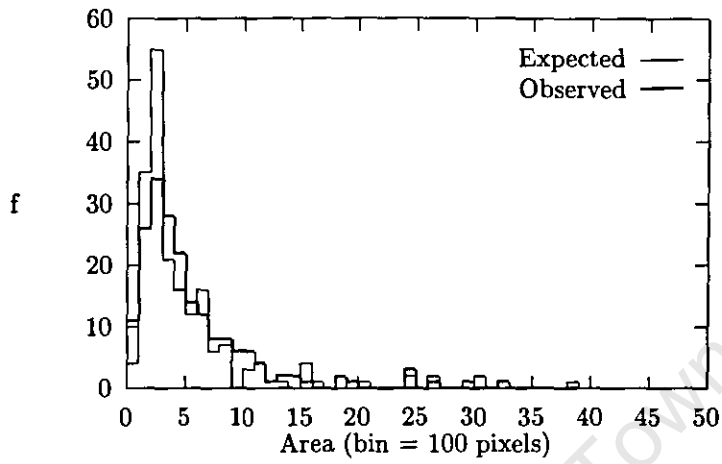


Figure 10.21: Overlaid graphs of BUB4 area distributions.

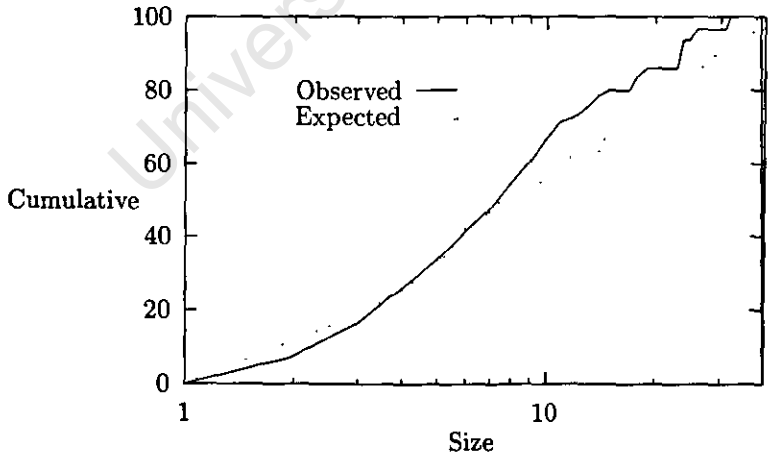
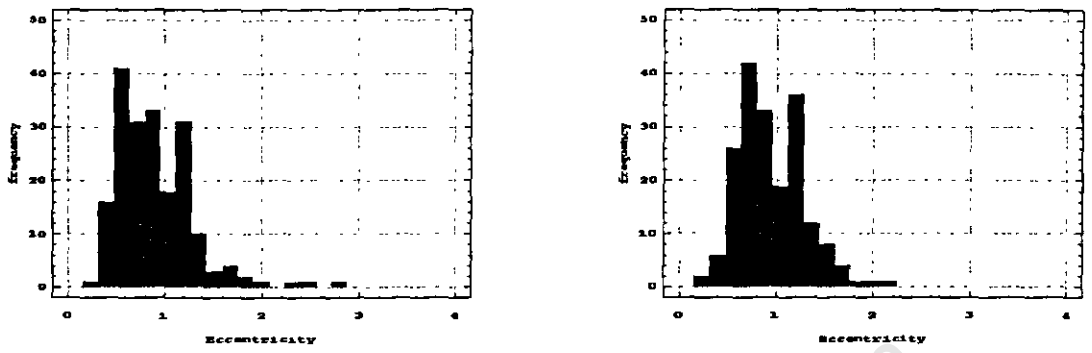
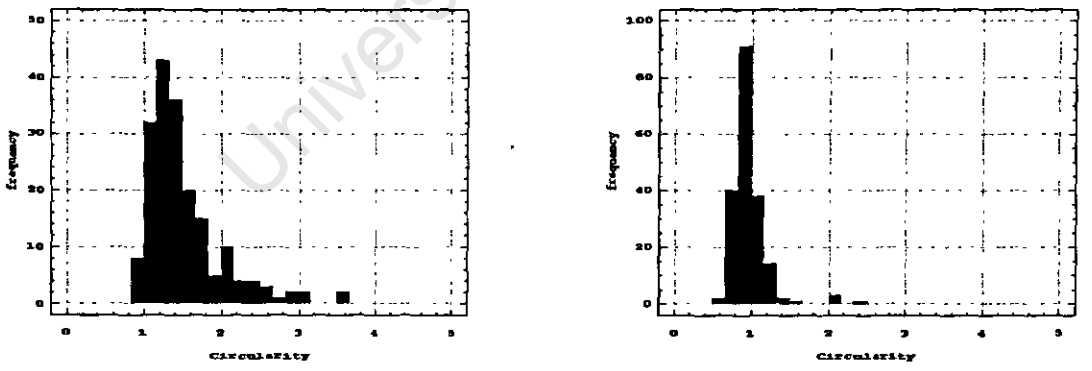


Figure 10.22: Cumlative size distributions of BUB4.



(a) (b)

Figure 10.23: (a) Expected *Eccentricity* distribution with $\overline{x} = 0.90$, $\sigma = 0.39$, mode = 0.81 and skewness = 1.48. (b) Observed *Eccentricity* distribution with $\overline{x} = 1.03$, $\sigma = 1.15$, mode = 0.88 and skewness = 12.16.



(a) (b)

Figure 10.24: (a) Expected *Circularity* distribution with $\overline{x} = 1.75$, $\sigma = 1.40$, mode = 1.55 and skewness = 4.51. (b) Observed *Circularity* distribution with $\overline{x}$ = 0.96, $\sigma = 0.23$, mode = 1.04 and skewness = 3.15.

10.1.4 Image BUB5

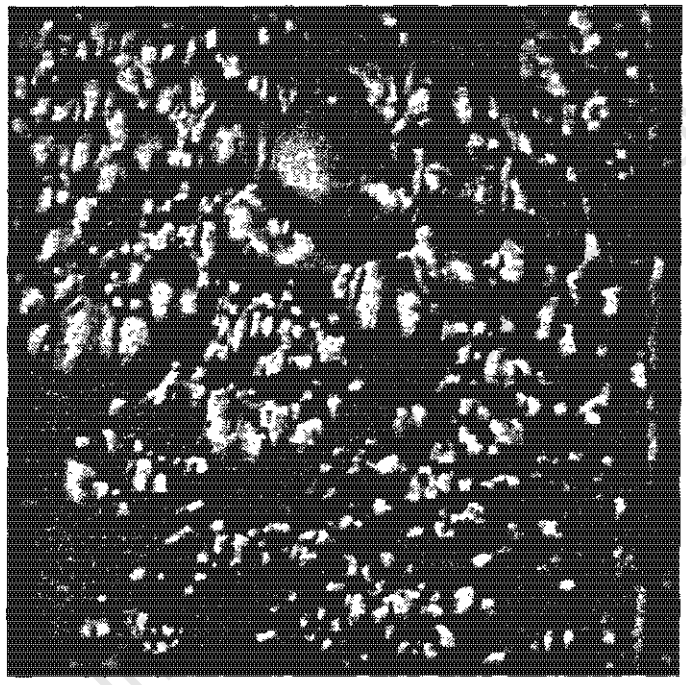
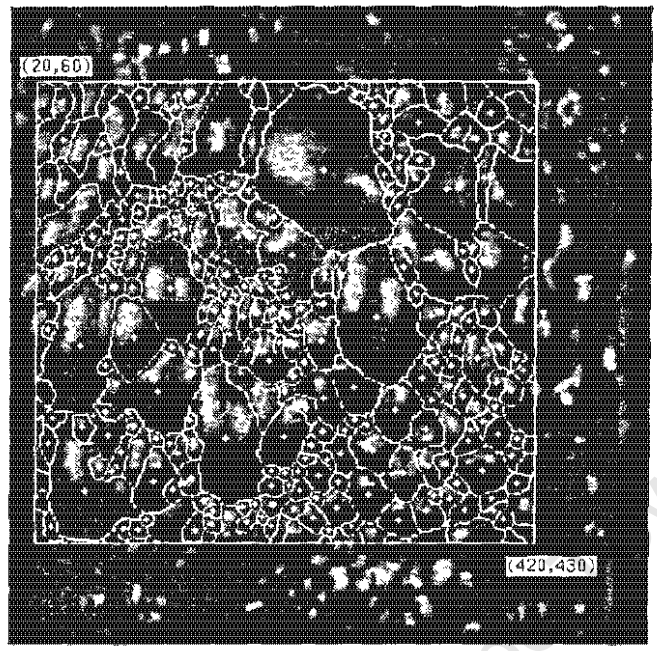
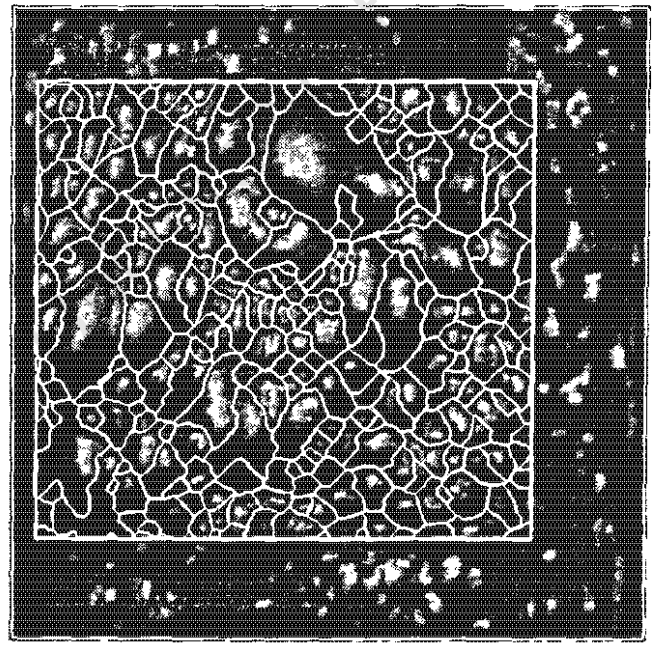


Figure 10.25: Original Image BUB5.

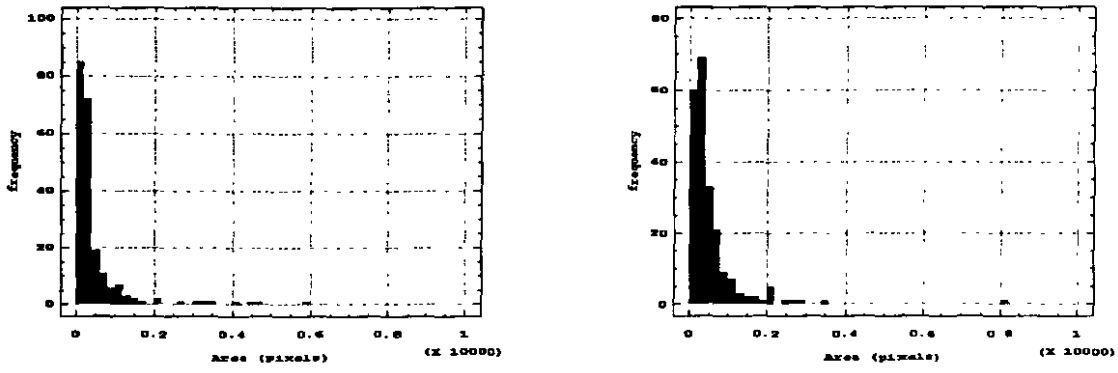


(a) Manually segmented AOI of BUB5.



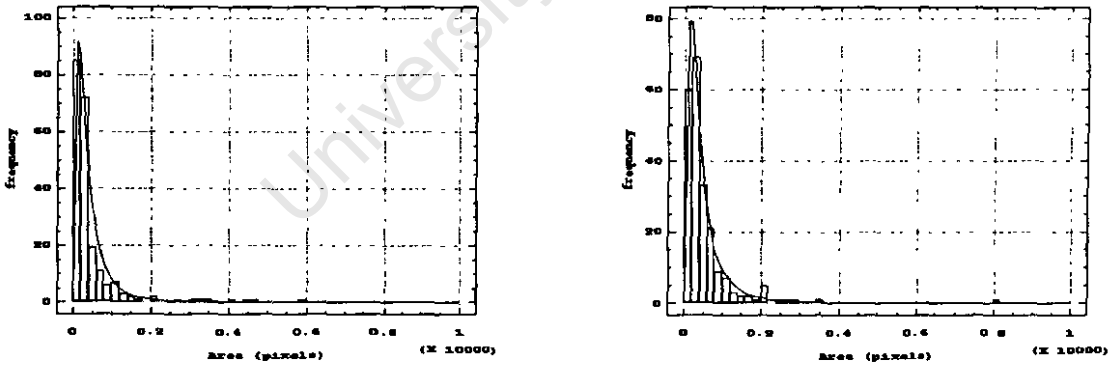
(b) Automatically segmented AOI of BUB5.

Figure 10.26: Segmented AOI's BUB5.



(a) (b)

Figure 10.27: (a) Expected *Area* distribution with $\overline{x} = 539.02$, $\sigma = 1047.81$, mode = 155 and skewness = 5.95. (b) Observed *Area* distribution with $\overline{x} = 538.27$, σ = 745.39, mode = 404 and skewness = 5.81.



(a) (b)

Figure 10.28: (a) Log normal fit to the expected *Area* distribution with the best fit parameters $\overline{x} = 447.44$ and $\sigma = 498.39$. (b) Log normal fit to the observed *Area* distribution, with the best fit parameters $\overline{x} = 522.65$ and $\sigma = 615.88$.

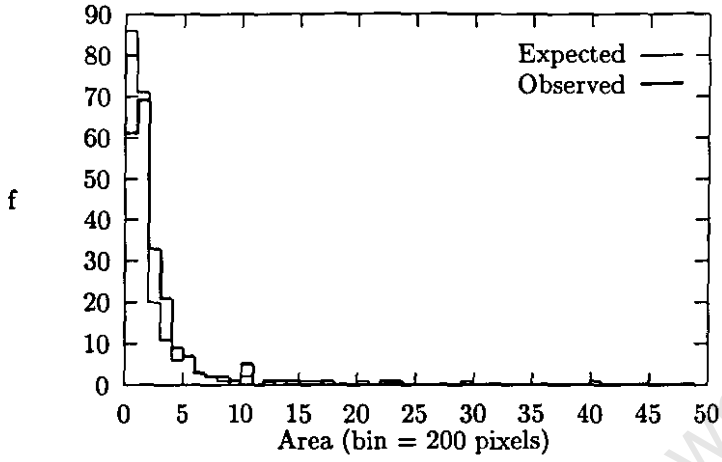


Figure 10.29: Overlaid graphs of BUB5 area distributions.

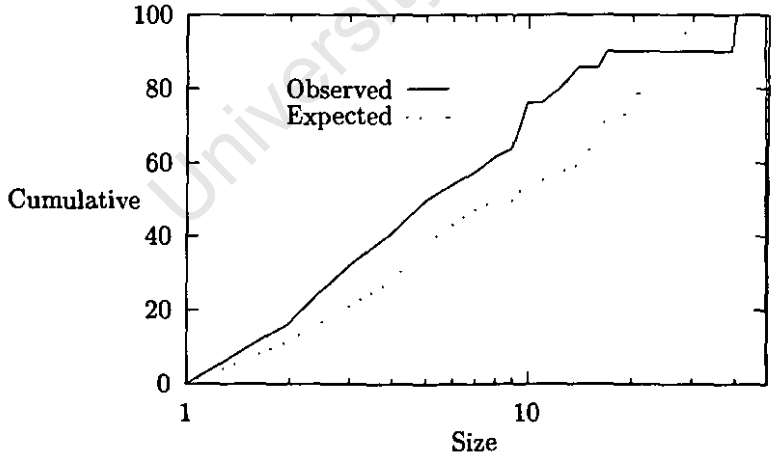
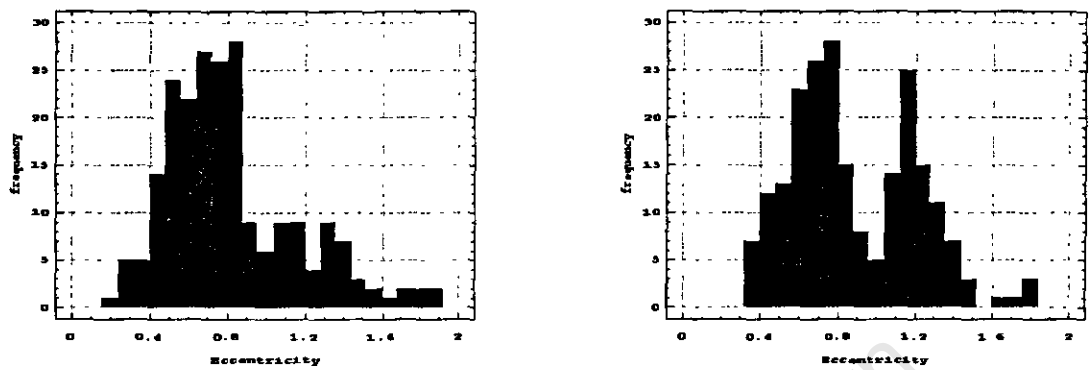
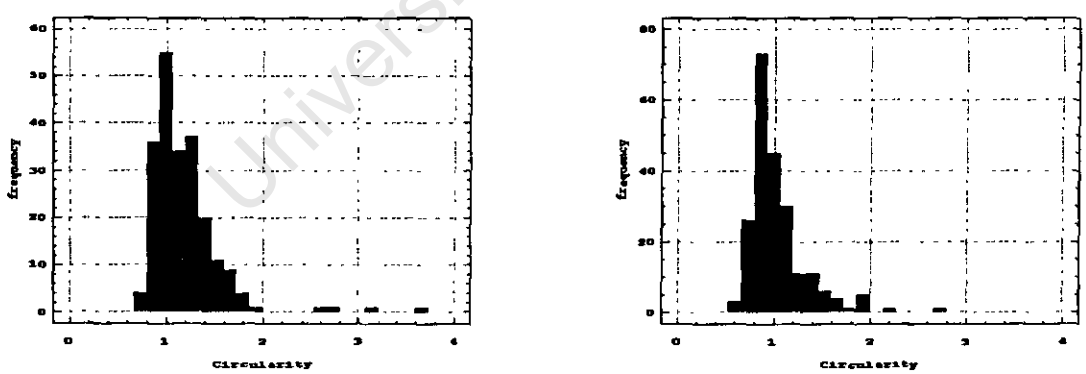


Figure 10.30: Cumlative size distributions of BUB5.



(a) (b)

Figure 10.31: (a) Expected *Eccentricity* distribution with $\overline{x} = 0.81$, $\sigma = 0.33$, mode = 0.75 and skewness = 0.96. (b) Observed *Eccentricity* distribution with $\overline{x} = 0.88$, $\sigma = 0.32$, mode = 0.79 and skewness = 0.46.



(a) (b)

Figure 10.32: (a) Expected *Circularity* distribution with $\overline{x} = 1.24$, $\sigma = 0.67$, mode = 1.07 and skewness = 6.76. (b) Observed *Circularity* distribution with $\overline{x}$ = 1.03, $\sigma = 0.29$, mode = 1.10 and skewness = 2.05.

10.1.5 Image BUB6

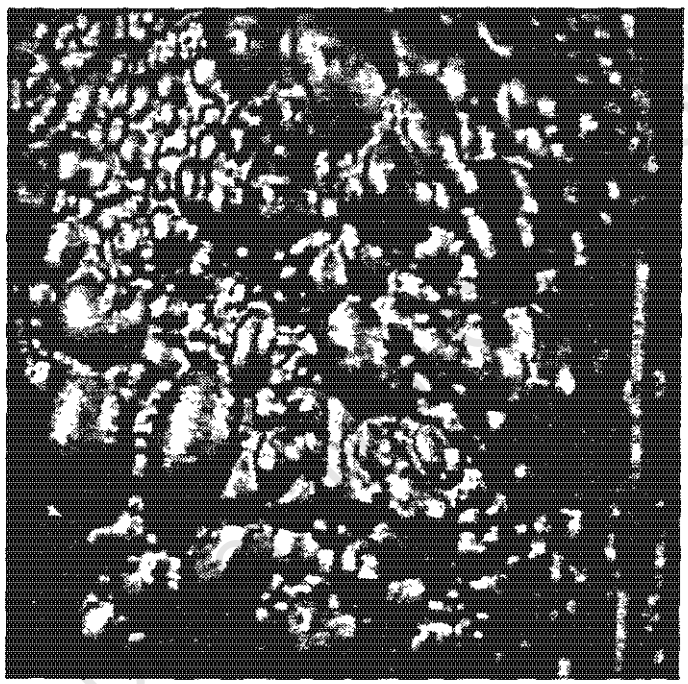
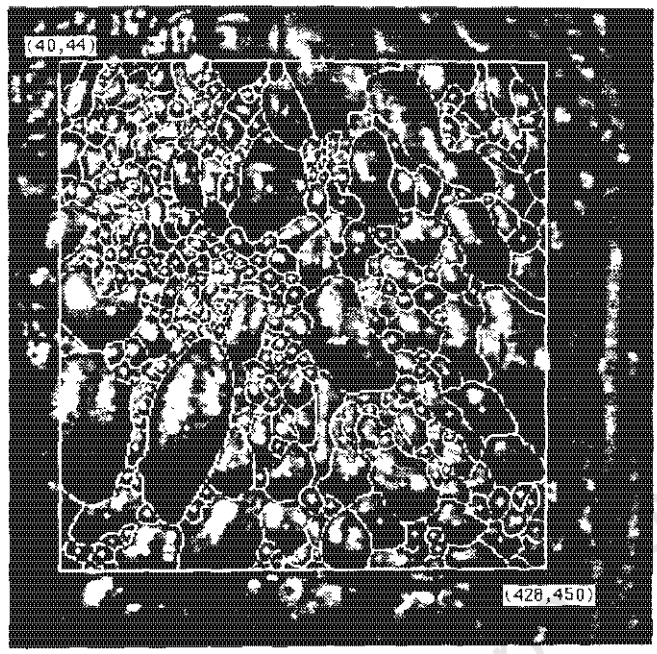
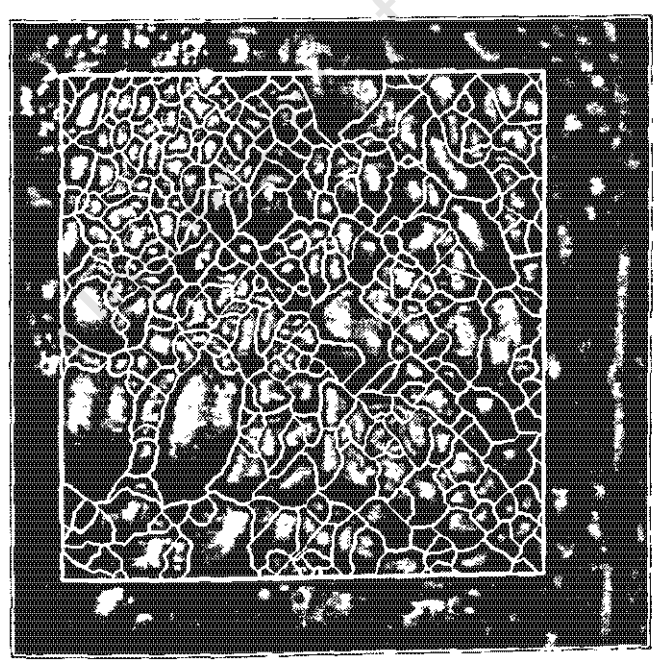


Figure 10.33: Original Image BUB6.

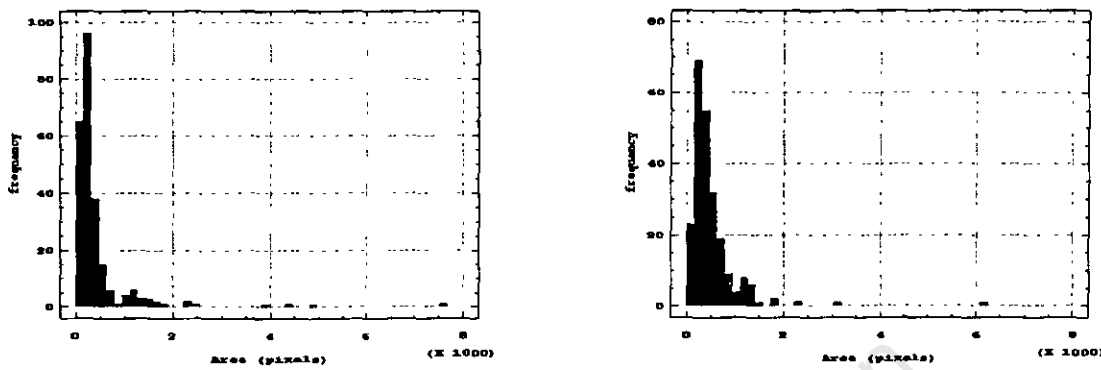


(a) Manually segmented AOI of BUB6.



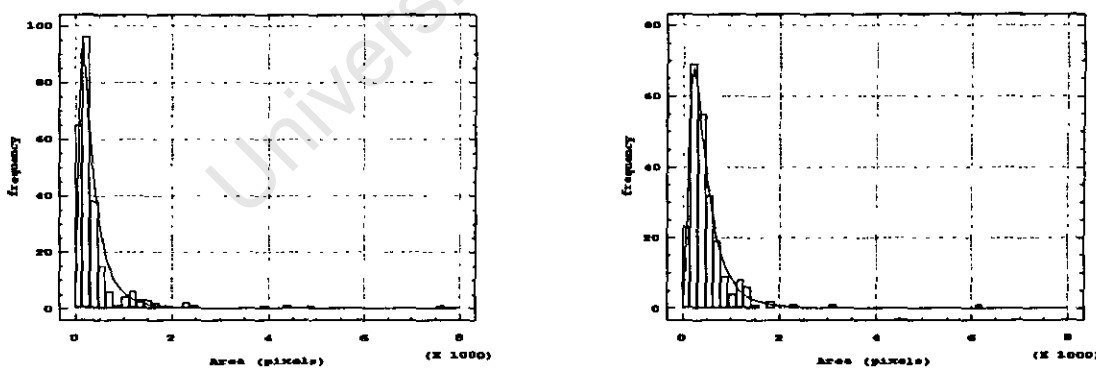
(b) Automatically segmented AOI of BUB6.

Figure 10.34: Segmented AOI's BUB6.



(a) (b)

Figure 10.35: (a) Expected *Area* distribution with $\overline{x} = 446.43$, $\sigma = 743.95$, mode = 170 and skewness = 5.73. (b) Observed *Area* distribution with $\overline{x} = 508.49$ σ = 541.26, mode = 199 and skewness = 5.94.



(a) (b)

Figure 10.36: (a) Log normal fit to the expected *Area* distribution with the best fit parameters $\overline{x} = 388.53$ and $\sigma = 374.32$ (b) Log normal fit to the observed *Area* distribution, with the best fit parameters $\overline{x} = 500.85$ and $\sigma = 431.84$.

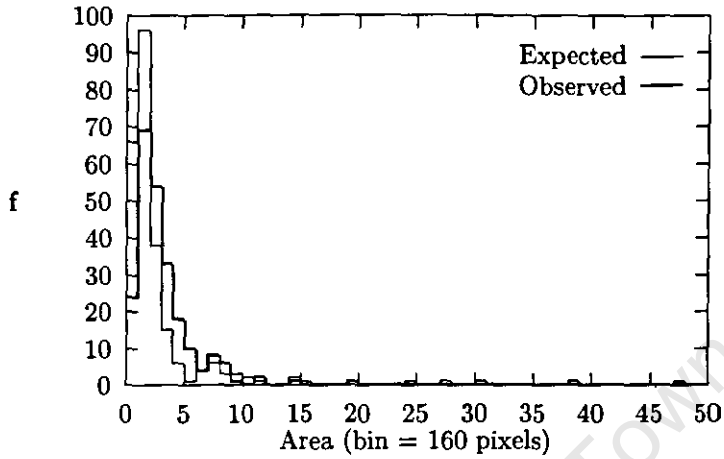


Figure 10.37: Overlaid graphs of BUB6 area distributions.

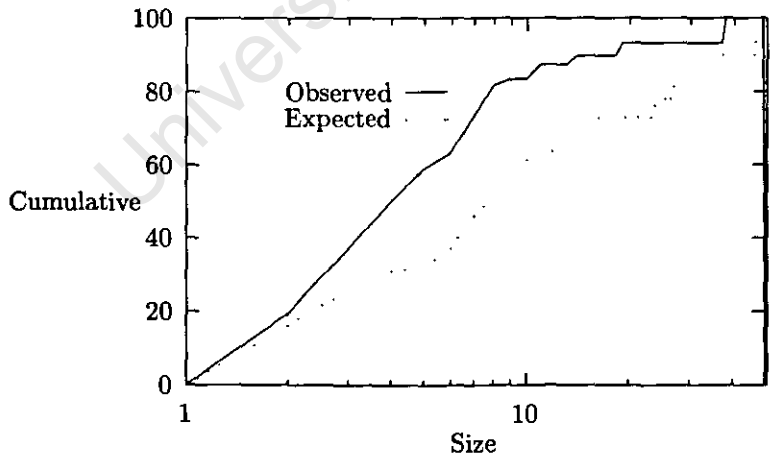
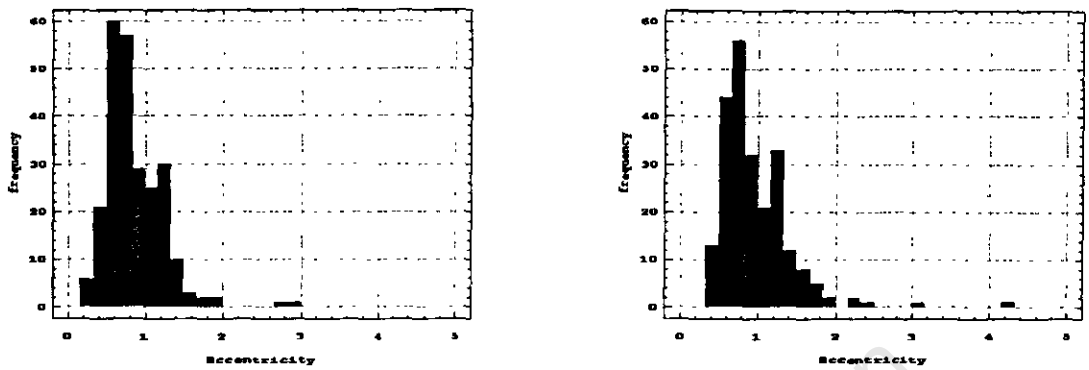
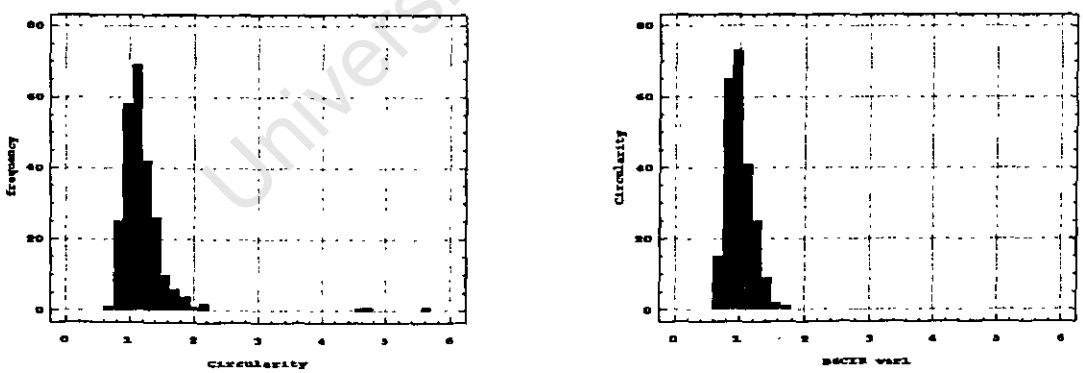


Figure 10.38: Cumlative size distributions of BUB6.



(a) (b)

Figure 10.39: (a) Expected *Eccentricity* distribution with $\overline{x} = 0.85$, $\sigma = 0.36$, mode = 0.75 and skewness = 1.64. (b) Observed *Eccentricity* distribution with $\overline{x} = 0.96$, $\sigma = 0.45$, mode = 0.92 and skewness = 2.72.



(a) (b)

Figure 10.40: (a) Expected *Circularity* distribution with $\overline{x} = 1.22$, $\sigma = 0.48$, mode = 1.03 and skewness = 5.93. (b) Observed *Circularity* distribution with $\overline{x}$ = 0.99, $\sigma = 0.19$, mode = 1.05 and skewness = 0.83.

10.1.6 Image BUB7

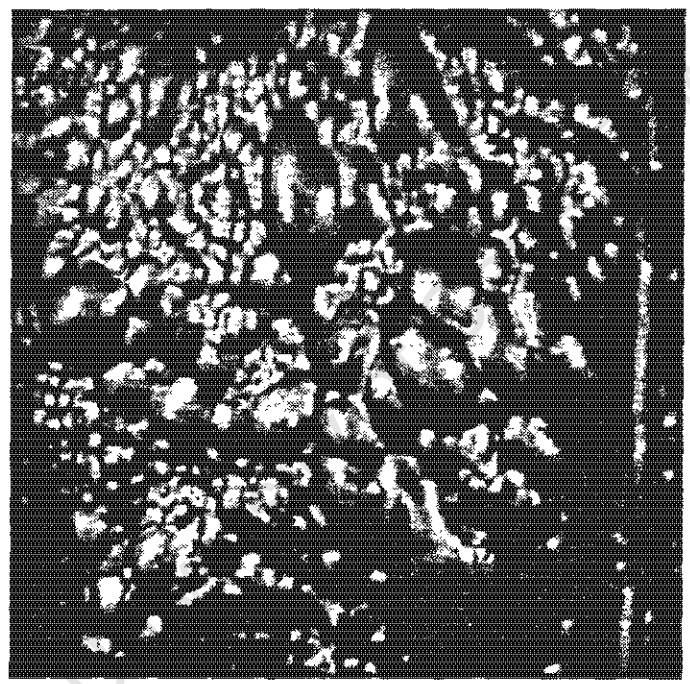
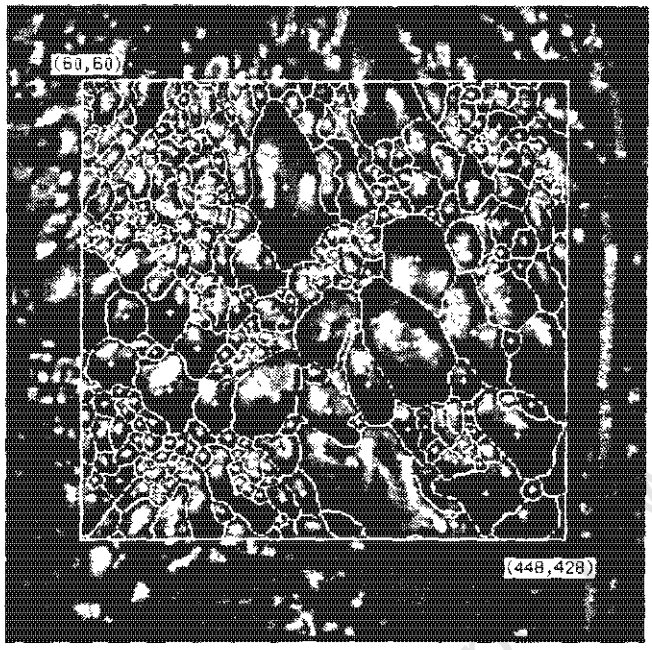
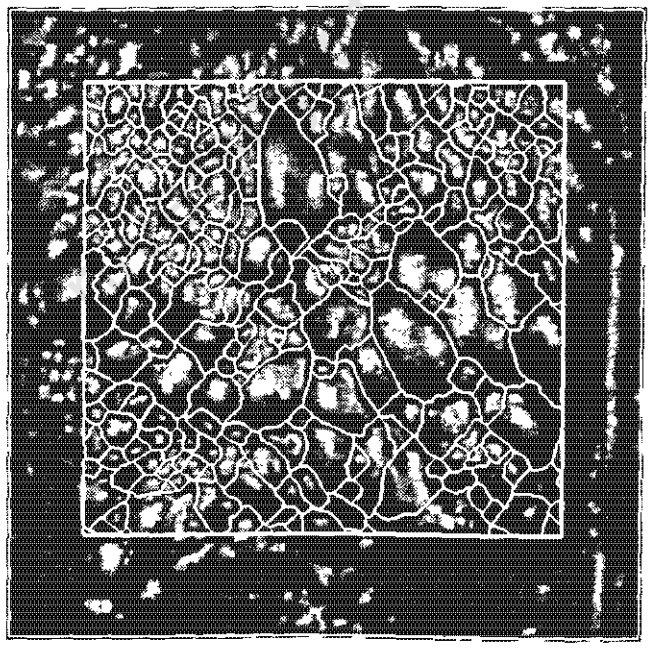


Figure 10.41: Original Image BUB7.

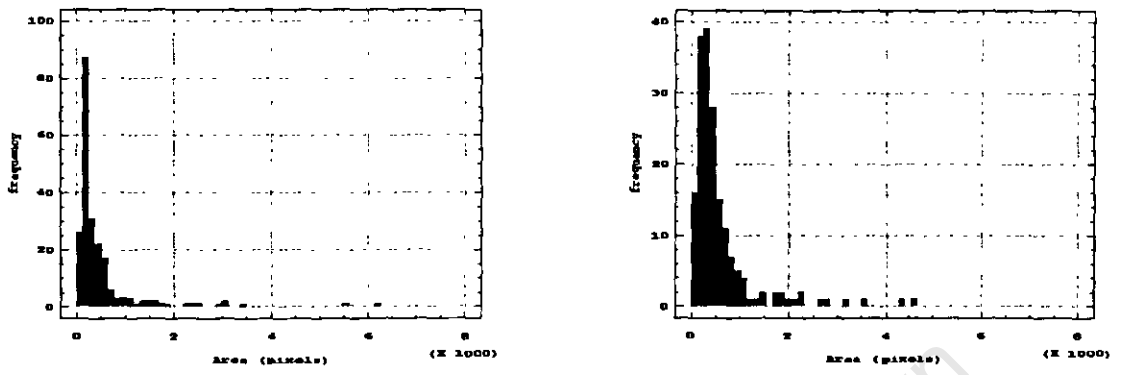


(a) Manually segmented AOI of BUB7.



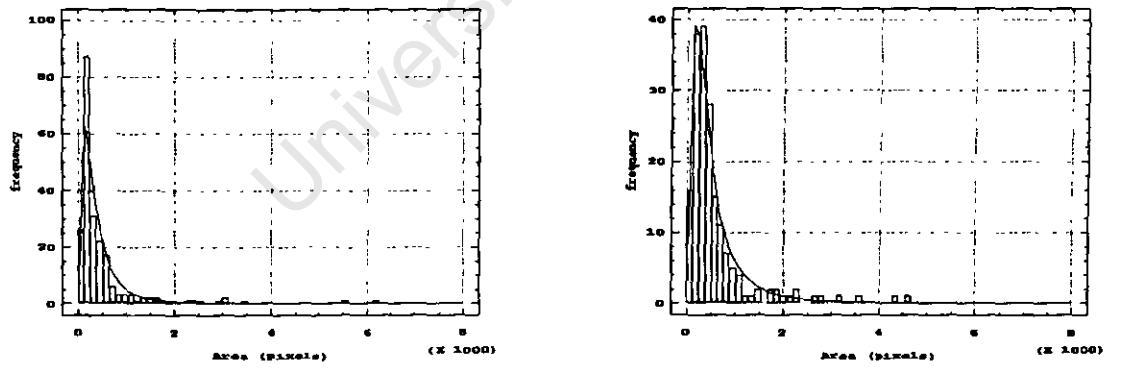
(b) Automatically segmented AOI of BUB7.

Figure 10.42: Segmented AOI's BUB7



(a) (b)

Figure 10.43: (a) Expected *Area* distribution with $\overline{x} = 492.72$, $\sigma = 753.86$, mode = 174 and skewness = 4.50. (b) Observed *Area* distribution with $\overline{x} = 602.42$, σ = 712.86, mode = 286 and skewness = 3.22.



(a) (b)

Figure 10.44: (a) Log normal fit to the expected *Area* distribution with the best fit parameters $\overline{x} = 436.56.11$ and $\sigma = 453.72$. (b) Log normal fit to the observed *Area* distribution, with the best fit parameters $\overline{x} = 585.86$ and $\sigma = 635.93$.

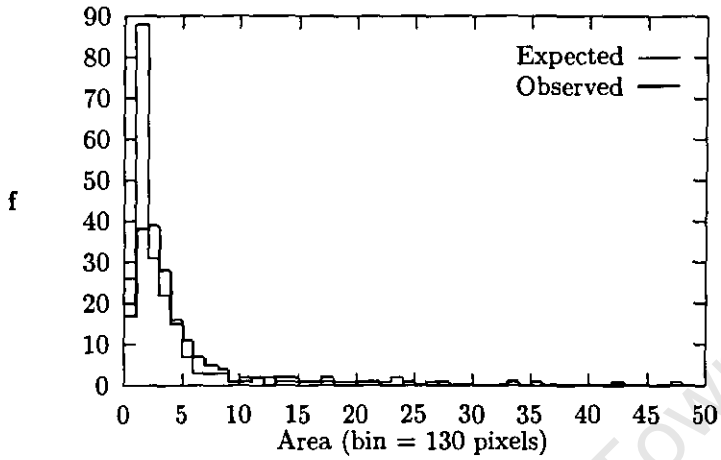


Figure 10.45: Overlaid graphs of BUB7 area distributions.

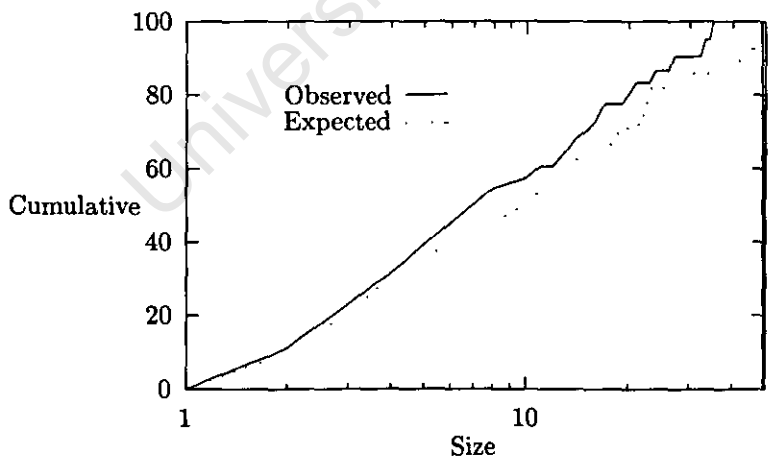
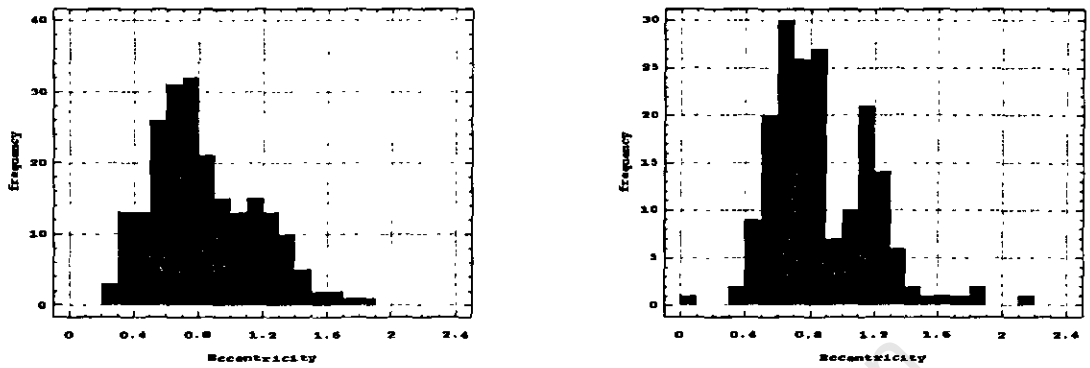
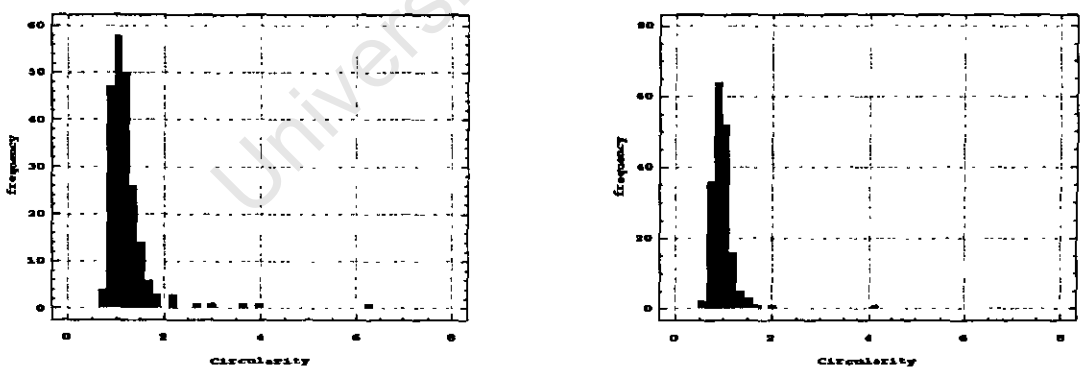


Figure 10.46: Cumlative size distributions of BUB7.



(a) (b)

Figure 10.47: (a) Expected *Eccentricity* distribution with $\overline{x} = 0.83$, $\sigma = 0.32$, mode = 0.77 and skewness = 0.60. (b) Observed *Eccentricity* distribution with $\overline{x} = 0.87$, $\sigma = 0.31$, mode = 0.86 and skewness = 0.90.



(a) (b)

Figure 10.48: (a) Expected *Circularity* distribution with $\overline{x} = 1.23$, $\sigma = 0.52$, mode = 1.14 and skewness = 5.66. (b) Observed *Circularity* distribution with $\overline{x}$ = 0.99, $\sigma = 0.30$, mode = 1.12 and skewness = 6.22.

10.1.7 Image BUB8

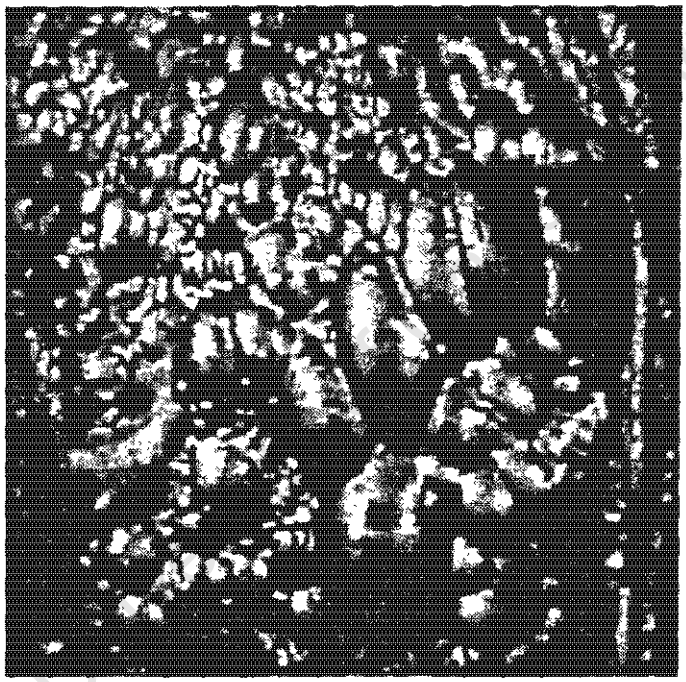
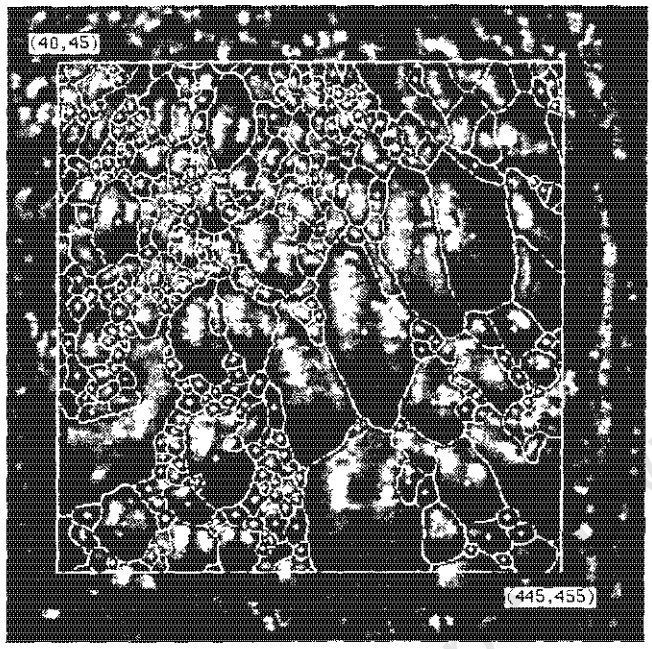
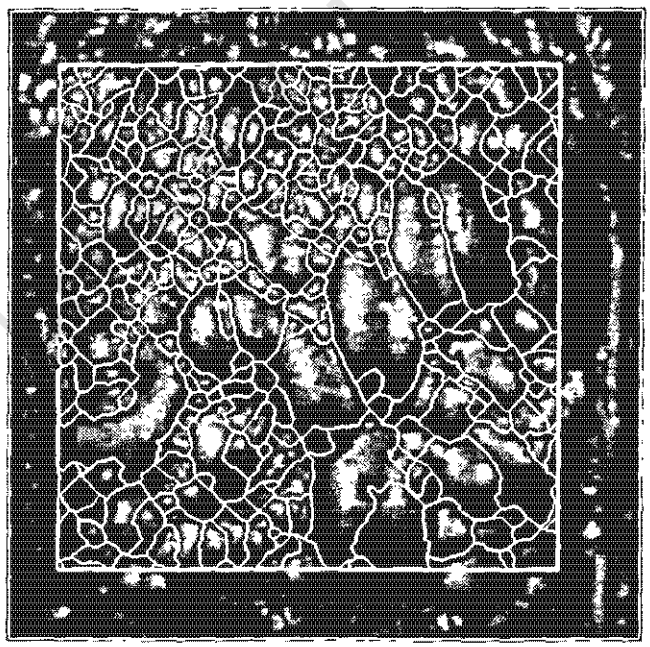


Figure 10.49: Original Image BUB8.

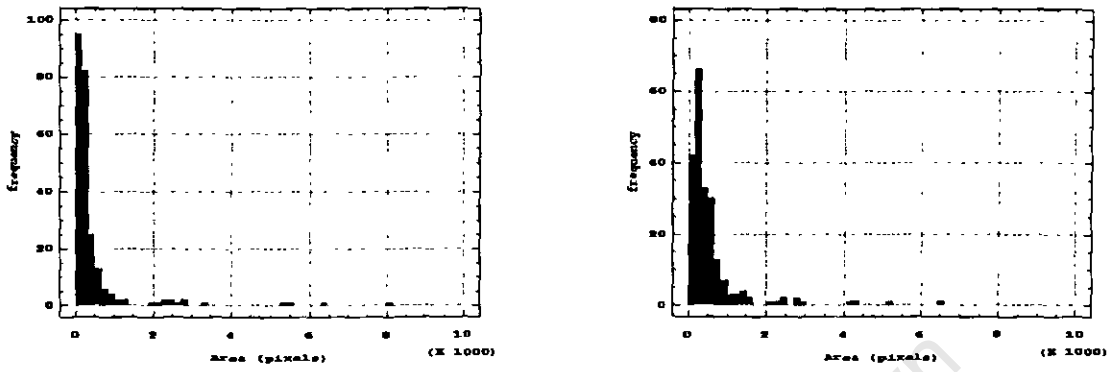


(a) Manually segmented AOI of BUB8.



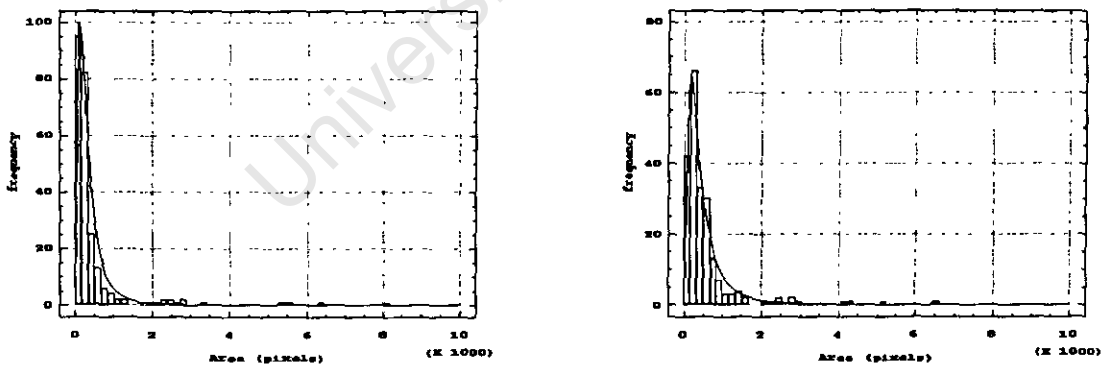
(b) Automatically segmented AOI of BUB8.

Figure 10.50: Segmented AOI's BUB8.



(a) (b)

Figure 10.51: (a) Expected *Area* distribution with $\overline{x} = 465.84$, $\sigma = 917.20$, mode = 148 and skewness = 5.33. (b) Observed *Area* distribution with $\overline{x} = 574.84$, σ = 801.44, mode = 170 and skewness = 4.24.



(a) (b)

Figure 10.52: (a) Log normal fit to the expected *Area* distribution with the best fit parameters $\overline{x} = 376.64$ and $\sigma = 389.46$. (b) Log normal fit to the observed *Area* distribution, with the best fit parameters $\overline{x} = 545.58$ and $\sigma = 626.60$.

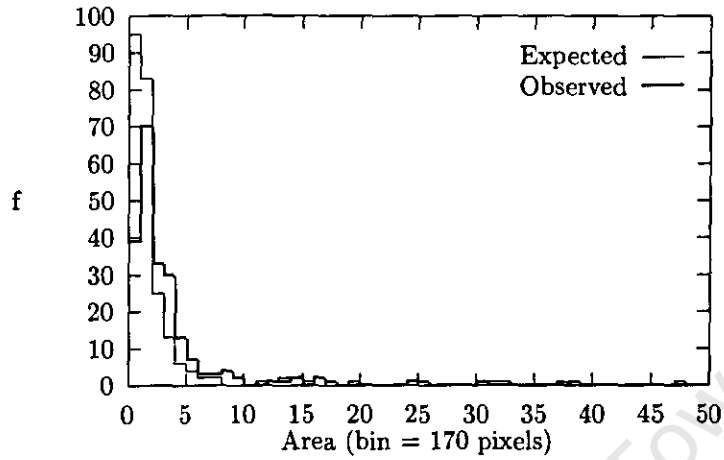


Figure 10.53: Overlaid graphs of BUB8 area distributions.

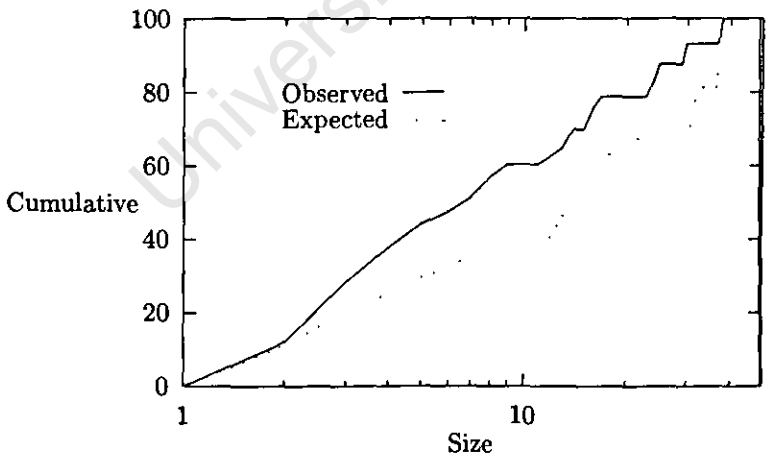
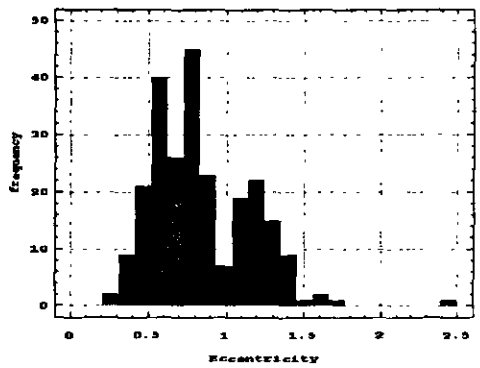
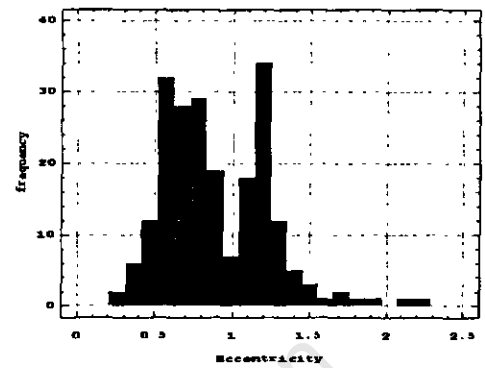


Figure 10.54: Cumlative size distributions of BUB8.

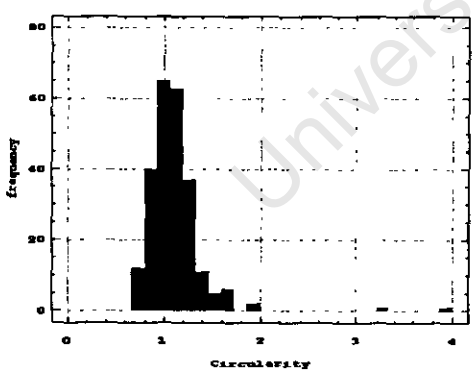


(a)

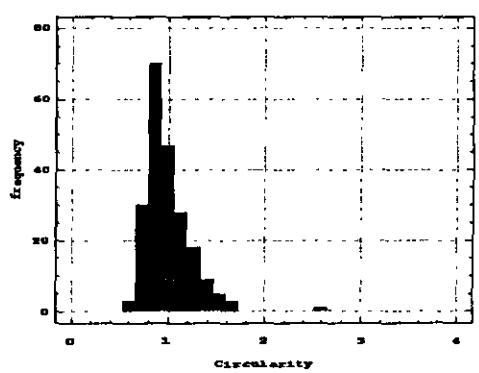


(b)

Figure 10.55: (a) Expected *Eccentricity* distribution with $\overline{x} = 0.83$, $\sigma = 0.31$, mode = 0.78 and skewness = 0.89. (b) Observed *Eccentricity* distribution with $\overline{x} = 0.90$, $\sigma = 0.34$, mode = 0.78 and skewness = 0.81.



(a)



(b)

Figure 10.56: (a) Expected *Circularity* distribution with $\overline{x} = 1.11$, $\sigma = 0.30$, mode = 1.04 and skewness = 4.83. (b) Observed *Circularity* distribution with $\overline{x}$ = 0.99, $\sigma = 0.23$, mode = 0.82 and skewness = 1.99.

10.2 Tables of the χ^2 Test Results

In this section, the results of **Chi-Squared** test to the bubble size and shape distributions of AOI's BUB2 to BUB8 are given. The purpose and the theories of **Chi-Squared** test have been introduced in Chapter 9. All of the results are given in table formats and the order is according to the order of the figures in Chapter 10.

Figure	χ^2 tests	*Conclusion*
10.3	Calculated : $\chi^2_{8,0.05}$= 24.91 Tabulated : $\chi^2_{8,0\ 05}$ =15.51	Reject H_0
10.4(a)	Calculated : $\chi^2_{9,0.05}$= 15.49 Tabulated : $\chi^2_{9,0\ 05}$= 16.91	Accept H_0
10.4(b)	Calculated : $\chi^2_{9,0.05}$= 22.12 Tabulated : $\chi^2_{9,0.05}$ = 16.91	Reject H_0

Table 10.1: χ^2 tests for Figures 10.3 and 10.4

Figure	χ^2 tests	*Conclusion*
10.7	Calculated : $\chi^2_{9,0.05}$= 18.96 Tabulated : $\chi^2_{9,0.05}$ =16.91	Reject H_0
10.8	Calculated : $\chi^2_{6,0.05}$= 22.62 Tabulated : $\chi^2_{6.05}$= 12.59	Reject H_0

Table 10.2: χ^2 tests for Figures 10.7 and 10.8.

Figure	χ^2 tests	*Conclusion*
10.11	Calculated : $\chi^2_{7,0.05}$= 27.08 Tabulated : $\chi^2_{7,0.05}$ =14.06	Reject H_0
10.12(a)	Calculated : $\chi^2_{6,0.05}$= 18.22 Tabulated : $\chi^2_{6,0.05}$=12.59	Reject H_0
10.4(b)	Calculated : $\chi^2_{8,0.05}$= 7.45 Tabulated : $\chi^2_{8,0.05}$ = 15.51	Accept H_0

Table 10.3: χ^2 tests for Figures 10.11 and 10.12.

Figure	χ^2 tests	*Conclusion*
10.15	Calculated : $\chi^2_{9,0.05}$= 10.12 Tabulated : $\chi^2_{9,0.05}$ =16.91	Accept H_0
10.16	Calculated : $\chi^2_{5,0.05}$= 149.88 Tabulated : $\chi^2_{5,0.05}$= 11.07	Reject H_0

Table 10.4: χ^2 tests for Figures 10.15 and 10.16.

Figure	χ^2 tests	*Conclusion*
10.19	Calculated : $\chi^2_{8,0.05}$= 34.6 Tabulated : $\chi^2_{8,0.05}$ =15.51	Reject H_0
10.20(a)	Calculated : $\chi^2_{10,0.05}$= 37.62 Tabulated : $\chi^2_{10,0.05}$= 18.30	Reject H_0
10.20(b)	Calculated : $\chi^2_{11,0.05}$=5.59 Tabulated : $\chi^2_{11,0.05}$ = 19.67	Accept H_0

Table 10.5: χ^2 tests for Figures 10.19 and 10.20.

Figure	χ^2 tests	*Conclusion*
10.23	Calculated : $\chi^2_{7,0.05}$=17.03 Tabulated : $\chi^2_{7,0.05}$ =14.06	Reject H_0
10.24	Calculated : $\chi^2_{5.05}$=125.37 Tabulated : $\chi^2_{5,0.05}$ = 11.07	Reject H_0

Table 10.6: χ^2 tests for Figures 10.23 and 10.24.

Figure	χ^2 tests	*Conclusion*
10.27	Calculated : $\chi^2_{7,0.05}$= 31.83 Tabulated : $\chi^2_{7,0.05}$ =14.06	Reject H_0
10.28(a)	Calculated : $\chi^2_{5,0.05}$=18.72 Tabulated : $\chi^2_{5,0.05}$ = 11.07	Reject H_0
10.28(b)	Calculated : $\chi^2_{6,0.05}$ = 4.54 Tabulated : $\chi^2_{6,0\,05}$ = 12.59	Accept H_0

Table 10.7: χ^2 tests for Figures 10.27 and 10.28.

Figure	χ^2 tests	*Conclusion*
10.31	Calculated : $\chi^2_{8,0.05}$= 47.61 Tabulated : $\chi^2_{8,0.05}$ =15.51	Reject H_0
10.32	Calculated : $\chi^2_{7,0.05}$=17.8 Tabulated : $\chi^2_{7,0.05}$= 14.06	Reject H_0

Table 10.8: χ^2 tests for Figures 10.31 and 10.32.

Figure	χ^2 tests	*Conclusion*
10.35	Calculated : $\chi^2_{6,0.05}$=124.46 Tabulated : $\chi^2_{6,0.05}$ =12.59	Reject H_0
10.36(a)	Calculated : $\chi^2_{6,0.05}$ = 33.07 Tabulated : $\chi^2_{6,0\ 05}$= 12.59	Reject H_0
10.36(b)	Calculated : $\chi^2_{7,0.05}$ = 6.62 Tabulated : $\chi^2_{7,0.05}$ = 14.06	Accept H_0

Table 10.9: χ^2 tests for Figures 10.35 and 10.36.

Figure	χ^2 tests	*Conclusion*
10.39	Calculated : $\chi^2_{7,0\ 05}$ = 27.71 Tabulated : $\chi^2_{7,0.05}$ = 14.06	Reject H_0
10.40	Calculated : $\chi^2_{6,0.05}$ = 17.64 Tabulated : $\chi^2_{6,0\ 05}$= 12.59	Reject H_0

Table 10.10: χ^2 tests for Figures 10.39 and 10.40.

Figure	χ^2 tests	*Conclusion*
10.43	Calculated : $\chi^2_{6,0.05}$=42.98 Tabulated : $\chi^2_{6,0\ 05}$ =12.59	Reject H_0
10.44(a)	Calculated : $\chi^2_{7,0.05}$ = 32.66 Tabulated : $\chi^2_{7,0\ 05}$=14.06	Reject H_0
10.44(b)	Calculated : $\chi^2_{9,0.05}$ = 8.73 Tabulated : $\chi^2_{9,0.05}$ = 16.91	Accept H_0

Table 10.11: χ^2 tests for Figures 10.43 and 10.44.

Figure	χ^2 tests	*Conclusion*
10.47	Calculated : $\chi^2_{10,0.05}$= 24.17 Tabulated : $\chi^2_{10,0.05}$ = 18.30	Reject H_0
10.48	Calculated : $\chi^2_{6,0\ 05}$ = 18.49 Tabulated : $\chi^2_{6,0\ 05}$ = 12.59	Reject H_0

Table 10.12: χ^2 tests for Figures 10.47 and 10.48.

Figure	χ^2 tests	*Conclusion*
10.51	Calculated : $\chi^2_{7,0.05}$=50.93 Tabulated : $\chi^2_{7,0\ 05}$ =14.06	Reject H_0
10.52(a)	Calculated : $\chi^2_{5,0.05}$ = 28.27 Tabulated : $\chi^2_{5,0\ 05}$ = 11.07	Reject H_0
10.52(b)	Calculated : $\chi^2_{7,0.05}$=13.59 Tabulated : $\chi^2_{7,0\ 05}$ = 14.06	Accept H_0

Table 10.13: χ^2 tests for Figures 10.51 and 10.52.

Figure	χ^2 tests	*Conclusion*
10.55	Calculated : $\chi^2_{11,0.05}$=18.49 Tabulated : $\chi^2_{11,0.05}$ =19.67	Accept H_0
10.56	Calculated : $\chi^2_{6,0.05}$ = 104.01 Tabulated : $\chi^2_{6,0\ 05}$ = 12.59	Reject H_0

Table 10.14: χ^2 tests for Figures 10.55 and 10.56.

10.3 Discussion

10.3.1 Size Distributions

Refer to the segmentation results of the BUB2 to BUB4 group of images. The automatically segmented bubble boundaries were well defined according to the basic concept of watershed theory - that is, for each highlight there is one watershed region. This observation applies to all three images. Since these are sequence images of BUB1, and also under the same illumination conditions, the processing parameters are almost the same as those for image BUB1. Taking the manually segmented images as the benchmark, the graphs of cumulative size distributions and overlaid size distributions show that the automatic segmentation results are close to these of the manually segmented images.

Refer now to the segmentation results of images BUB5 to BUB8. From the surface of the original images, the dual highlights are clearly visible in the comparatively big and middle size bubble regions. This caused a watershed segmentation problem such that the regions between the highlights are erroneously identified as bubble boundaries. Also, another problem occurred at the shadow regions which caused gradient calculations to miss their directions and the resulting bubble boundaries were ill defined. From the figures of overlaid segmentation results, graphs of cumulative size distributions and overlaid size distributions, the effect could be seen clearly whereby, extra segmentation lines appeared in automatically segmented images when compared with manually segmented images. For those comparativly big size bubbles, the extra lines could be deleted by applying the "variance" post-processing method. As a result, the detected bubbles and mean bubble sizes compare favourably with those of the manually segmented images.

However, it may be seen from all the χ^2 results that most of the tests failed. First of all, the tests were performed for a 95% confidence level. This is a high level, and if a level of 90% had rather been used, it is clear that most tests would have been passed. Further, one must not judge the entire algorithm and its performance simply on these χ^2 test results, as this would be meaningless. Visually, the automatically and manually segmented images seem very similar, as do the distribution plots. Finally, the main reason for the discrepancies is that the manually segmented images have been segmented to a high level of detail, especially with regard to the small bubbles. On the other hand, due to the Gaussian filtering of the images, many small bubble areas were lost and thus do not feature in the results. This factor immediately biases the automatical segmentation results away from acceptable χ^2 test results. If it were possible to repeat all the processing and analyses on a microscopic scale, such that both the manual and automatic segmentation focussed only on middle and large size bubbles, the results of the

χ^2 tests would be far more pleasing. As mentioned before, it is the segmentation of middle and large size bubbles that is of importance to the mineral extraction industry.

Table 10.15 is a summarised table of manual and automatic segmentation results with detected bubble numbers and determined mean bubble sizes for all the processed images. It may be seen that in most cases the observed total number of segmented bubbles, N_o, is lower than the expected counterpart, N_e. Correspondingly, the observed mean bubble sizes are greater than expected. Thus, if one performed the χ^2 tests on more fair grounds, where the expected results did not take into account the small bubbles on each image, the tests could well be expected to pass, even at a 95% confidence level.

	Manual results		Automatic results	
Image	N_E	Mean size	N_O	Mean size
BUB1	172	625.16	174	618.09
BUB2	189	609.73	171	680.67
BUB3	193	510.23	171	559.76
BUB4	194	545.11	192	543.91
BUB5	217	539.02	217	538.27
BUB6	247	446.43	231	508.49
BUB7	216	492.72	181	602.42
BUB8	243	465.84	214	574.84

Table 10.15: Summarised table of manually and automatically determined mean bubble sizes, with corresponding number of detected bubbles.

10.3.2 Circularity Distributions

Although the graphs of circularity distributions were similar for the automatically and manually segmented images, the graphs were shifted for reasons discussed in Chapter 9. The χ^2 test of both the manually and automatically segmented results failed for both groups of images. As a result, it could be said that the circularity factor is not the optimum descriptor of bubble shape.

10.3.3 Eccentricity Distributions

The results of χ^2 test for the two groups of images show that some of the distributions were accepted as similar. The eccentricity factor which is computed with the use of the VSHAPE program, which in turn calculates eccentricity using moments, is thus a more reliable and robust manner of quantifying bubble shape.

Chapter 11

Conclusions and Recommendations

A need has arisen in the mining industry for automatical measurement of the size and shape of bubbles that constitute surface froth structures in flotation cells. In this dissertation, computer vision has been put forward as a possible solution, together with a suitable method to achieve the required accuracy of segmentation.

11.1 Conclusions

This research work and the results presented here show that the watershed boundary technique provides a reliable method for the segmentation of surface froth structures. Minor errors which occur do not significantly influence the statistical parameters determined from the segmented images. The results show this beyond doubt.

The accuracy of the watershed segmentation process is to an extent dependent upon the camera and lighting configurations used to view the scene. The segmented images show that, if more than one lighting source is used to illuminate the surface froth, particularly for middle size to large bubbles, the watershed method leads to more inaccurate segmentation than found when only one light source is used.

The run-time performance of the method presented here is a significant improvement over the processing time required for more sophisticated algorithms such as those used by Symonds [21]. The performance of a full processing session takes less than 5 minutes. Therefore, it is possible for the watershed method to be used

in real-time segmentation systems.

The statistical comparison of the expected and observed distributions of bubble size, circularity and eccentricity revealed that the machine vision system does not mimic the human visual process exactly, but does however still achieve accurate segmentation which is favourable with regard to the expected results. In the final analysis, the computed parameters of bubble sizes and the measured bubble eccentricity, can be used as robust and reliable measurements to describe the froth surface and bubble shapes. These parameters may thus be used to describe quantitatively the mineral grade and recovery values and be used in the optimisation of control of industrial flotation cells.

11.2 Recommendations

Future research work should be able to provide even better performance, both in terms of processing speed and accuracy if the following factors are looked into:

1. Since the lighting and camera placement configuration play an important role in this system, it is required to investigate the effects thereof more closely so as to obtain an optimal camera and lighting layout. As a start, one lighting source is recommended; as opposed to two or more such sources.
2. The code used for image labelling, as described in Chapter 9, could be rewritten to specifically handle byte images. This would result in the Khoros *VLABEL* routine becoming redundant and could thus reduce runtime.
3. The present algorithm has been coded as a prototype and also for ease of readability. If it were to be optimised, witten in a more linear fashion (by loop and recursion unrolling) a significant improvement in speed may be expected. Further, the code may be ported to a lower level platform and embedded in hardware, thus ensuring true real-time processing capabilities.
4. Lastly, someone with a strong mathematical and computing background may well be able to reduce and optimise the theorethical design of the watershed segmentation algorithm.

Appendix A

Khoros Workspaces

The **cantata** workspace for the processing of image groups BUB1 to BUB4 and BUB5 to BUB8 are given here. All **khoros**, **cantata** and 'C' sources for the processing may be obtained from the author.

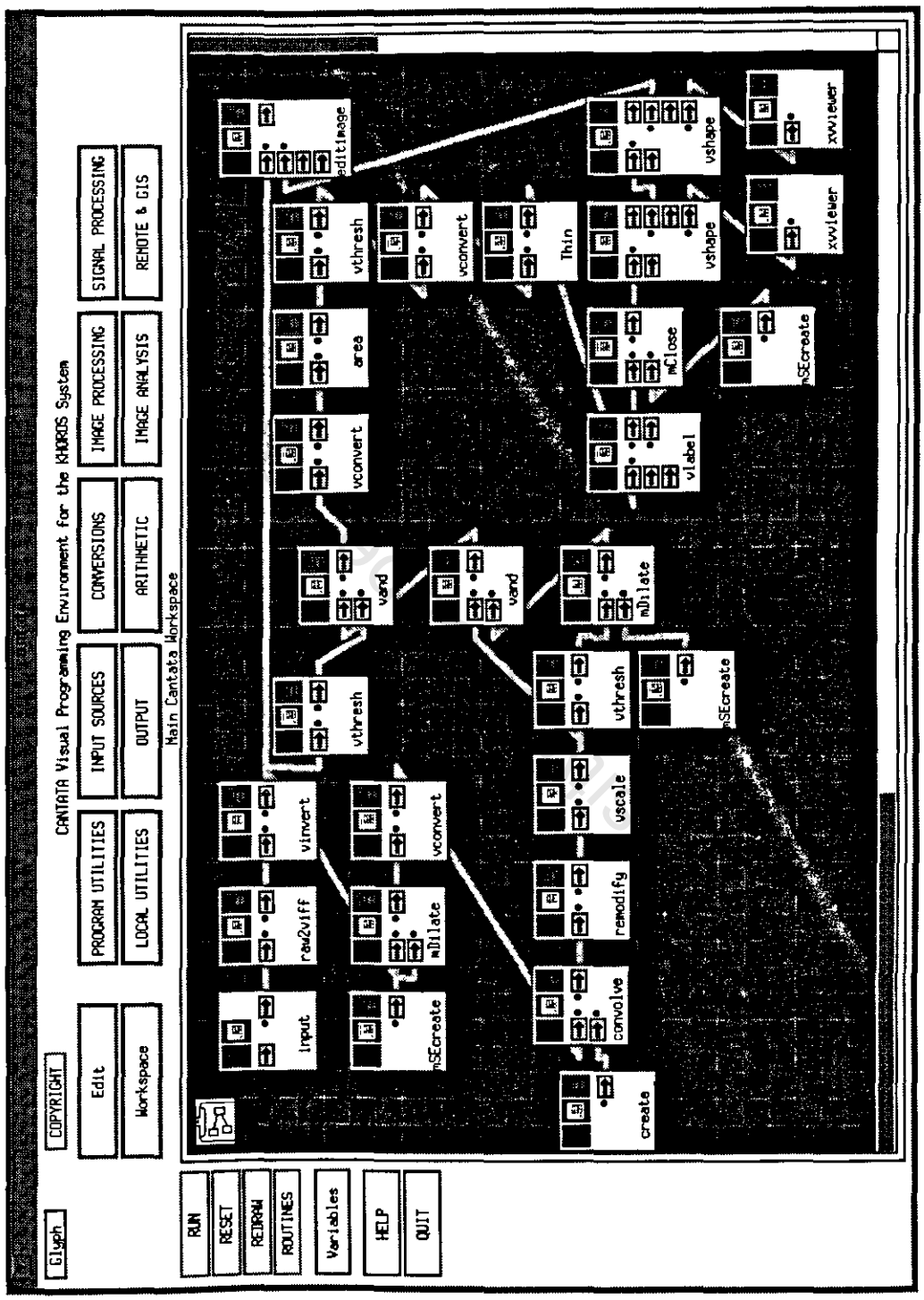


Figure A.1: Cantata workspace for the processing of images BUB1 to BUB4.

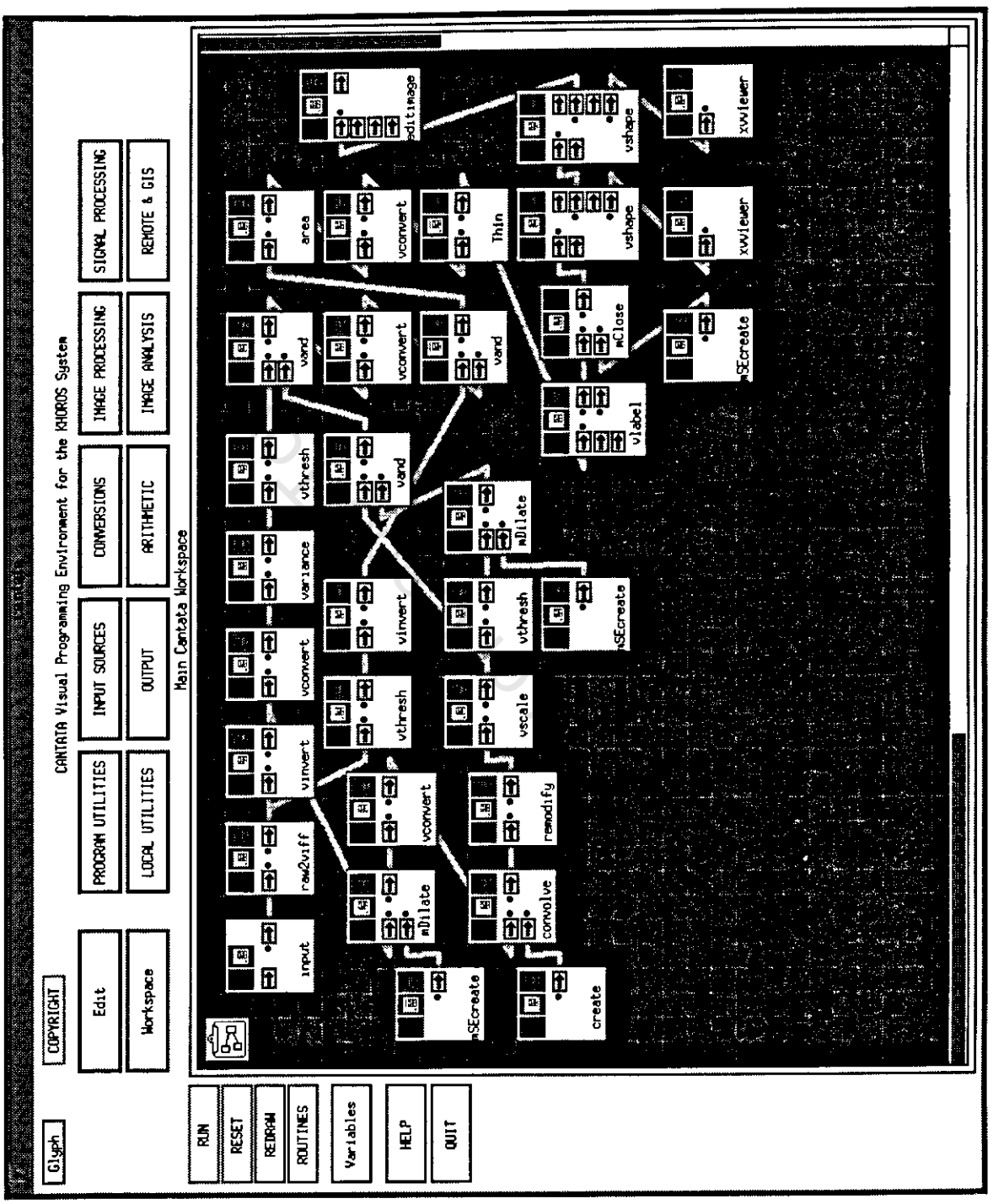


Figure A.2: Cantata workspace for the processing of images BUB5 to BUB8.

Bibliography

[1] Ming-Hua Chen and Ping-Fan Yan. A multiscaling approach based on morphology. *IEEE Transactions on Pattern Analysis and Machine Intelligence.*, 11(7):694–700, July 1989.

[2] Edward R. Dougherty. *An Introduction to Morphological Image Processing.* SPIE- The International Society for Optical Engineering, 1992.

[3] Woodburn E.T., Stockton J.B., and Robbins D.J. Vision-based characterization of three-phase froths. *International Colloquium Developments in Froth Flotation, South African Institute of Mining and Metallurgy, Gordon's Bay, South Africa.*, 1:1–30, 1989.

[4] John M. Gauch and Stephen M. Pizer. Miltiresolution analysis of vertex curves and watershed boundaries. *SPIE Medical Imageing VI: Image Processing*, 1652:226–240, 1992.

[5] John M. Gauch and Stephen M. Pizer. Multiresolution analysis of ridges and valleys in grey-scale images. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(6):635–646, June 1993.

[6] Robert M. Haralick and Linda G. Shapiro. *Computer And Robot Vision.* Addison-Wesley Publishing Compay, 1993.

[7] Kenneth J. Ives, editor. *The Scientific Basis of Flotation.* NATO ASI Series, 1984.

[8] The Khoros Group, Dept. of Electrical and Computer Engineering, University of New Mexico, Albuquerque, NM 87131. *Khoros Manuals*, 1991.

[9] Martin D. Levine. *Vision In Man and Machine.* McGraw-Hill Book Company, 1985.

[10] David G. Lowe. Organization of smooth image curves at multiple scales. *Second International Conference on Computer Vision (Florida, USA)*, 1988.

[11] Petros Maragos. Morphological filters-part I : Their set-theoretic analysis and relations to linear shift-invariant filters. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, ASSP-35(8):1153–1169, 1987.

[12] Petros Maragos and Ronald W. Schafer. Morphological systems for multidimensional signal processing. *Proceedings of the IEEE*, 78:690–710, 1990.

[13] J. S. Milton and Jesse C. Arnold. *Probability and Statistics in the Engineering and Computing Sciences.* McGraw-Hill Book Company, 1986.

[14] D. W. Moolman, C. Aldrich, J. S. J. van Deventer, J. J. Eksteen, W. W. Stange, P. Marais, C. Goodall, and R. S. Veitch. On-line image analysis to improve industrial flotation plant performance. *IFAC MMM95*, 1995.

[15] C. T. O'Connor, E. W. Randall, and C. M. Goodall. Measurement of the effects of physical and chemical variables on bubble size. *International Journal of Mineral Processing*, 28:139–149, 1990.

[16] L. M. Popplewell and M. Peleg. Calculating the beta function from three common particle size distributions. *American Institute of Chemical Engineers (AIChE)*, 35(2):347–349, February 1989.

[17] H. Rneave and P. L. Worthington. *Distribution-Free Tests.* Unwin Hyman Ltd, 1988.

[18] Azriel Rosenfeld and Avinash C. Kak. *Digital Picture Processing*, volume 2. Academic Press, 1982.

[19] S. R. Sternberg. Grayscale morphology. *Computer Vision, Graphics, and Image Processing*, 35:333–335, 1986.

[20] T. V. Subrahmanyam and Eric Forssberg. Froth stability, particle entrainment and drainage in floatation - a review. *International Journal of Mineral Processing*, 23:33–53, 1988.

[21] Paul James Symonds. The investigation of the characterisation of flotation froths and design of a machine vision system for monitoring the operation of a flotation cell ore concentrator. Master's thesis, University of Cape Town, 1992.

[22] Luc Vincent. Morphological grayscale reconstruction in image analysis: Applications and efficient algorithms. *IEEE TRANSACTIONS ON IMAGE PROCESSING*, 2(2):176-210, April 1993.

[23] Luc Vincent and Pierre Soille. Watersheds in digital spaces: An efficient algorithm based on immersion simulations. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(6):583–597, June 1991.