APPLICATION OF X-RAY TOMOGRAPHY TO MEASUREMENT OF FRACTURES IN ROCKS

A REPORT
SUBMITTED TO THE DEPARTMENT OF PETROLEUM ENGINEERING
OF STANFORD UNIVERSITY
IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE DEGREE OF
MASTER OF SCIENCE

By
Meiqing He
June 1998

I certify that I have read this report and that in my opinion it is fully adequate, in scope and in quality, as partial fulfillment of the degree of Master of Science in Petroleum Engineering.

Roland N. Horne
(Principal advisor)

Acknowledgments

This research was conducted under the guidance of Dr. Cengiz Satik and Professor Roland Horne. I would like to extend my deep appreciation for their constant help and patience throughout this research.

Special thanks to my fellow students in Geothermal group for the constant learning experience with them and the help from them, to Liping Wei for bringing me the idea of the active contour model, and to Stephanie Bertels for sharing her experimental data.

The project was funded by the Department of Energy under the grant DE-F607-95ID13370 to the Stanford Geothermal Program.

Finally, greatest thanks to my family for their support and understanding.

Abstract

The aim of this research was to identify and characterize the small fractures in geothermal rocks using the X-Ray CT scanner. Geothermal rocks typically have small porosity and a large number of tiny fractures scattered in a dense rock matrix. The small fractures are difficult to measure accurately by CT imaging when their aperture is of the order of or even smaller than the CT scanner's resolution. Previous research in the area of fracture calibration using CT imaging was focused on artificial fractures with apertures larger than the CT resolution or fractures existing in rock lighter and more homogeneous than geothermal rock.

In this research we focused on the detection and calibration of natural fractures in samples of Geysers core. We investigated the basic theory of the CT technique. Previous results in the area of CT measurement were enhanced by incorporating image-processing techniques for image enhancement and natural fracture detection. We proposed a systematic procedure for fracture identification, fracture aperture calibration and saturation calculation in an identified fracture region. Specifically, we used edge detection methods to obtain the edge map, the Hough transform to estimate the fracture orientation, and the Active Contour approach to refine the edge map and to determine a well-connected fracture region. The detection quality of the fractured region was significantly improved.

Contents

Acknowledgments iii

Abstract iv

Table of Contents v

List of Tables viii

List of Figures ix

1 Introduction 1

2 Theoretical Background 4

2.1 CT Technology and Measurement 4

2.1.1 Measuring Porosity and Saturation Using CT 6

2.2 Edge Detection 7

2.3 Denoising 9

2.3.1 Wavelet Transform 9

2.3.2 Soft-Thresholding 11

2.4 Hough Transform 12

2.5 Active Contour Model 14

3 Calibration Experiment 17

4 Results **27**
4.1 Denoising Core CT Image 27
4.2 Hough Transform 29
4.3 Finding the Contour Using Active Contour Model 30
4.3.1 Finding the Artificial Fracture Contour 30
4.3.2 Finding the Contour of Natural Fractures 30

5 Conclusions and Discussions **36**

Bibliography **38**

Appendix **40**

A Core Functions **41**
A.1 Hough Transform 41
A.2 Solving the Snake Model 43
A.3 Soft-Thresholding Denoising 46

B Utility Functions **49**
B.1 Connect Two Points with a Straight Line 50
B.2 Selection of a Circular Region out of an Image 54
B.3 Selection of a Vector from a Circular Buffer 55
B.4 Determination of Adjacent Points 56
B.5 Marking the Region inside a Contour 57
B.6 Difference between Two Vectors 60
B.7 Calculation of Manhanttan Distance 61
B.8 Matrix A Generator 62
B.9 Components Labeling 63
B.10 Table Look-up 66
B.11 Image Data Conversion 67

C Programs Used for Calibration Purpose **68**
C.1 Application of the Hough Transform to the Artificial Fracture 69

C.2 Integration of the Missing CT 73
C.3 Least Square Fitting 78
C.4 Example of Setting Initial Contour and Solve the Snake 80

D Programs used for Geysers Sample **83**
D.1 Calculation of Fracture Aperture in Geysers Sample 84
D.2 Displaying the Contours 90

List of Tables

List of Figures

2.1 Roberts operator 8
2.2 Sobel operator 8
2.3 Scheme of wavelet decomposition 10
2.4 Comparison between clean block signal, noisy block signal and denoised block signal 12
2.5 (a) Normal representation of a line. (b) Quantization of $\rho\theta$ plane into cells. 13

3.1 Experimental setup 17
3.2 Fracture: aperture = 0.04 in 18
3.3 Fracture: aperture = 0.02 in 19
3.4 Fracture: aperture = 0.012 in 20
3.5 Fracture: aperture = 0.008 in 21
3.6 Fracture: aperture = 0.004 in 22
3.7 Fracture: aperture = 0.002 in 23
3.8 Fracture: aperture = 0.001 in 24
3.9 Minimal CT over the gap vs. aperture 25
3.10 Relationship between integrated missing CT and aperture: y=1.398e5*x-194.5402 26

4.1 Coefficient pattern after one level of wavelet decomposition 27

4.2 Denoising effect. (a) Original core image. (b) Edge map using Roberts gradient operator (thr=20) on the original image. (c) Denoised core image. (d) Edge map using Roberts gradient operator (thr=20) on the denoised image. 28
4.3 (a) Edge map of sample of basalt core with a fracture of 0.04in. (b) Hough transform corresponding to (a). 29
4.4 Finding a contour using Active Contour Model (a) Initial contour of fracture. (b) Locus of snake after one iteration. (c) Stable locus of snake after 300 iterations. (d) Final contour of fracture. 31
4.5 Case study of setting initial contour not close enough to object of interest. (a) Initial contour of fracture. (b) Locus of snake after one iteration. (c) Stable locus of snake after 300 iterations. (d) Final contour of fracture. 32
4.6 Finding fracture contours using snake (a) Original core image. (b) Edge map using Sobel gradient operator (thr=20) on the original image. (c) Connected contours found by snake. 33
4.7 (a) Contour of a typical fracture in Geysers core. (b) Aperture calculation on the contour shown in (a). 34
4.8 Aperture distribution in Geysers sample 35

Section 1

Introduction

Enormous effort has been applied in advancing the understanding of fluid flow in heterogeneous porous media. The effect of fracture aperture variability on water or steam distribution, especially under geothermal reservoir conditions, is still unknown. Many concerns related to the necessity of evaluating the heterogeneity of rocks arises from previous experiments. We found that adsorption and the distribution of water and steam in a network consisting of tiny fractures are, in fact, not negligible in mass transfer processes. Geothermal rocks usually have dense matrix within which tiny fractures are scattered. Therefore fracture aperture has significant impact on mass transfer in geothermal rock and it is important to characterize the fracture geometry.

X-ray computerized tomography (CT) is a non-destructive measurement technology that has been applied increasingly in the study of rock properties and fluid flow in porous media, Huang *et al.* (1995), Bonner (1997), Akin (1996) and Johns (1991). In a CT image, the CT number represents the intensity which resulted from the attenuation of photons. The attenuation is correlated to atom number or density of material. Therefore the CT number is proportional to density. In the two-dimensional intensity image, a fracture is shown as dark region if the aperture is not too small. If we consider a perpendicular line across the fracture and plot the CT number along this line in a two-dimensional graph, we get a profile of the CT response. We will be able to find a dip in that profile corresponding to the fracture. It was found that neither the depth nor the width of the dip can be used to define the aperture size

over a wide range of scales. In particualr, when it comes to using CT to calibrate a fracture in geothermal rock, the accuracy of measurement becomes more crucial. The aperture of some of the fractures in geothermal rock is of the order of the CT scanner resolution or even smaller. The resolution of CT scanner we are using is $300\mu m$. Therefore some fractures may be sub-pixel-sized. Johns(1991) proposed a quantitative calibration between the gap size and the CT response. In his study, the cumulative CT response was correlated to gap size. Our fracture calibration methods were based upon this approach with improvement in automation of natural fracture detection and calibration using image processing techniques.

In order to obtain the accumulation over the gap, the starting point and ending point of the integration need to be identified. In a two-dimensional intensity image the fracture is distinguished by a discontinuity in intensity. The boundary of a fracture usually is acquired by an edge detection process. However, the edge segments generated by edge detection are seldom well-connected. This means there is no guarantee of finding the integration boundaries. To solve this problem, we introduced the Active Contour model to find the contour of the fracture. The Hough transform was used to find the orientation of a fracture and the direction of integration. We will discuss the details of these methods in this report.

Chapter 2 elucidates the theoretical background of the approaches used to detect and measure natural fractures. It is divided into five sections. Section 2.1 briefly reviews the theoretical background of the CT technology and discusses the issues that limit the resolution of a CT scanner. Section 2.2 focuses on edge detection. Section 2.3 discusses the necessity of denoising prior to edge detection and soft-thresholding denoising using wavelet transform. Section 2.4 discusses the necessity of edge linking and the functionality of the Hough transform for detecting the line features in an image. Section 2.5 introduces the Active Contour approach as an elegant way to refine the edge map.

Chapter 3 describes the necessary experiments in the entire calibration procedure and discusses the issues related to obtaining good quality core CT images.

Chapter 4 presents the results of application of denoising, edge detection, edge linking methods to images of cores.

Chapter 5 summarizes the general procedures for detecting natural fractures and calibrating them. This chapter also includes suggestions about utilizing results from this research in other experiments and research projects.

Section 2

Theoretical Background

2.1 CT Technology and Measurement

Computer-Aided Tomography (CT) can produce a higher resolution internal image of an object than conventional X-ray methods. The technique has been applied increasingly in the study of rock properties. Unit elements in the two-dimensional image matrix generated by the X-ray CT scanner are referred to as CT numbers. Each CT number corresponds to the average attenuation coefficient (μ_{voxel}) within a voxel. The object is divided into voxels which are cubic and can be described as pixels with a thickness. The thickness of voxel is equal to the slice thickness (X-ray beam thickness) and the square dimension of a voxel is defined by the length of the detectors divided by the dimension of the image matrix which results from the sampling rate. Prior to display of a CT image, it is conventional in the medical field to normalize the X-ray attenuation data to the value of water in the following fashion:

$$CT = \frac{2048(\mu_{voxel} - \mu_{water})}{\mu_{water}} \tag{2.1}$$

where μ is the X-ray attenuation coefficient, and CT is the CT number, whose unit is called the Hounsfield. This results in CT numbers of -1024 Hounsfield for air, 0 Hounsfield for water, and 1024 Hounsfield for bone. The relationship between detected intensity $I_d(x,y) = \int I_0(\epsilon) \exp\left[-\int \mu(x,y,z,\epsilon)\, dz\right] d\epsilon$ and incident intensity

$I_d(x,y) = \int I_0(\epsilon) \exp\left[-\int \mu(x,y,z,\epsilon)\, dz\right] d\epsilon$ can be expressed as

$$I_d(x,y) = \int I_0(\epsilon) \exp\left[-\int \mu(x,y,z,\epsilon)\, dz\right] d\epsilon \tag{2.2}$$

where $I_0(\epsilon)$ is the incident X-ray beam intensity as a function of the energy per photon ϵ and $\mu(x,y,z,\epsilon)$ is the linear attenuation coefficient at each region of the object. The μ of all materials depends on the photon energy of the beam and the atomic number of the elements in the material. The decreasing of intensity of the X-ray beam after traversing an object is described by the photon absorption or scattering. Attenuation mechanisms are composed of coherent or Rayleigh scatter, photoelectric absorption, and Compton scatter in the diagnostic range, below 200 kev (kiloelectron volts).

$$\mu = \mu_R + \mu_P + \mu_C \tag{2.3}$$

where R, P, and C refer to Rayleigh (coherent) scattering, photoelectric effect and Compton scattering respectively. Photoelectric absorption dominates in materials with higher atomic number. The attenuation coefficient undergoes a sharp increase in the energy region corresponding to the K shell. For high-atomic-number material, the K absorption edge occurs within the spectrum of interest. On the other hand, Compton scatter dominates in low-atomic-number material, such as water and soft tissue (Macovski, 1983). With respect to materials such as water, steam and rock, only the Compton effect and photoelectric adsorption are expected to be dominant while coherent scatter only happens when the X-ray energy is of the order of a few kev which is not within the energy level range of our experiment. The Compton attenuation contribution μ_C to the total attenuation is dependent on energy E, but independent of atomic number. The photoelectric component is dependent on both atomic number and energy level. Elements in Eq. 2.3 take the form

$$\mu = \rho N_g \left\{ f(\epsilon) + C_R \frac{\overline{Z}_r^k}{\epsilon^l} + C_P \frac{\overline{Z}_p^m}{\epsilon^l} \right\} \tag{2.4}$$

where ϵ is the photon energy in kev, N_g is the electron mass density in electron per gram as given by

$$N_g = N_A \sum_i w_i \frac{Z_i}{A_i} \tag{2.5}$$

w_i is the proportion by weight of the i th constituent, and $\overline{Z}_r$ and $\overline{Z}_p$ are the effective atomic number as given by

$$\overline{Z}_r = \left(\sum_i \alpha_i Z_i^k \right)^{1/k} \tag{2.6}$$

$$\overline{Z}_p = \left(\sum_i \alpha_i Z_i^m \right)^{1/m} \tag{2.7}$$

In Eqs. 2.6 and 2.7 k, m, l, and n are empirically determined exponent, and α_i is the electron fraction of the i th element given by

$$\alpha_i = \frac{N_{gi}}{\sum_i N_{gi}} = \frac{N_A w_i \left(\frac{Z_i}{A_i} \right)}{\sum_i N_{gi}} \tag{2.8}$$

A_i is the atomic mass of i th constituent, and N_A is Avagadro's number. The Compton scattering function $f(\epsilon)$, which is independent of the atomic number, can be described by the Klein-Nishina function (Macovski, 1983):

$$f_{kn}(\alpha) = \frac{1+\alpha}{\alpha^2} \left[\frac{2(1+\alpha)}{1+2\alpha} - \frac{1}{\alpha} \ln(1+2\alpha) \right] + \frac{1}{2\alpha} \ln(1+2\alpha) - \frac{1+3\alpha}{(1+2\alpha)^2} \tag{2.9}$$

$$\alpha = \epsilon / 510.975 kev \tag{2.10}$$

These equations provide us a way to identify material in a porous medium. As we can see from Eq. 2.4, the attenuation coefficient is linearly dependent on the density of the material.

2.1.1 Measuring Porosity and Saturation Using CT

Conventionally, porosity and saturation are obtained from CT measurement after saturating a porous medium with contrast agents, e.g. water and air are most commonly used. There are two sets of CT scans needed to calculate porosity. One is the dry scan (CT_{dry}), which is conducted when the porous medium is fully saturated with air; the other is the wet scan (CT_{wet}), which is conducted after the porous medium has been saturated with water. Then the porosity is obtained by

$$\phi = \frac{CT_{wet} - CT_{dry}}{CT_{water} - CT_{air}} \tag{2.11}$$

To calculate saturation another scan ($CT_{p_1p_2}$) is required, which is performed when two phases, p_1 and p_2, are coexisting in the porous medium. Saturation is given by

$$S_{p_1} = \frac{CT_{p_1p_2} - CT_{p_2r}}{CT_{p_1r} - CT_{p_2r}} \tag{2.12}$$

$$S_{p_2} = 1 - S_{p_1} \tag{2.13}$$

where r refers to rock.

In such a measuring process one must make sure the core is fully saturated with contrast agent, which usually requires a long experimental time in low permeability rocks. In addition, this process may not be fully appropriate for the determination of porosity in fractured rocks, since the saturating fluids are not distributed uniformly. In fractured rocks, we expect to require a different procedure that identifies the fractures specifically.

2.2 Edge Detection

A fracture in a two-dimensional image is identified as a discontinuity in intensity (see Figure 3.2(a)). In a CT profile a fracture is indicated by a valley (see Figure 3.2(d)). To detect such features we usually rely on edge detection methods.

Edge detection methods can be divided into two categories: finding the local maximum from first order gradients, or finding the zero-crossing from the second derivative.

From the line profile of core CT number we can tell the existence of a fracture by a valley in the profile. In the two-dimensional density image, a fracture is shown as a dark region. In Figure 4.2(a), four main fractures can be identified as valleys. Commonly the gaps are filled with air. Note that the CT numbers inside the gaps do not give the typical CT number for air, which is -1024, as we expect. Instead, the CT numbers in the valleys indicate values around 1700. This is an effect often referred to as dispersion of the CT response, which is caused by the finite beam width and oversampling. Usually an infinitesimal point can appear within four to nine voxels surrounding the voxel in which the point is located. Also if the voxel contains more

$$\textit{Row Gradient} \quad \begin{bmatrix} 0 & 0 & -1 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix} \qquad \textit{Column Gradient} \quad \begin{bmatrix} -1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$$

Figure 2.1: Roberts operator

$$G_x\ Mask \quad \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix} \qquad G_y\ Mask \quad \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix}$$

Figure 2.2: Sobel operator

than one component, the resulting attenuation response is determined by the volume fraction of each component and the attenuation response for each pure component. The integrated missing mass (CT number) corresponds to the gap size linearly. Thus a general approach is to run a calibration using homogeneous material with an artificial fracture. The fracture aperture is calibrated mechanically before being scanned in the CT scanner. The correlation between the integrated CT number over the gap and the calibrated aperture can be obtained using line fitting. With this preknowledge, we can characterization the natural fractures once we calculate the integrated "missing" CT in the fracture area. In order to obtain the integration, we need to identify the starting and ending points. Edge detection will allow us to mark these boundaries of the integration. We used the Roberts and Sobel gradient operators (Pratt, 1991) to detect edges. Figure 2.1 shows the impulse response arrays for the 3x3 orthogonal differential Roberts operator. We convolve these impulse response arrays with a two-dimensional image. Each pixel in the resulting image is set to 1 if the value is larger than the given threshold, or set to 0 otherwise. Therefore we acquire a binary image called an edge map. The advantage of the Roberts operator is its simplicity, however this method is sensitive to noise. In our experiment we used the Sobel operator (shown in Figure 2.2) to obtain the edge map.

2.3 Denoising

The edge detection methods are all sensitive to noise, although to different degrees, because they have similar processes such as taking derivatives of the signal. Thus for most natural CT images, we usually perform denoising on the data before it is passed to edge detection module. We applied soft-thresholding using the wavelet transform for denoising the core CT images. This was proposed by Donoho (1993). The beauty of this method is in its efficiency in preserving the true underlying data while suppressing noise. We will first talk about the wavelet transform, then we will discuss the soft-thresholding algorithm.

2.3.1 Wavelet Transform

Similar to the Fourier transform, the wavelet transform decomposes a time function on an orthogonal basis represented by a and τ (variables characterize the dilation and translation of basic wavelet). The wavelet transform of a function $f(t) \in L^2(R)$ is defined by

$$Wf(a,\tau) = \int_{-\infty}^{+\infty} f(t)\sqrt{a}\psi(a(t-\tau))dt \tag{2.14}$$

where $\psi(t)$ is called a basic wavelet (mother wavelet). The translations and dilation of the basic wavelet form a family given by $\sqrt{a}\psi(a(t-\tau))$, which is an orthonormal basis. Normally we use a discretized form:

$$Wf(m,n) = 2^{\frac{m}{2}} \sum_k f(k)\,\psi(2^m k - n) \tag{2.15}$$

In order to use this formula the signal has to be in dyadic length, i.e., signal length is equal to 2^J, where J is an integer.

In a wavelet multiresolution representation frame (Mallat, 1989), the signal is decomposed into approximations and details. The approximation at resolution $0 < j < J$ is in space V_{2^j}, which is spanned by an orthogonal basis formed by the dilation and translations of $\psi(t)$. The difference between the approximations at resolution 2^j and 2^{j+1} is the detail signal at resolution 2^j, which belongs to the space O_{2^j}, the

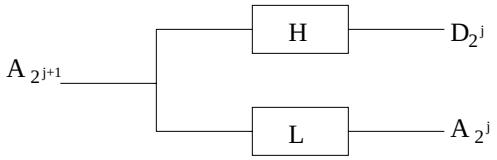


Figure 2.3: Scheme of wavelet decomposition

orthogonal complement of V_{2^j} in $V_{2^{j+1}}$. Space O_{2^j} is spanned by an orthogonal basis formed by the dilation and translation of a scaling function $\varphi(t)$. Between $\varphi(t)$ and $\psi(t)$, there is a pair of filters H and L, called the quadrature mirror filter (QMF), to correlate them. There are many papers (Cohen, 1992, Cohen, 1993 and Chui, 1992) talking about how to construct the QMF and consequently obtain $\varphi(t)$ and $\psi(t)$ with desired properties. Figure 2.3 shows the scheme of decomposition using the QMF pair H and L. A_{2^j} and D_{2^j} denote the approximation and detail at resolution 2^j. A_{2^j}, the approximation to $A_{2^{j+1}}$ at next coarser level 2^j is obtained by convolving $A_{2^{j+1}}$ with low-pass filter L; the corresponding complement detail signal D_{2^j} is obtained by convolving $A_{2^{j+1}}$ with high-pass filter H. Therefore the approximations and details are obtained by cascade filtering using QMF. For two-dimensional signals, we assume the columns and rows are separable and the scaling function and wavelet function can also be constructed separately.

$$\psi^1(x,y) = \psi(x)\,\varphi(y) \tag{2.16}$$

$$\psi^2(x,y) = \varphi(x)\,\psi(y) \tag{2.17}$$

$$\psi^3(x,y) = \psi(x)\,\psi(y) \tag{2.18}$$

This construction corresponds to the filtering on column and row data sequentially.

The advantage of the wavelet transform over the conventional Fourier transform is that wavelets can form an unconditional base of many functional spaces. The wavelet transform gives us a look at the signal at multiple resolutions and screens out the useful signal more efficiently.

2.3.2 Soft-Thresholding

Conventionally we filter noise in the frequency domain by assuming that the signal is banded and noise is spread over the entire frequency domain. Usually we use a low-pass filter to cut out the high frequency noise. This procedure is based on the representation of the signal in a Fourier series. Theoretical proofs and experiments show that this representation is not very efficient. Often the Gibbs phenomenon (oscillation in a rapid changing area) is induced during such process. By applying the wavelet transform, a signal is decomposed into its building blocks. In those wavelet coefficients, signal coefficients rise above the noise level. Since the wavelet transform compresses signal with finite energy l^2 into a small number (limited for discretized wavelet transform) of coefficient groups, signal amplitude will consequently increase. Also the Gaussian white noise property is kept during orthogonal transformation, therefore the noise level remains the same. The amplitude of the noise level is proportional to $\sqrt{\log(n)}$, where n is the signal length (Donoho, 1993). We can take advantage of this property to threshold the noisy wavelet coefficient while preserving the signal.

Assume the measured noisy data is represented as

$$y_i = f(t) + \sigma z_i i = 0, ..., n-1 \tag{2.19}$$

The procedure of soft-thresholding denoising consists of three steps (Donoho, 1992):

(1) Decompose the measured y_i using the wavelet transform and obtain the empirical wavelet transform coefficients wc.

(2) Apply the thresholding nonlinearity $\bar{wc} = sgn(wc)\,(|wc| - thr)$ with the threshold chosen as $thr = \sqrt{2\log(n)\,\sigma}$.

(3) Invert the wavelet transform to yield the $\hat{f(t)}$, which is the restored version of the $f(t)$ from noisy sample y_i. Figure 2.4 depicts the Gaussian noise effects in the wavelet coefficients domain. Plot (a) shows the noisy signal. Plot (b) shows the true signal behind the noisy features in (a). Plots (c) and (d) display the wavelet coefficients obtained by transformation using a simple Haar wavelet. Comparing (c) and (d), we can tell that the signal coefficients do stand out from the noisy background. Plot (e) displays the thresholded wavelet coefficients, which look similar

to those of clean signal (b). We can apply the same procedure to CT image data to obtain an efficient recovery of a true image.

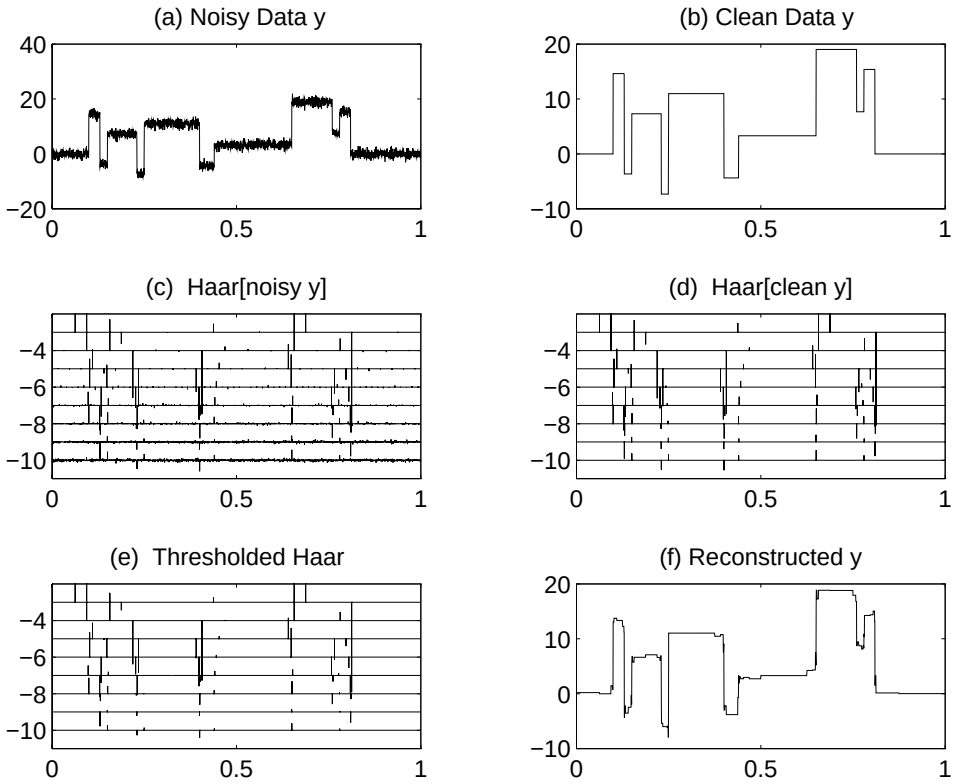
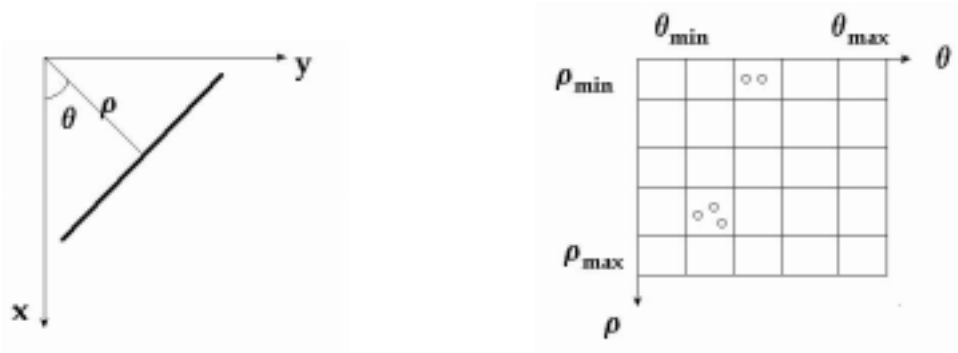


Figure 2.4: Comparison between clean block signal, noisy block signal and denoised block signal

2.4 Hough Transform

The Hough transform can detect specific structural relationships between pixels in an image. The transform was proposed by Hough in 1962. Commonly, we use this approach to find subsets of n points in an image that lie on straight lines (Pratt, 1991). A line can be represented as

$$x\cos\theta + y\sin\theta = \rho \tag{2.20}$$



(a) (b)

Figure 2.5: (a) Normal representation of a line. (b) Quantization of $\rho\theta$ plane into cells.

The meaning of the parameters used in Eq. 2.20 is illustrated in Figure 2.5(a). The Hough transform subdivides the parameter space into the accumulator cells shown in Figure 2.5(b).

The cell at coordinate (i, j) in the quantized $\rho\theta$ plane, with accumulator value $A(i, j)$, corresponds to a square associated with parameter space coordinates (ρ_i, θ_j). Initially, these cells are set to zero. Then, for every point (x_k, y_k) in the image plane, we let parameter θ equal each of the allowed subdivision values on the θ axis and solve for the corresponding ρ using Eq. 2.20. The resulting ρ values are then rounded off to the nearest allowed value in the ρ axis (Gonzalez, 1987). Therefore M collinear points lying on a line with parameter pair (ρ_i, θ_j) will yield M sinusoidal curves that intersect at (ρ_i, θ_j) in the parameter space, i.e. $A(i, j) = M$. In the process of line detection, we usually set a threshold. If the value of the accumulator cell $A(i, j)$ is larger than the threshold, we claim that there is a line described by (ρ_i, θ_j) (van der Heijden, 1994).

2.5 Active Contour Model

The Active Contour approach is an energy-minimizing model (Kass, Witkin and Terzopoulos, 1987). Because of the dynamic behavior of the approach, the curve is often referred to as a snake. A snake is an energy-minimizing spline regulated by internal forces, image forces and constraint forces. The energy of the snake can be represented as

$$E^*_{snake} = \int \left[E_{int}\left(v(s)\right) + E_{image}\left(v(s)\right) + E_{cont}\left(v(s)\right)\right] ds \tag{2.21}$$

where $v(s) = (x(s), y(s))$ represents the position of snake parametrically. E_{int} represents controlled continuity spline forces which impose a piecewise smoothness constraint. The constraint can be written

$$E_{int} = \frac{\left(\alpha(s)\left|v_s(s)\right|^2 + \beta(s)\left|v_{ss}(s)\right|^2\right)}{2} \tag{2.22}$$

The first-order term makes the snake act like a membrane and the second-order term makes it act like a thin plate. $\alpha(s)$ and $\beta(s)$ are the weights of the two terms. E_{image} represents the image forces. Here we set

$$E_{image} = -\left|\nabla I(x, y)\right|^2 \tag{2.23}$$

In this way a snake is attracted to a contour with large gradients. There are other applications in which the image force can include line force and termination force. According to our image content, we are focused on the edge information.

Minimizing the energy function given by Eq. 2.21 gives rise to the Euler equation in finite difference form, Eq. 2.26. Let

$$f_x(i) = \frac{\partial E_{ext}}{\partial x_i} \tag{2.24}$$

$$f_y(i) = \frac{\partial E_{ext}}{\partial y_i} \tag{2.25}$$

where $E_{ext} = E_{image} + E_{con}$.

$$\begin{aligned} &\alpha_i (v_i - v_{i-1}) - \alpha_{i+1} (v_{i+1} - v_i) \\ &+\beta_{i-1} (v_{i-2} - 2v_{i-1} + v_i) - 2\beta_i (v_{i-1} - 2v_i + v_{i+1}) \\ &+\beta_{i+1} (v_i - 2v_{i+1} + v_{i+2}) + (f_x(i), f_y(i)) = 0 \end{aligned} \tag{2.26}$$

Eq. 2.26 can be reorganized into matrix form.

$$Ax + f_x(x, y) = 0 \tag{2.27}$$

$$Ay + f_y(x, y) = 0 \tag{2.28}$$

where A is a pentadiagonal banded matrix. Eqs. 2.27 and 2.28 are solved using an explicit Euler method with respect to the external forces. The resulting equations are

$$Ax^t + f_x\left(x^{t-1}, y^{t-1}\right) = -\tau\left(x^t - x^{t-1}\right) \tag{2.29}$$

$$Ax^t + f_y\left(x^{t-1}, y^{t-1}\right) = -\tau\left(y^t - y^{t-1}\right) \tag{2.30}$$

where τ is the step size. At equilibrium, the time derivative vanishes and a stable solution is obtained. Eqs. 2.29 and 2.30 can be solved by matrix inversion resulting in

$$x^t = (A + \tau\, I)^{-1}\left(x^{t-1} - f_x\left(x^{t-1}, y^{t-1}\right)\right) \tag{2.31}$$

$$y^t = (A + \tau\, I)^{-1}\left(y^{t-1} - f_y\left(x^{t-1}, y^{t-1}\right)\right) \tag{2.32}$$

The common problem in solving the Eqs.2.29 and 2.30 is the instability due to the discretization of the evolution (Cohen, 1990). The choice of τ can not be too large even if the external forces in only a few points is large. Instead of adjusting the time step, we normalize the forces.

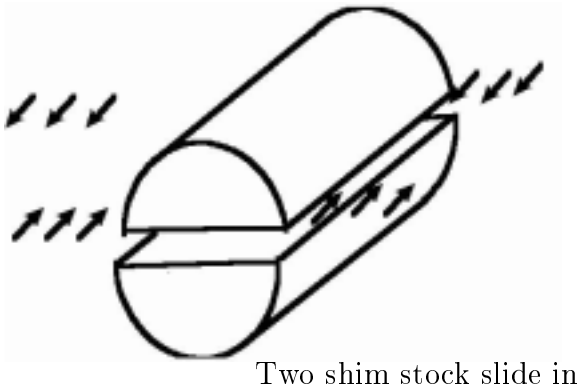
$$\left(\bar{f}_x(v), \bar{f}_y(v)\right) = \frac{(f_x(v), f_y(v))}{||\,(f_x(v), f_y(v))|\,|} \tag{2.33}$$

In this way the small and large image gradients have the same influence on the curve.

Section 3

Calibration Experiment

This experiment served the purpose of investigating the dispersion effect inherent in the CT scanner and calibrating the linear curve of the fracture aperture size vs. the integrated missing CT number over the gap (Johns, 1991). The calibration experiment should be done in a material with a distinctive density value within which the fracture will appear.



Two shim stock slide in along the arrows

Figure 3.1: Experimental setup

The experimental apparatus is illustrated in Figure 3.1. A basalt core was used because of its homogeneity. The original basalt core was 4.5 cm in diameter and 8 cm in length. The core was cut in half and both planar surfaces were optically polished.

The smoothness of the surfaces was 0.5 micron. Two shim stocks with calibrated thickness were inserted from both ends of the core along the direction of the arrows and an artificial fracture was created. The range of fracture size was 0.001∼0.04 inch. An aluminum ring was used to surround the basalt core in order to reduce the X-ray beam hardening effect.

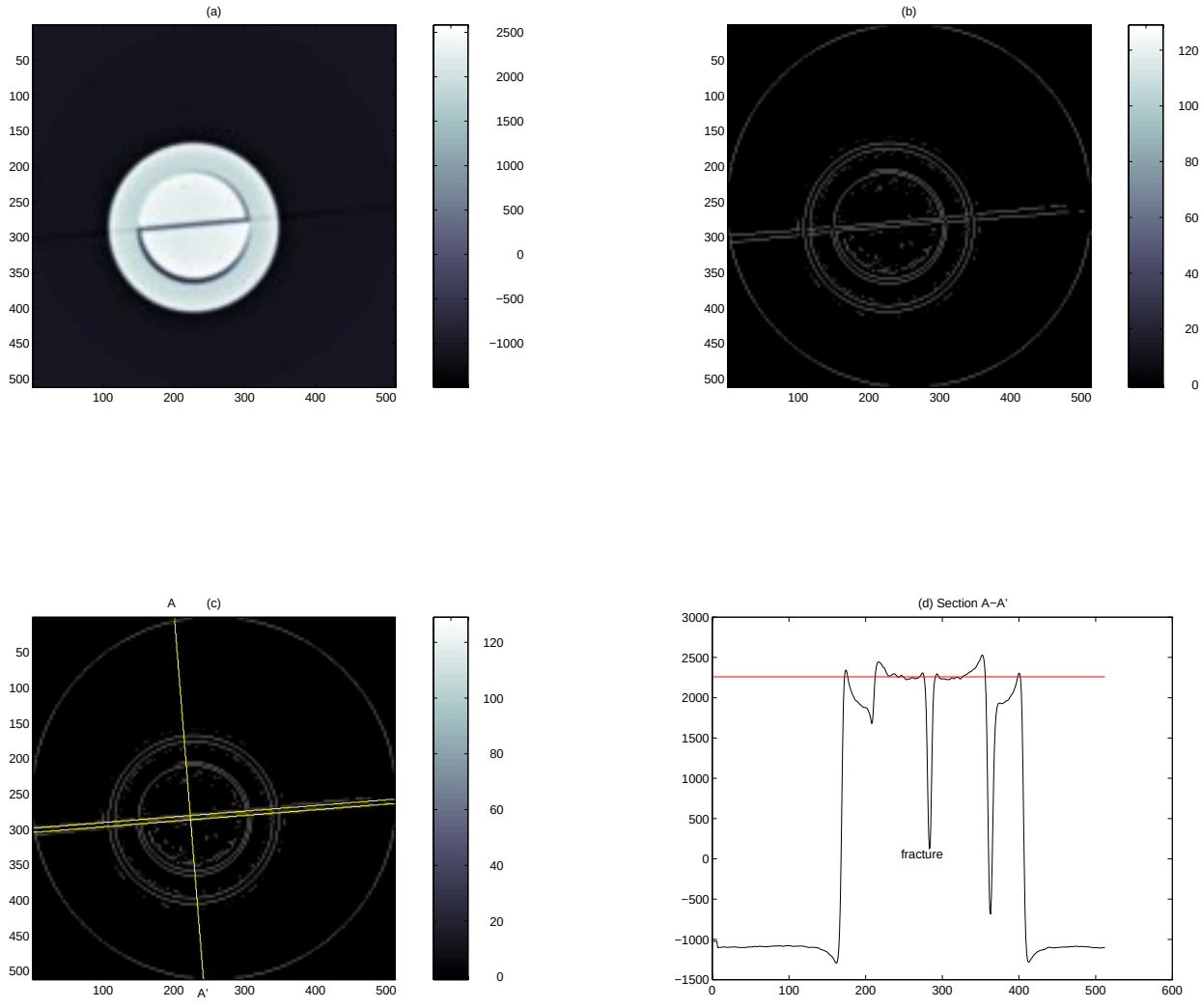


Figure 3.2: Fracture: aperture = 0.04 in

Figures 3.2 to 3.8 demonstrate the variation of fracture aperture. The (a) plots

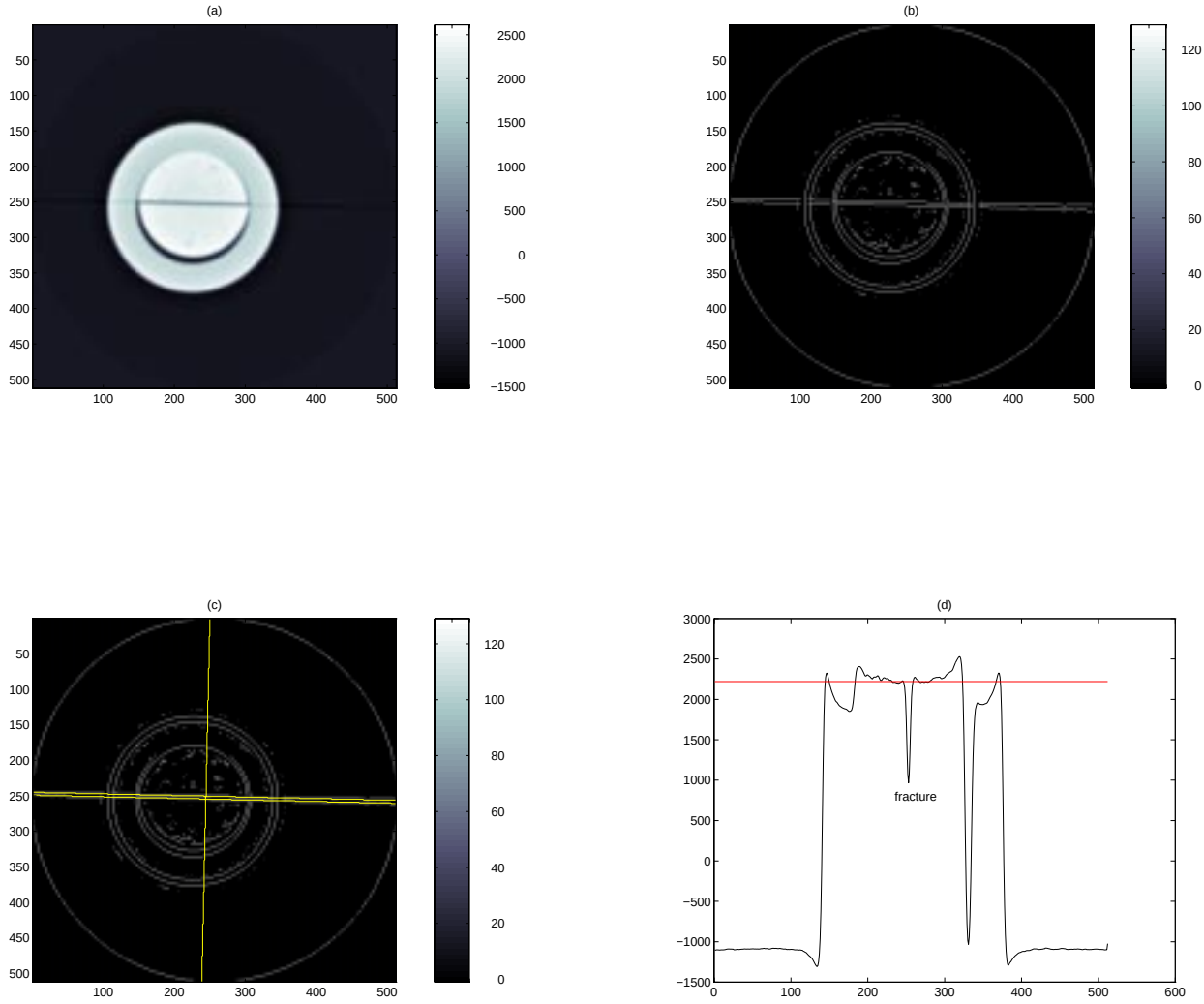


Figure 3.3: Fracture: aperture = 0.02 in

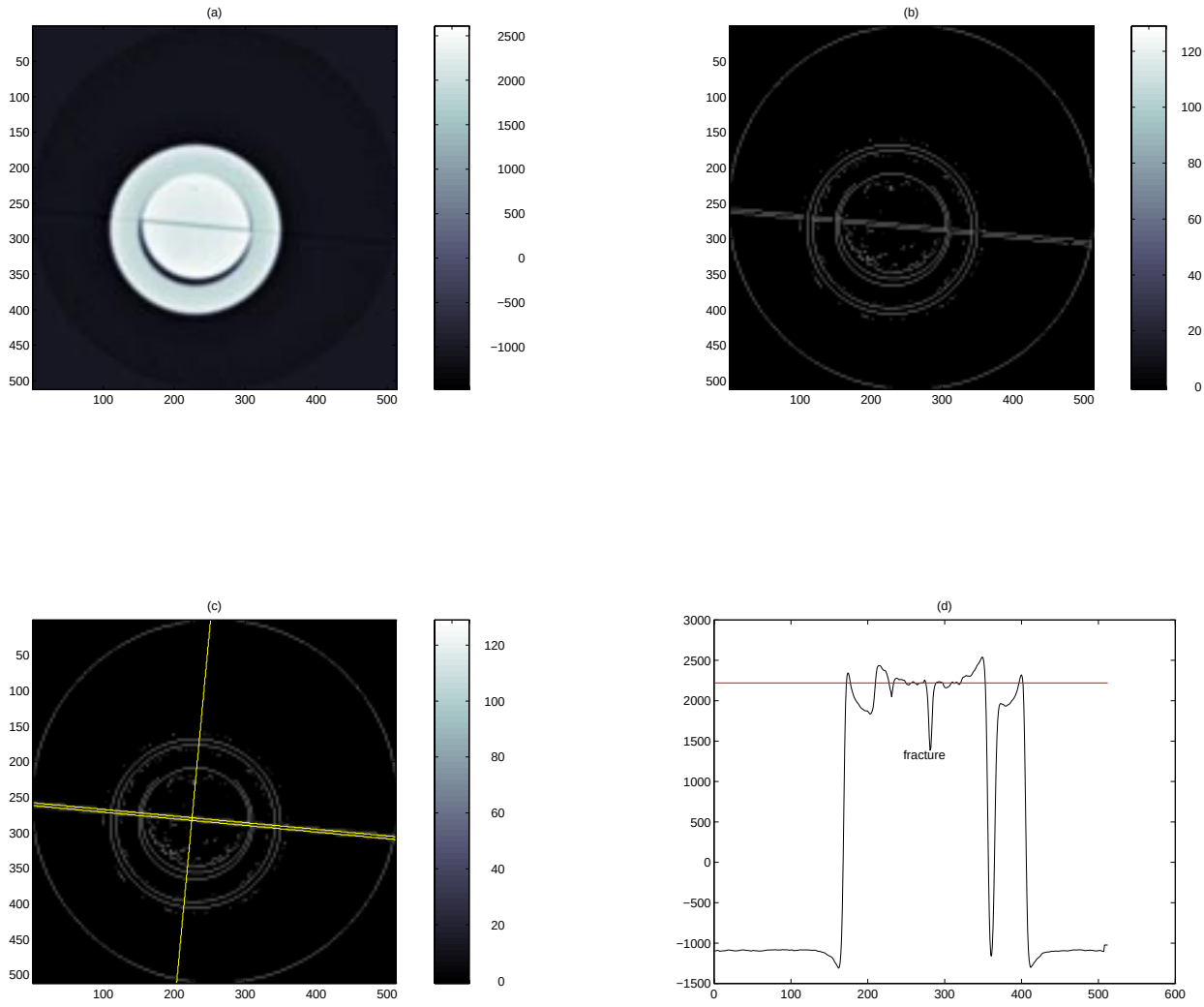


Figure 3.4: Fracture: aperture = 0.012 in

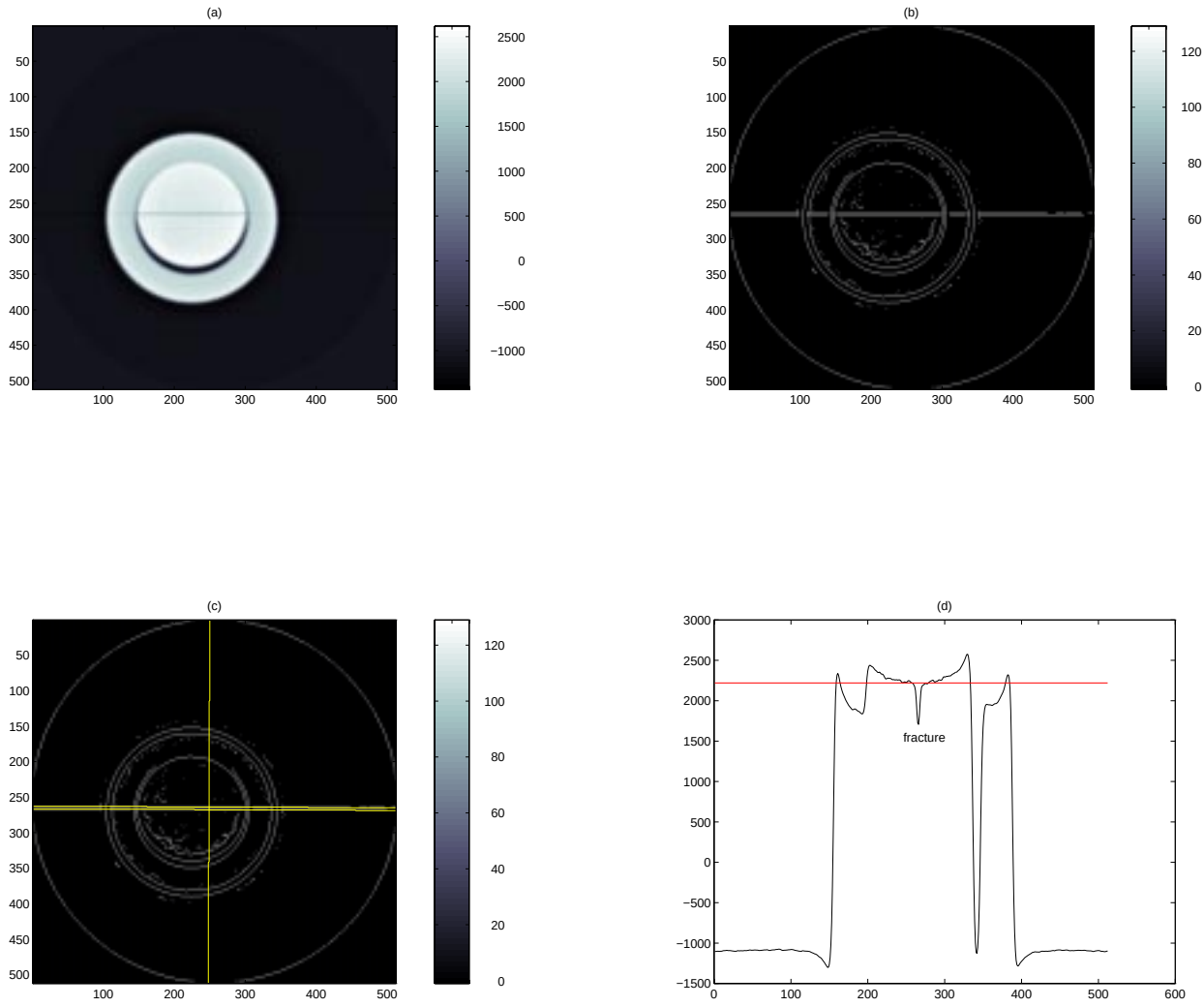


Figure 3.5: Fracture: aperture = 0.008 in

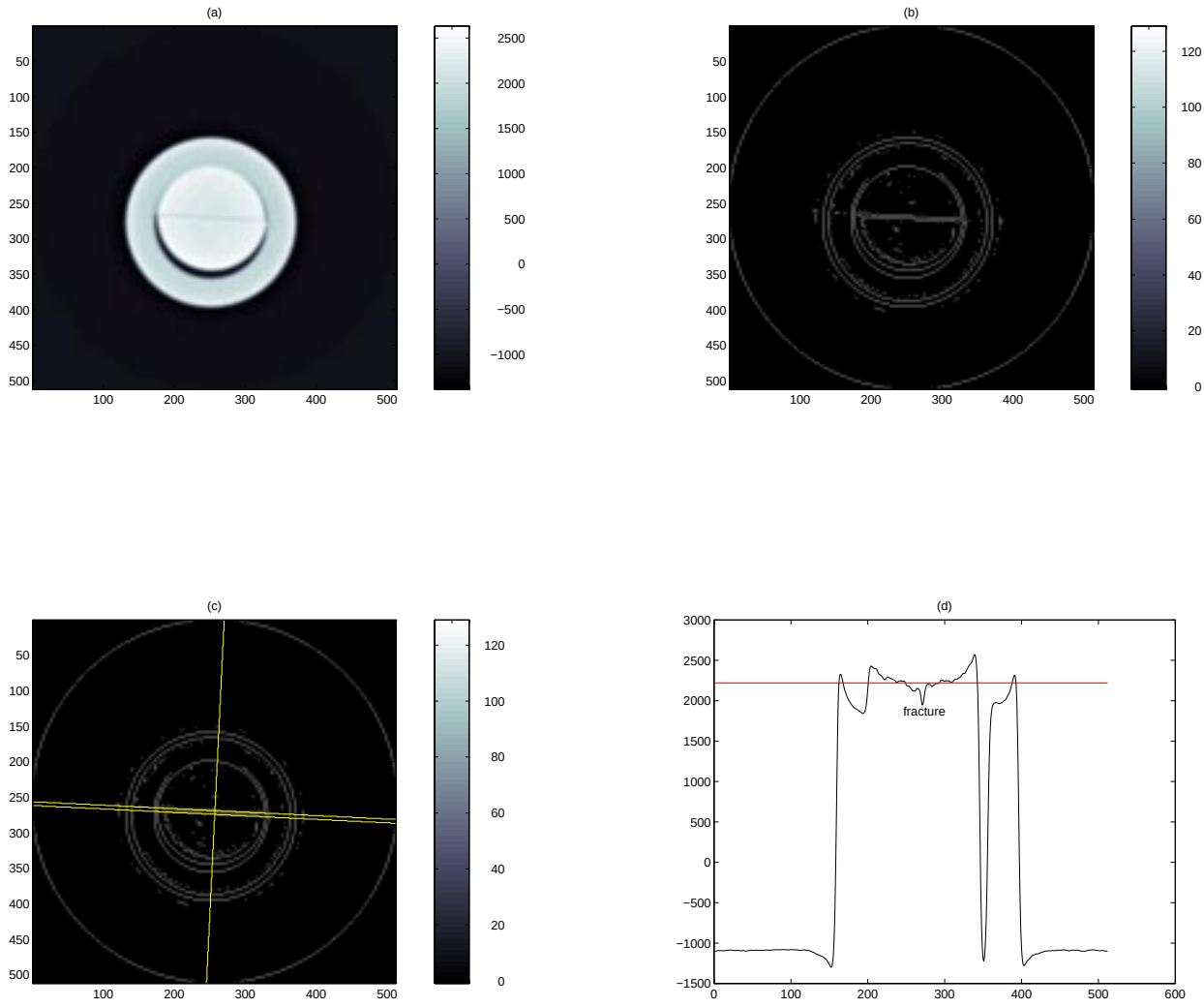


Figure 3.6: Fracture: aperture = 0.004 in

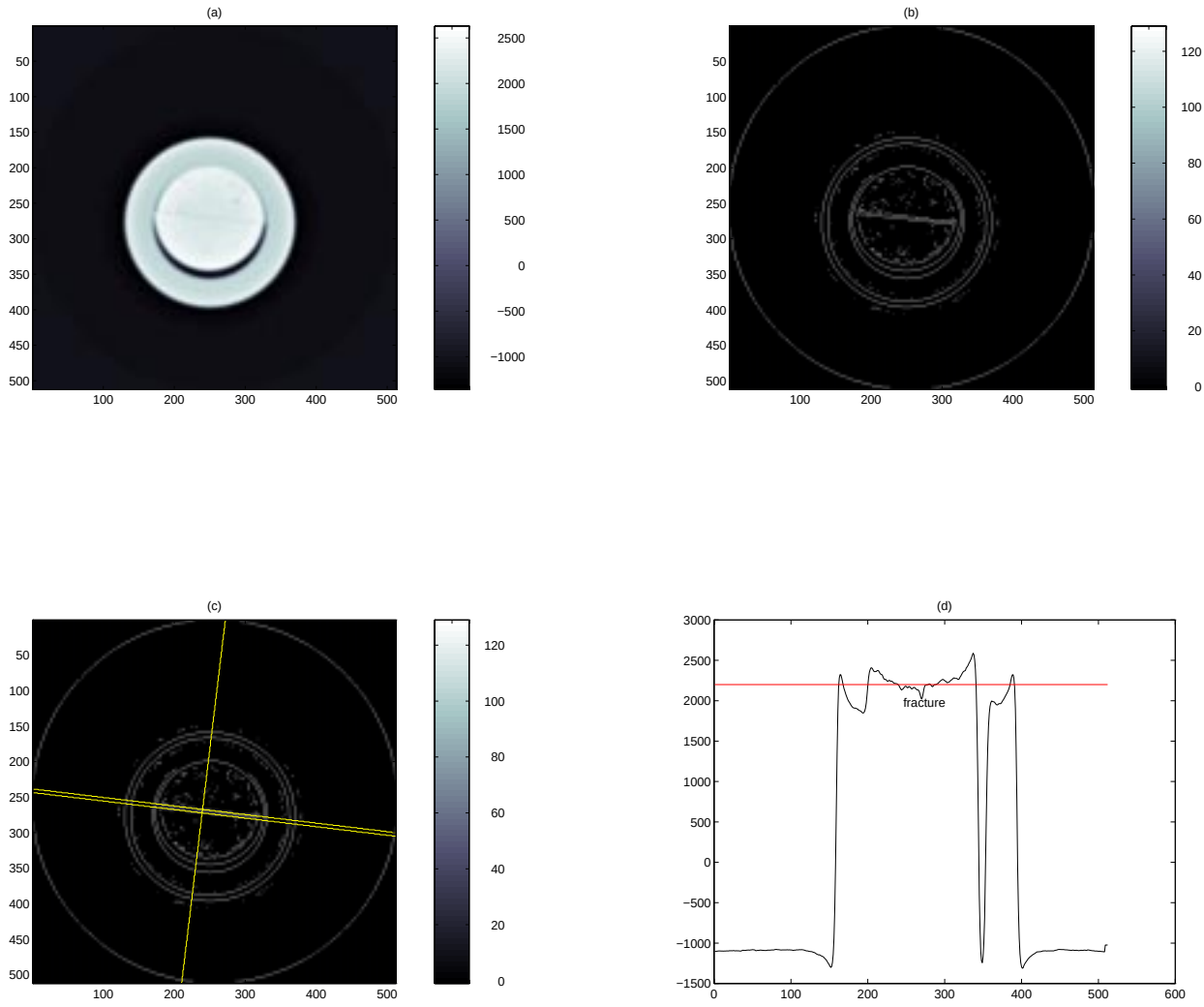


Figure 3.7: Fracture: aperture = 0.002 in

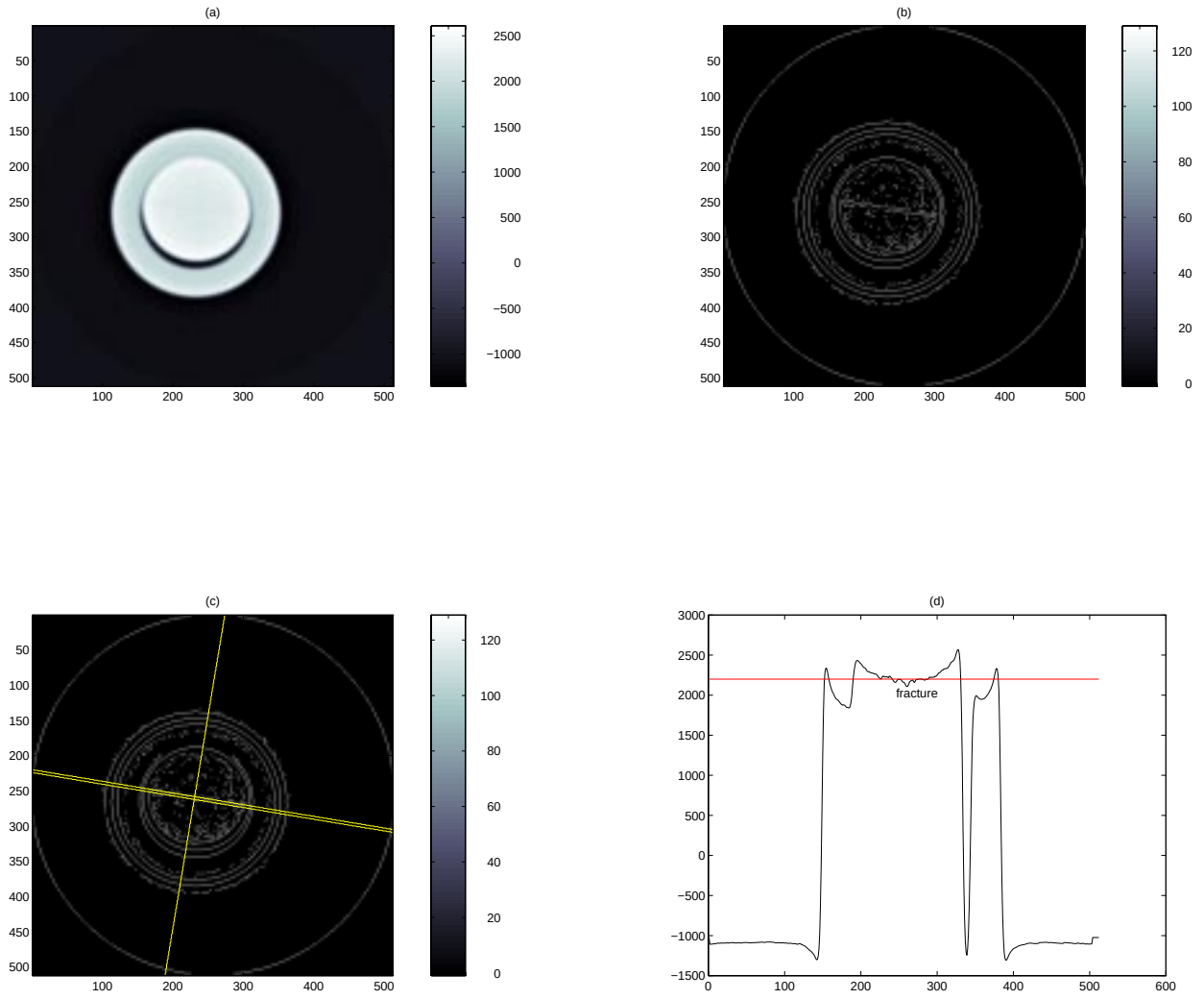


Figure 3.8: Fracture: aperture = 0.001 in

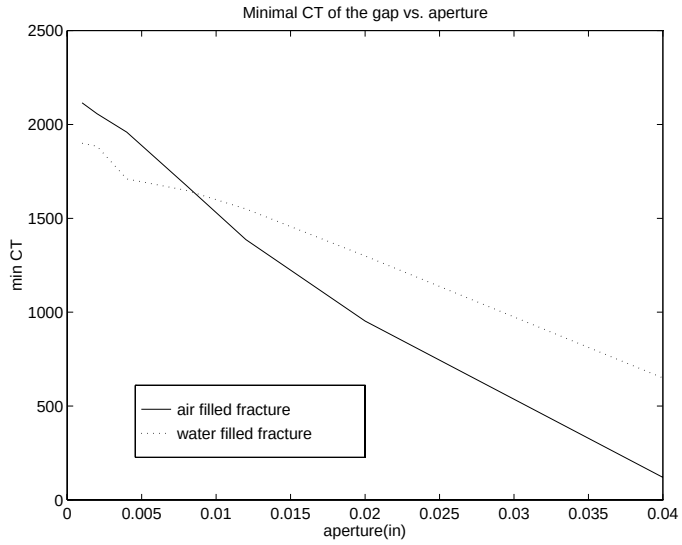


Figure 3.9: Minimal CT over the gap vs. aperture

in these figures show the intensity image of the basalt core with fracture. The (b) plots show the edge maps. The (c) plots show the location of profiles plotted against the edge maps with fracture boundaries emphasized by two parallel straight lines. The (d) plots show the profiles of the CT numbers along the line AA' shown in (c). The valley in the profile corresponding to the fracture becomes shallower when the fracture is narrower. Figure 3.9 shows that the minimum CT number in the valley of profile increases as the aperture decreases. However when the fracture was filled with water, the two smallest fractures became indistinguishable and the valley in the profile is buried. Figure 3.10 shows the linear relationship between the integrated CT and the fracture aperture. In our calculation of integrated missing CT, the CT number in the neighborhood of the fracture boundary is the density level with which each pixel in the gap is being compared.

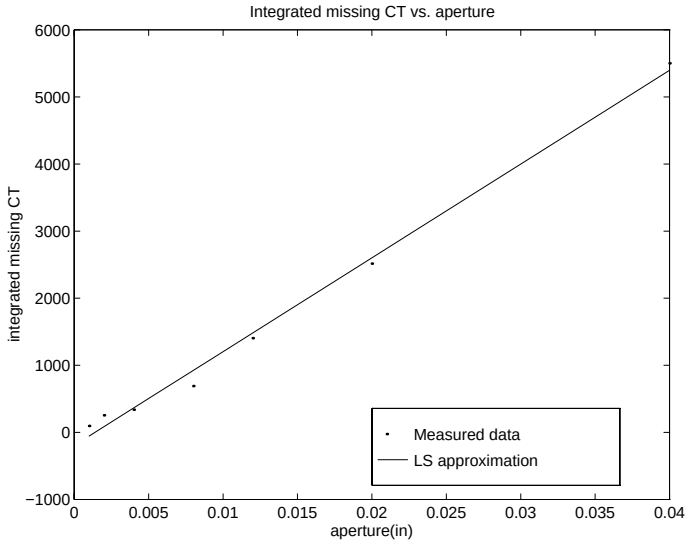


Figure 3.10: Relationship between integrated missing CT and aperture: y=1.398e5*x-194.5402

Section 4

Results

4.1 Denoising Core CT Image

The soft-thresholding algorithm was introduced in Section 2.5 along with an example of one-dimensional signal denoising. This denoising procedure consists of three steps, (1) forward wavelet transform, (2) wavelet coefficient thresholding and (3) inverse wavelet transform. In the two-dimensional case, we apply a high-pass filter and low-pass filter on the rows and columns of the image data respectively. After one level decomposition, we obtain the pattern shown in Figure 4.1.

The noise level σ is estimated from the median value of detail sub-image HH. We also know that white noise is proportional to $\log \sqrt{(N)}$, where N is the signal length. For the two-dimensional core image the threshold is chosen as $\sigma \log N$, where N is the dimension of image.

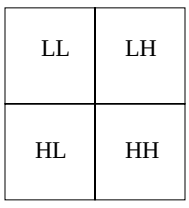


Figure 4.1: Coefficient pattern after one level of wavelet decomposition

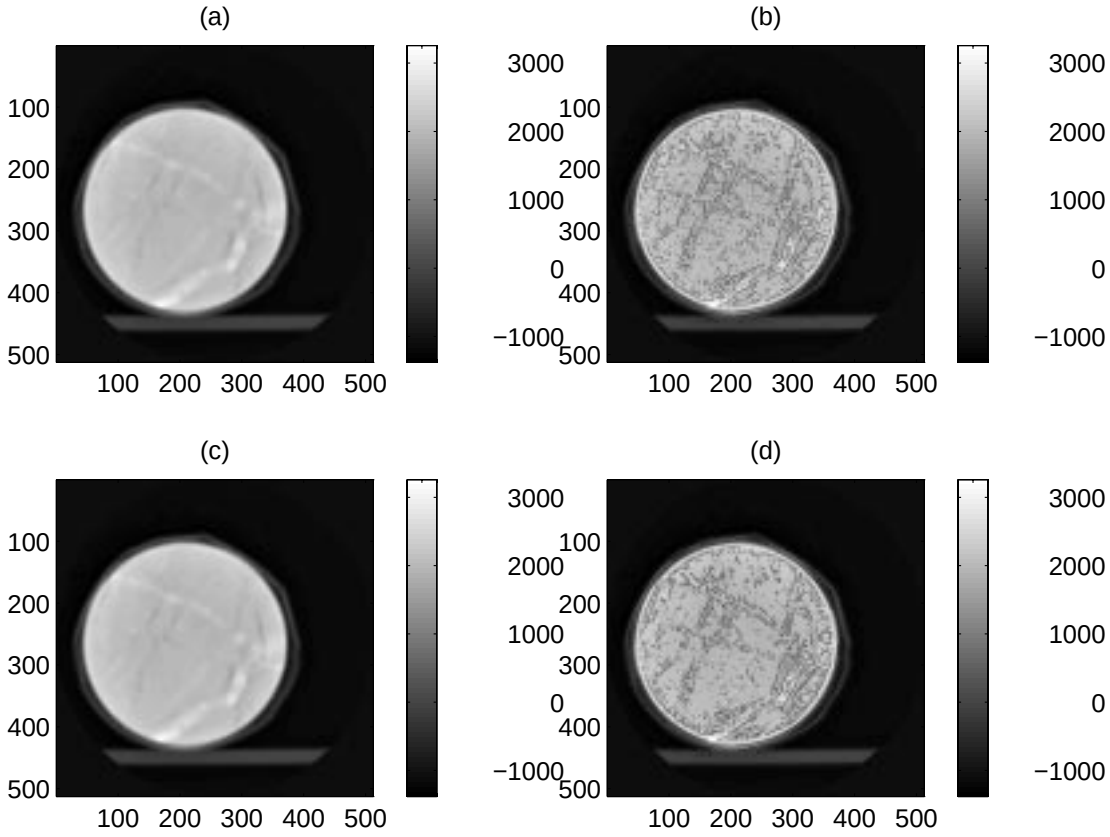
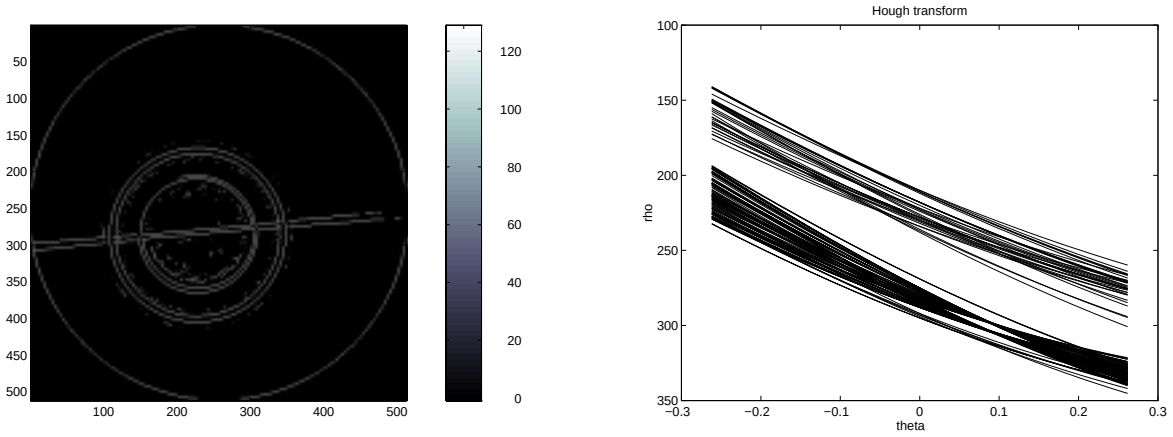


Figure 4.2: Denoising effect. (a) Original core image. (b) Edge map using Roberts gradient operator (thr=20) on the original image. (c) Denoised core image. (d) Edge map using Roberts gradient operator (thr=20) on the denoised image.



(a) (b)

Figure 4.3: (a) Edge map of sample of basalt core with a fracture of 0.04in. (b) Hough transform corresponding to (a).

4.2 Hough Transform

As described in Section 2.4, the edge map of the basalt core with artificial fracture was used as input data to calculate the Hough transform in order to obtain the slope of the two boundaries of the fracture. We are looking for the distinctive common intersects of sinusoidal curves in the parameter plane. Figure 4.3(a) depicts the input edge map of fractured basalt core. Figure 4.3(b) shows the Hough transform of edge points in (a). Two intersects are distinguished, $\rho_1 = 303.6699$, $\theta_1 = 0.0754$, and $\rho_2 = 297.6210$, $\theta_2 = 0.0843$. We drew lines according to these parameter pairs as is shown in Figure 4.3(b). These two lines correctly indicate the location of the boundary of the artificial fracture in basalt core (see Figure 3.2(c)). We also obtained the orientation of the profile subsequently. The line perpendicular to these two parallel lines shows the location of the CT profile in Figure 3.2(d).

4.3 Finding the Contour Using Active Contour Model

In our application of the Active Contour Model the external constraint forces are not included and the image forces only include edge forces. Eqs. 2.29 and 2.30 solve the snake model implicitly with respect to the internal forces and explicitly with respect to the external forces. The time step τ should be adjusted according to the external forces. Large external forces may cause the v^{t-1} move too far across the desired minimum position and lose the convergence. We normalized the external forces to make the convergence less sensitive to τ.

4.3.1 Finding the Artificial Fracture Contour

The active contour method described in Section 2.5 was applied to the artificial fracture in the basalt core. Figure 4.4 shows the moving path of the snake during evolution. The correct contour was obtained at equilibrium.

Figure 4.5 demonstrates the problem with two adjacent objects and starting the snake not close enough the object of interest. The snake was attracted by edges of objects other than the fracture under study. This problem is due to the theoretical basis of the snake model. The snake model attempts to find the equilibrium of the forces or the minimum of the energy. However this minimum is not unique. Since in most situations we are only interested in the local minimum, we can certainly confine our solution. Therefore we have to push the snake closer to the fracture of interest in order to let the snake find the right path.

4.3.2 Finding the Contour of Natural Fractures

We applied the Active Contour Model to a CT image of a Geysers rock sample. Figure 4.6 shows the contours of all the detected fractures in the Geysers sample.

After we have found the contours of the fracture, we can calculate the fracture aperture based on the calibration curve shown in Figure 3.10. Figure 4.7 shows an example of this approach, as applied to one of the identified fracture in Figure 4.6. Since the orientation of the contour shown in Figure 4.7 does not change much,

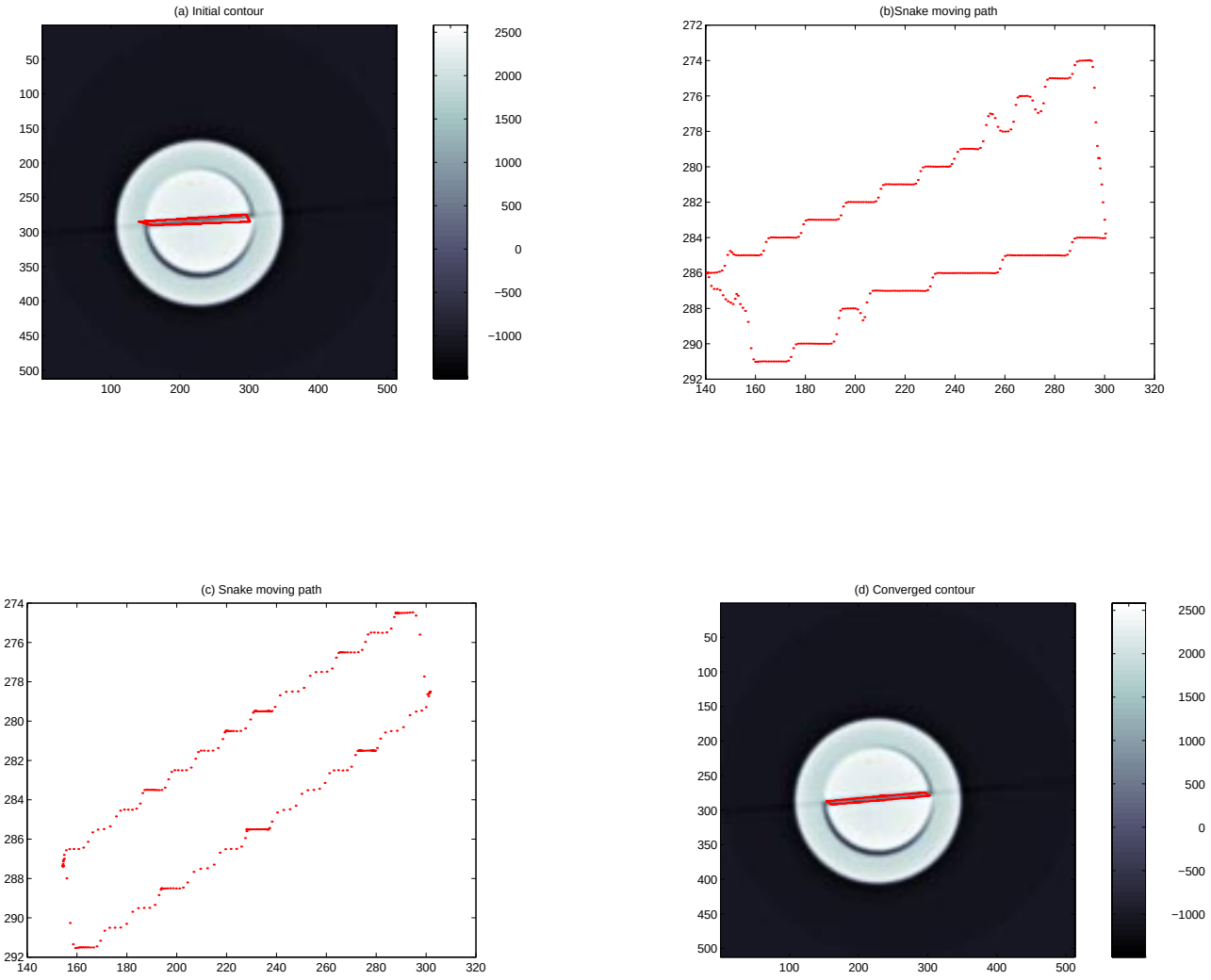


Figure 4.4: Finding a contour using Active Contour Model (a) Initial contour of fracture. (b) Locus of snake after one iteration. (c) Stable locus of snake after 300 iterations. (d) Final contour of fracture.

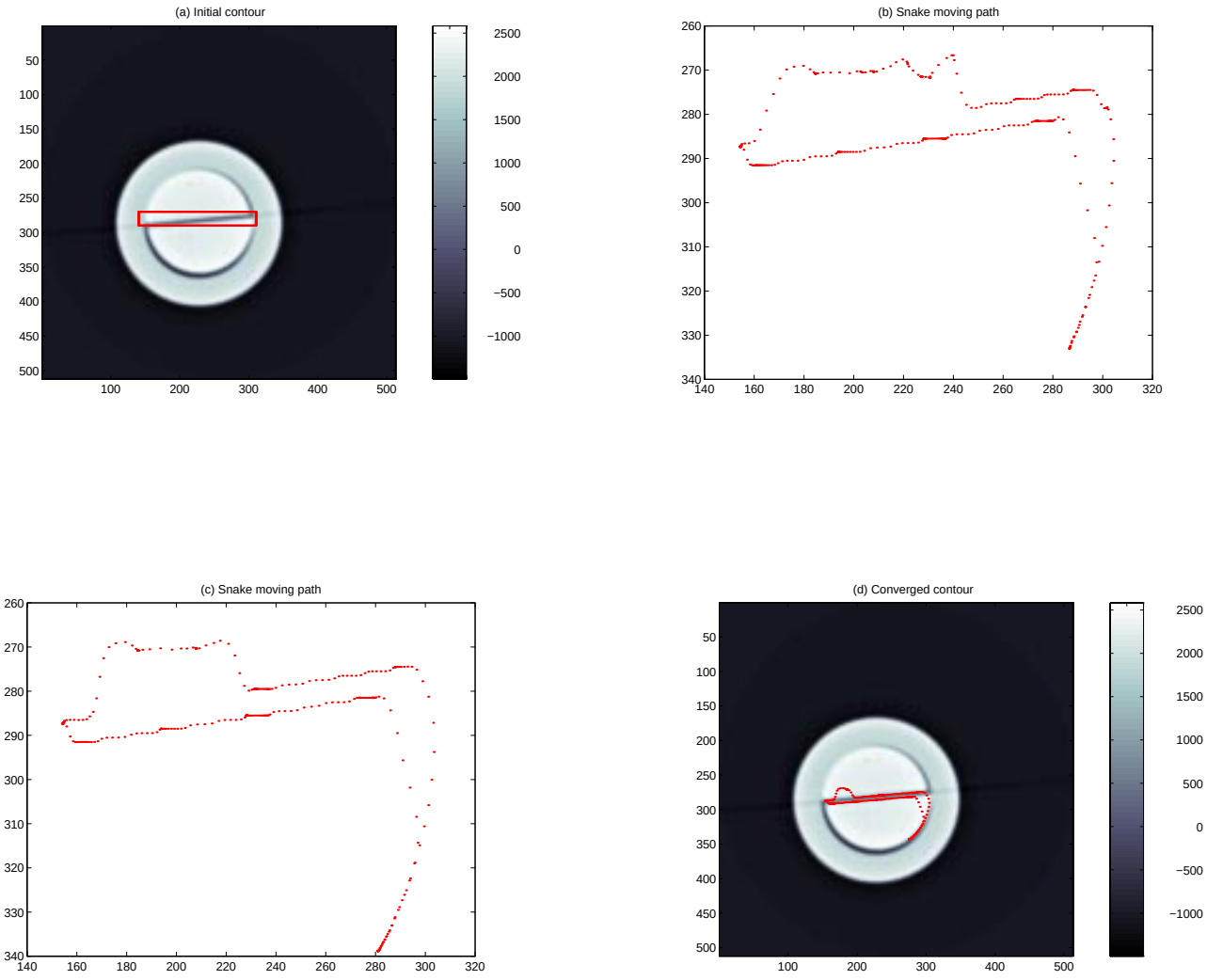


Figure 4.5: Case study of setting initial contour not close enough to object of interest. (a) Initial contour of fracture. (b) Locus of snake after one iteration. (c) Stable locus of snake after 300 iterations. (d) Final contour of fracture.

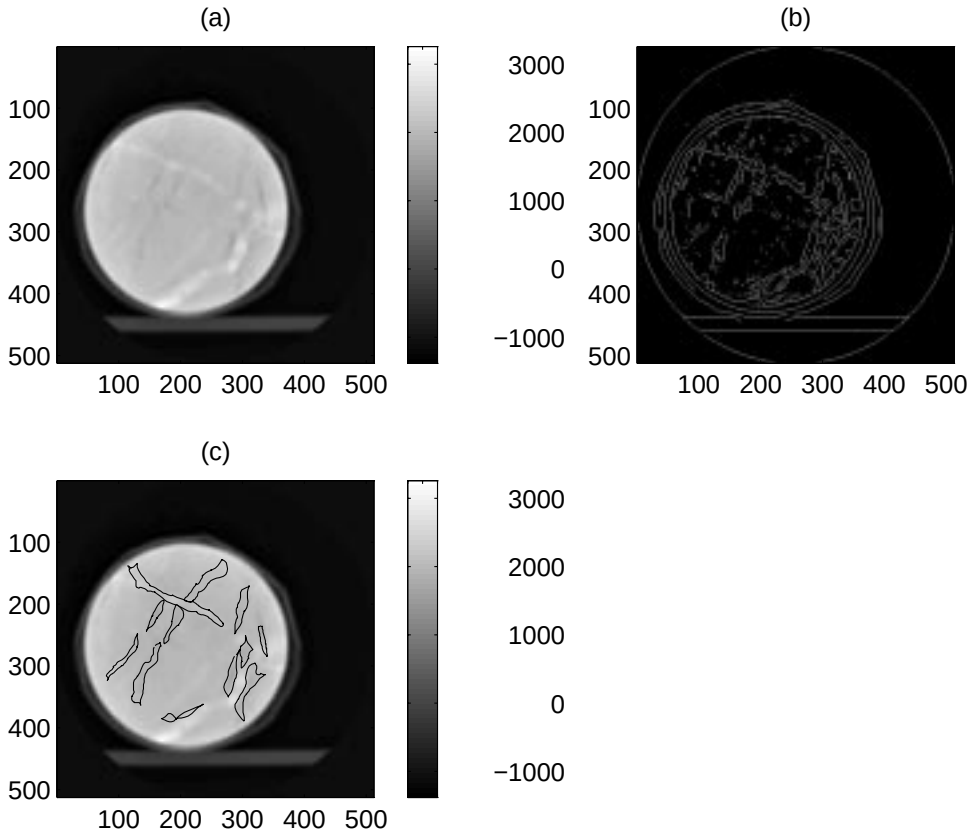
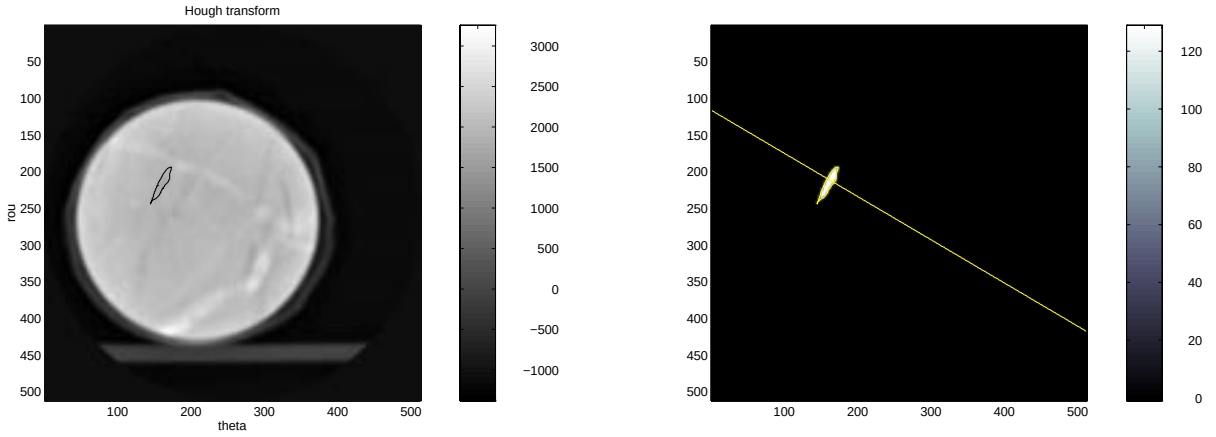


Figure 4.6: Finding fracture contours using snake (a) Original core image. (b) Edge map using Sobel gradient operator (thr=20) on the original image. (c) Connected contours found by snake.



(a) (b)

Figure 4.7: (a) Contour of a typical fracture in Geysers core. (b) Aperture calculation on the contour shown in (a).

we used the global Hough transform to estimate the orientation of the fracture and determine the appropriate location of the profile. The orientation of the contour can be described by θ of 1.0472. (See Figure 2.5(a) for the definition of the θ.) At the location shown in Figure 4.7(b) the aperture is 0.1226mm. Natural fractures usually have changing orientation and aperture (see Figure 4.6). The procedure used to calculate aperture shown in Figure 4.7 may not be applied in such cases. To account for the variation of aperture, we calculated the aperture at each pixel on each contour of the fractures in the entire slice of the sample of Geysers core. Figure 4.8 shows the distribution of fracture aperture in Geysers sample. Most apertures are below 250 micron.

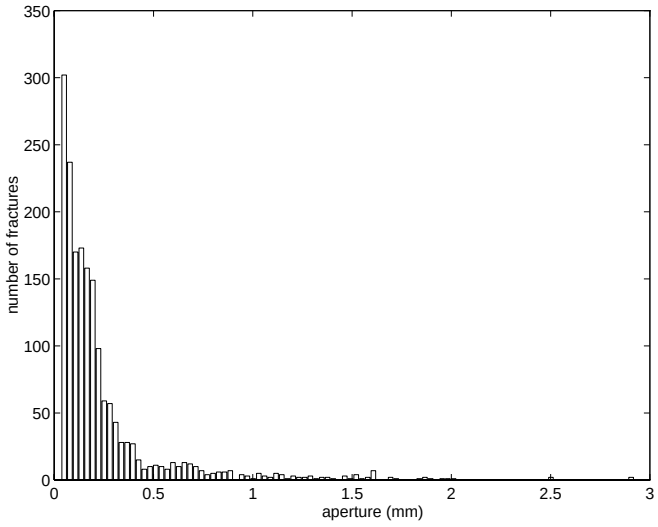


Figure 4.8: Aperture distribution in Geysers sample

Section 5

Conclusions and Discussions

A general procedure for determining fracture aperture in geothermal rocks has been developed. The procedure includes the following steps:

(1) The experimental calibration of a curve of integrated missing mass vs. aperture using homogeneous material having similar density to the natural fractured core.

(2) Edge detection on the natural fractured core. A denoising procedure may be applied prior to edge detection.

(3) The active contour algorithm is used to refine the edge map.

(4) Components are labeled if necessary. This procedure is useful when we need to address an individual contour in an image.

(5) The identified fracture orientation is estimated using the Hough transform. The fracture aperture size is inferred from the empirical curve of integrated missing CT number vs. fracture aperture obtained from the calibration experiment. The saturation of fluid in the fracture region can also be estimated.

For the detection of a regular fracture, we can use the global Hough transform to estimate the orientation of the fracture. For a natural fracture, we can use the local Hough transform. The window of the local transform is adjustable according to the variation of the fracture orientation.

In our application of the snake model, the initial curve is placed outside of the object of interest, and internal forces guide the snake to find its way. Spurious points inside the desired contour do not have any influence on the contour obtained. This

way of setting the initial contour is easier than the one presented by Cohen(1990). However, if there is a nearby object that is too close to the object of interest, the initial curve has to be placed closer to the one under study.

The denoising procedure is only necessary when the edge map is too noisy. The Active Contour method has resistance to noise while finding the contour.

One of the advantages of the snake model that can be explored in the future is the flexibility of including constraints, such as edge information. We can utilize the previous achievement in edge detection, for instance the Canny-Deriche edge detector (Canny, 1986) and edge detection using the wavelet transform. The attraction forces can be defined by simulating a potential obtained by convolving the binary edge image with a Gaussian impulse response. This can attract the snake to the small edge segments.

In general, the Active Contour model combined with the edge detection and Hough transform can significantly improve the detection quality of closed fractures, while enhancing the computational stability and reducing the complexity. Making use of edge detection, the Hough transform and the Active Contour model, we were able to characterize the fracture aperture distribution in a sample core from the Geysers geothermal field.

Bibliography

[1] Akin, S., Birol, D. M. R. and Okandan, E: "A Novel Method of Porosity Measurement Utilizing Computer Tomography," *In Situ* (1996) **20**, No. 3, 347–365.

[2] Aydin, T., Yemez, Y., Anarim, E. and Sankur, B.: "Multidirectional and Multiscale Edge Detection via M-Band Wavelet Transform," *IEEE Transanction on Image Processing* (September 1996) **5**, No. 9.

[3] Bonner, B. P., Roberts, J. J. and Schnelbert, D. J.: "X-ray Evidence for Capillary Pressure Driven Flow in Preserved Core from Geysers," Proceedings of the 23rd Stanford Workshop on Geothermal Reservoir Engineering (January 27–29 1997).

[4] Canny, J.: "A Computational Approach to Edge Detection," *IEEE Transanctions on Pattern Analysis and Machine Intelligence* (November 1986).

[5] Chan, Y. T.: *Wavelet Basics*, Kluwer Academic Press (1995).

[6] Chui, C. K.: *An Introduction to Wavelets*, Academic Press (1992).

[7] Chui, C. K.: *Wavelets: A Tutorial in Theory and Application*, Academic Press (1992).

[8] Cohen, A., Daubechies, I. and Feauveau, J. C.: "Biorthogonal Bases of Compactly Supported Wavelets," Communications on Pure and Applied Mathematics (1992).

[9] Cohen, A., Daubechies, I. and Vial, P: "Wavelets on the Interval and Fast Wavelet Transforms," *Applied and Computational Analysis* (1993) **1**.

[10] Cohen, L. D. and Cohen, I.: "A Finite Element Method Applied to New Active Contour Models and 3D Reconstruction from Cross Sections," Proceedings of the 3rd International Conference on Computer Vision (1990).

[11] Donoho, D. L.: "Denoising by Soft-Thresholding," Dept. of Statistics, Stanford University (1992).

[12] Donoho, D. L.: "Nonlinear Wavelet Methods for Recovery of Signals, Densities, and Spectra from Indirect and Noisy Data," Proceedings of Symposia in Applied Mathematics (1993).

[13] Gonzalez, R. C. and Wintz, P.: *Digital Image Processing*, second edition, Addison Wesley Publishing Company Ltd. (1987).

[14] Graps, A.: "A Introduction to Wavelets," *IEEE Computational Science and Engineering: Signal and Image Processing* (1995).

[15] Huang, Y., Ringrose, P. S. and Sorbie, K. S: "X-ray Imaging of Waterflood Fluid Saturation in Heterogeneous Slabs," *SPE30000* (1995) 483–495.

[16] Johns, R. A.: *Diffusion and Dispersion of Solute in a Variable Aperture Fracture*, PhD dissertation, Stanford University, Stanford, CA (November 1991).

[17] Kass, M., Witkin, A. and Terzopoulos, D.: "Snake: Active Contour Models," *International Journal of Computer Vision* (1987).

[18] Macovski, A.: *Medical Imaging Systems*, Prentice Hall (1983).

[19] Mallat, S. G.: "A Theory for Multi-resolution Signal Decomposition: The Wavelet Representation," *IEEE Transanction on Pattern and Machine Intelligence* (July 1989) **11**, No. 7.

[20] Marr, D. and Hildreth, E.: "A Theory of Edge Detection," Proceedings of Royal Society (1979).

[21] Pratt, W. K: *Digital Image Processing*, third edition, John Wiley & Sons, Inc, London, England (1991).

[22] Segal, E., Notea, A. and Segal, Y.: "Dimensional Information Through Industrial Computerized Tomography," *Material Evaluation* (November 1982).

[23] Strang, G. and Nguyen, T.: *Wavelets and Filter Banks*, Wellesley-Cambridge Press (1996).

[24] van der Heijden, F.: *Image Based Measurement Systems - Object Recognition and Parameter Estimation*, John Wiley & Sons (1994).

Appendix A

Core Functions

The following MATLAB programs were used to process core functions of the fracture detection procedure.

- hough.m — calculate the Hough transform. Accumulator array of the discretized parameter plane is returned. Be sure to supply the rough estimation of the range of the θ in which the orientation of the edge is expected to appear.
- out_in.m — solve the Active Contour Model with an initial contour specified around the object of interest.
- denoise_core.m — Soft-thresholding based on the wavelet transform.

A.1 Hough Transform

```
%=====================
%  File Name : hough.m
%  by Meiqing He
%  Last Modified  6/5/1998
%-------------------
%hough transform subroutine
%Functionality:
% Calculate the Hough transform. Accumulator array of the discretized
% parameter plane is returned. Be sure to supply the rough estimation
```

```
% of the range of the theta in which the orientation of the edge is
% expected to appear.
%=====================

function [Acc, Rou, theta]=hough(thetaMin, thetaMax, numStep, numAccCell, x , y)
% thetaMin    in degree
% thetaMax    in degree
% numStep     integer, discretization of the theta,
% limis the resolution
% numAccCell      integer, discretization of the Rou,
%         limits the resolution

N= length(x);

theta = linspace(thetaMin,thetaMax, numStep);
theta = theta*pi/180;

Rou = zeros(N,numStep);
for loop = 1:N,
Rou(loop,:) = x(loop)* cos(theta)+y(loop)*sin(theta);
end

RouMin = min(min(Rou));
RouMax = max(max(Rou));

%numAccCell also limits the resolution

Acc = zeros(numAccCell,numStep);
resolution = (RouMax-RouMin)/(numAccCell-1);
for loop1 = 1:numStep,
  for loop2 = 1: N,
    index = floor((Rou(loop2,loop1)-RouMin)/resolution)+1;
    Acc(index, loop1) = Acc(index, loop1)+1;
  end
end
```

A.2 Solving the Snake Model

```
%=============
% File Name: out_in.m
% by Meiqing He
% Last Modified  6/5/1998
%--------------
%subroutine solving the avtive contour (discrete Euler equations)
%Functionality:
%  For a given image "img" and an initial curve (x0,y0) around the object of
%  interest, function out_in(x0,y0,img)  finds the contour.
% =============

function [ctrI, ctrJ]=out_in(x0,y0,img)

n=length(x0);

% generate  n*n unit matrix I:
 I=eye(n);

% generate n*n matrix A:
 a=mtrxa(n)/4;

x1=zeros(n,1);
y1=zeros(n,1);
%tau is the discretization parameter of the time, has effect on the
%convergence, can be modified.
tao=1;

[m,m]=size(img);
px=zeros(m);
py=zeros(m);
fx=zeros(m);
fy=zeros(m);
potntl=zeros(m);
p=zeros(m);
% calculate the gradient
[px,py]=gradient(img);
```

```
%calculate the potential
potntl=absgrad(px,py);
%calculate the derivative of the potential
[fx,fy]=gradient(potntl);

% normalize external forces: fx,fy
p=absgrad(fx,fy);

for i=1:m
  for j=1:m
    if p(i,j)>.0001
     p(i,j)=sqrt(p(i,j));
     fx(i,j)=fx(i,j)/p(i,j);
     fy(i,j)=fy(i,j)/p(i,j);
     end
   end
 end

b=I+tao*a;

b_inv=inv(b);

   for ii=1:1000
    ii
%interpolate the external forces at the decimal location of the snake
   ffx=interp4(fx,x0,y0);
   ffy=interp4(fy,x0,y0);
%solve the snake position
   x1 = b_inv*(x0+tao*ffx);
   y1=b_inv*(y0+tao*ffy);

%  track the snake and the error step by step
% the if condition can be eliminated if you do not want to monitor the snake
% and want to save some time. Plotting can take a while to accomplish.

   if ii ==round(ii/300)*300 | ii ==1
```

```
      figure;
      plot([x1;x1(1)],[y1;y1(1)],'r.'), title('snake moving path');
      axis('ij')
      pause;
     end

%evaluate the movement of the snake. Compare the position difference between
%the new position acquired after the current iteration and the old one
   err=err1(x0,y0,x1,y1)

%update the position of the snake
   x0=x1;y0=y1;
   end

   ctrI = y0;
   ctrJ = x0;
```

A.3 Soft-Thresholding Denoising

```
%==============
% File Name: denoise_core.m
% by Meiqing He
% Last modified  6/6/98
%-------------
%This code implements two approaches.
%The first one approximates the noise process as Gaussian distribution
%the threshold for the Soft-Thresholding method is derived based on this
% See Donoho(1992) and Masters' report by Meiqing He(1998).
%The second one takes the noise process as it is(Poisson distribution) and
%transforms to Gaussian distribution in order to apply the Soft-Threshoding
%mehtod
%==============

%----------------
%read and display 2D core CT image
%512*512 grids
%---------------
DATAPATH = '/pangea/play/pete/meiqing/';
        filepath = [ DATAPATH '4635-01.txt'];
        fid = fopen(filepath,'r');

        data= fscanf(fid,'%g',[512 512]);
        fclose(fid);

img = data';

%-------------------------
%display the sub-images after one level decomposition
%-------------------------
D4QMF = MakeONFilter('Daubechies',4);
wcNoise = FWT2_PO(img,8,D4QMF);
wc1 = wcNoise(1:256, 1:256);
wc2 = wcNoise(1:256,257:512);
wc3 = wcNoise(257:512,1:256);
wc4 = wcNoise(257:512,257:512);
```

```
subplot(221);
imagesc(wc1);
colormap(bone);
axis('image');
title('LL');

subplot(222);
imagesc(wc2);
colormap(bone);
axis('image');
title('LH');

subplot(223);
imagesc(wc3);
colormap(bone);
title('HL');

subplot(224);
imagesc(wc4);
colormap(bone);
title('HH');
%----------------------
%calculate the threshold
%----------------------
nlevel = median(median(abs(wcNoise(256:512,256:512)))); %estimate the
%standard deviation of the noise from the HH

% thr = 2*nlevel;
% thr = 2.*(thr+3/8).^0.5;  % thr is transformed from Poisson
  %distribution to Gaussian distribution
thr = nlevel*sqrt(2*log(512*512-64))/0.6745; %thr of the Gaussian
                                                    %distribution

wcNoise = FWT2_PO(img,3,D4QMF); %forward transform
coarse = wcNoise(1:8,1:8);
Thr_wcNoise = SoftThresh(wcNoise, thr);
```

```
Thr_wcNoise(1:8,1:8) = coarse;
cleanCore = IWT2_PO(Thr_wcNoise,3, D4QMF);  %inverse transform
save cleanCore1 cleanCore;

%--------------------------
%transform the CT noisy data from Poisson distribution to Gausian
%distribution
%-------------------------

%%%%%%%%%%%%%%%%%%
%The second approach
%------------------------
img_t = 2.*(img+1024+3/8).^0.5; %Anscombe variance stabilizing
%transformation

D4QMF = MakeONFilter('Daubechies',4);
wcNoise = FWT2_PO(img_t,8,D4QMF);
nlevel = median(median(wcNoise(256:512,256:512)));
thr = 2*nlevel; &threshold level for Poisson process

wcNoise = FWT2_PO(img_t,3,D4QMF);
coarse = wcNoise(1:8,1:8);
Thr_wcNoise = SoftThresh(wcNoise, thr);
Thr_wcNoise(1:8,1:8) = coarse;
cleanCore = IWT2_PO(Thr_wcNoise,3, D4QMF);
cleanCore = (cleanCore./2.0).^2 -1024-3/8; % transform back to
   %Gaussian distribution
```

Appendix B

Utility Functions

The following MATLAB utility function were used.

- CalcLineXY.m — calculate the line vector to connect the two given end points specified by (x_1, y_1) and (x_2, y_2).
- CalcLine.m — calculate the line vector confined in 512x512 Cartesian coordinates according to the parameter pair (ρ, θ).
- CircMask.m — set the pixels outside the specified circle while maintaining the value of pixel inside the region.
- CircWindow.m — select a sub-vector from the original vector. The original vector acts like a cicular buffer. The center and length of the selection window has to be supplied.
- adjacent.m — judge if two points are adjacent.
- ctr2region.m — mark the region surrounded by the given contour with 1.
- err1.m — calculate the differences between the positions of the two contours. Mean square error standard is applied.
- manhanttanDist.m — calculate the Manhanttan distance of two points.
- mtrxa.m — generate the matrix A in Eq. 2.27 and 2.28.

- labeling.m — label all the features in an image.
- lookup.m — interpolate the fracture aperture according to the empirical linear curve of integrated missing CT vs. aperture.
- roi_try.imp — script file run under FPImage to convert an image data in specified region from IMG format to an ASCII data file.

B.1 Connect Two Points with a Straight Line

```
%==============
% File Name: CalcLine.m
% by Meiqing He
% Last modified  6/7/98
%--------------

function [I,J] = CalcLine(Rou, theta)

%according to the supplied parameter of a line
%calculate the coordinates of the line vector
%see Masters' report by Meiqing He(1998) for the definition of Rou and theta
%  0---------->J(y)
%  |
%  |
%  |
%  |
%  I(x)
%==============

if abs(theta)==pi/2
    I = [1:512];
    J = round(Rou)*ones(1,512);

elseif abs(theta)==0
    I = round(Rou)*ones(1,512);
    J = [1:512];
```

```
elseif abs(theta) > pi/4
    I = [1:512];
    J = round((Rou-I*cos(theta))/sin(theta));

else
%if abs(theta) < pi/4
    J = [1:512];
    I = round((Rou-J*sin(theta))/cos(theta));
end
```

```
%===============
% File Name: CalcLineXY.m
% by Meiqing He
% Last modified  6/7/1998
%---------------

function [I,J] = CalcLineXY(xStart, yStart, xEnd, yEnd)

%according to the supplied coordinates of starting point and ending point
%of a line, calculate the coordinates of the points of the line connecting two
%points
% 0------->J(y)
% |
% |
% |I(x)
%===============

if xStart == xEnd, %line parallel to Y axis
   if yStart < yEnd,
     J = [yStart:yEnd];
   else
     J = [yStart:-1:yEnd];
   end
   I = xStart*ones(size(J));
elseif yStart == yEnd,
   if xStart < xEnd
     I = [xStart:xEnd];
   else
     I = [xStart:-1:xEnd];
   end
   J = yStart*ones(size(I));
else
   a = (yStart-yEnd)/(xStart-xEnd);
   b = yStart-a*xStart;

   if abs(a) < 1
     if xStart < xEnd
```

```
        I = [xStart:xEnd];
      else
        I = [xStart:-1:xEnd];
      end
      J = round(a*I+b);

   else % iterate on J
      if yStart < yEnd,
        J = [yStart:yEnd];
      else
        J = [yStart:-1:yEnd];
      end

      I = round((J-b)/a);
   end
end
```

B.2 Selection of a Circular Region out of an Image

```
%================
% File Name: CircMask.m
%  by Meiqing He
% Last modified 6/7/1998
%----------------
%functionality:
% Select the pixels inside the specified circle.
% centerX & centerY: position of the center of the circle
% radius: radius of the circle.
% img:    image matrix.
%Only pixels of image within the specified circle will be remained, rest are
%set to zero.
%================

function imgCut = CircMask(centerX,centerY,radius, img)

imgCut = zeros(size(img));
for i=1:512,
  for j= 1: 512,
     tmp = (i-centerX)*(i-centerX)+(j-centerY)*(j-centerY);
     if(tmp<=radius*radius)
       imgCut(i,j)=img(i,j);
    end
  end
end
```

B.3 Selection of a Vector from a Circular Buffer

```
%=================
% File Name: CircWindow.m
% by Meiqing He
% Last modified 6/7/1998
%-----------------
%Functionality:
% Select a vector from the given vector.
% The original vector acts like a circular buffer
% winLen: length of the screen window.
% ind:    index of the center of the window.
%================

function selectVector=CircWindow(originalVec, ind,winLen)

n = length(originalVec);
if ind+winLen > n,
selectVector = [originalVec(ind-winLen:n) originalVec(1:(ind+winLen-n))];
elseif ind-winLen <1,
selectVector = [originalVec((n+ind-winLen):n) originalVec(1:ind+winLen)];
else selectVector = originalVec(ind-winLen :ind+winLen);
end
```

B.4 Determination of Adjacent Points

```
%============
% File Name: adjacent.m
% by Meiqing He
% Last modified  6/7/1998
% ------------
% functionality:
% To determine if two points are adjacent. 8-connectivity rule is applied
% val = 1, not adjacent
%     = 0, adjacent
%============

function val=adjcent(x1,y1,x2,y2)

dis=manhanttanDist(x1,y1,x2,y2);
if dis==1
val=1;
else if dis==2,
if abs(x2-x1)==1 & abs(y1-y2)==1
val=1;
else val=0;
end
else val=0;
end
end
```

B.5 Marking the Region inside a Contour

```
%===============
%File: ctr2region.m
% by Meiqing He
% Last modified  6/7/1998
%-----------------
% Functionality:
% For a given contour obtained from Snake Model, make it connected
%mark the region inside the contour with certain value
%===============
function [bmp, fnlI, fnlJ] = ctr2region(ctrI, ctrJ)

%Since the coordinates of the contour was rounded from floating points, there
%may be redundant points which have same interger coordinates.
%We need to abridge the contour.

len = length(ctrI);
ctrI=[ctrI ctrI(1)];
ctrJ=[ctrJ ctrJ(1)];
contour_x=ctrI(1);
contour_y=ctrJ(1);
for loop = 1:len,
  if ctrI(loop)~=ctrI(loop+1) | ctrJ(loop)~=ctrJ(loop+1)
    contour_x=[contour_x ctrI(loop+1)];
    contour_y=[contour_y ctrJ(loop+1)];
  end
end

fnlI = contour_x(1);
fnlJ = contour_y(1);
len = length(contour_x);

for i=1:len-1,
  isAdj =  adjacent(contour_x(i), contour_y(i),contour_x(i+1),contour_y(i+1));

  if isAdj==0
%link pixels in between, assume the contour is smooth enough. This is true
```

```
% if the contour data set is obtained from the Active contour snake model
%using linear interpolation

    [tmpI,tmpJ] = CalcLineXY(contour_x(i),contour_y(i),contour_x(i+1),
contour_y(i+1));


    lenTmp = length(tmpI);
    fnlI = [fnlI tmpI(2:lenTmp)];
    fnlJ = [fnlJ tmpJ(2:lenTmp)];
  else

    fnlI = [fnlI contour_x(i+1)];
    fnlJ = [fnlJ contour_y(i+1)];
  end

end

%assume the contour is convex
bmp = zeros(512,512);
minX = min(fnlI);
minY = min(fnlJ);
maxX = max(fnlI);
maxY = max(fnlJ);

geom_x = abs(maxX-minX);
geom_y = abs(maxY-minY);

if geom_x > geom_y
%iterate on x
for i = minX:maxX,
  ind = find(fnlI==i);
  startP = min(fnlJ(ind));
  endP = max(fnlJ(ind));
  for j=startP:endP,
    bmp(i,j)=1;
  end
```

```
        end

else for j=minY:maxY,
ind = find(fnlJ==j);
startP = min(fnlI(ind));
endP = max(fnlI(ind));
for i=startP:endP,
  bmp(i,j)=1;
end
     end

end
```

B.6 Difference between Two Vectors

```
%============
% File Name: err1.m
%by Meiqing He
% Lst modified  6/7/1998
%-------------
% function used to calculate the difference between two contour vectors
%===========

function err1=err1(x0,y0,x1,y1)
n=length(x0);
err1=0.;
for i=1:n
  err1=err1+(x0(i)-x1(i))*(x0(i)-x1(i))+(y0(i)-y1(i))*(y0(i)-y1(i));
 end
err1=sqrt(err1/n);
```

B.7 Calculation of Manhanttan Distance

```
%=============
% File Name: manhanttanDist.m
%by Meiqing He
% Last modified  6/7/1998
%-------------
%functionality:
% Calculate the Mahanttan distance between two points.
%=============
function distance = manhanttanDist(x1,y1,x2,y2)

vert = abs(x1-x2);
horz = abs(y1-y2);
distance = vert+horz;
```

B.8 Matrix A Generator

```
%==============
% File Name: mtrxa.m
%by Meiqing He
% Last modified  6/7/1998
%--------------
% function to generate coefficients matrix A
%=============

function a=mtrxa(n)

a=zeros(n,n);
a(1,1:3)=[8 -5 1];
a(1,n-1:n)=[1 -5];
a(2,1:4)=[-5 8 -5 1];
a(2,n)=1;
for i=4:n-2
  a(1,i)=0;
 end
for i=5:n-1
  a(2,i)=0;
 end
for i=1:n
  a(n,i)=a(1,n+1-i);
  a(n-1,i)=a(2,n+1-i);
 end
for i=3:n-2
  for j=1:n
    a(i,j)=0;
    a(i,i-2)=1;
    a(i,i-1)=-5;
    a(i,i)=8;
    a(i,i+1)=-5;
    a(i,i+2)=1;
  end
end
```

B.9 Components Labeling

```
%==============
%File Name: labeling.m
%by Meiqing He
% Last modified  6/7/1998
%---------------
% According to the connectivity to label the edge segments in an binary
% edge map.
% The generation of edge map can be modified. For instance, utilize other
% advanced edge detection and contour linking methods such as Snake Model,
% Canny-Deriche edge detector and singularity detection using wavelet
% transform
%4-connectivity is considered here during linking. Can be modified to
%8-connectivity region.
%===============

%Generate the input data, edge map, using simple edge detecion method.

DATAPATH = '/wasson/home/geoth/meiqing/research/imagebasalt/';
        filepath = [ DATAPATH '6398-11.txt'];
        fid = fopen(filepath,'r');

        data= fscanf(fid,'%g',[512 512]);
        fclose(fid);
data=data';
bmp=edge(data,20,'sobel'); % the bmp can also be generated by
   % reading the results generated by Snake
   % Model

%core part after edge map acquired
MAXEQUI = 20000;
equivalences = linspace(1, MAXEQUI, MAXEQUI);
label = 1;
        outBMP = zeros(size(bmp));
outBMP(1,:) = label*ones(1,512);
outBMP(:,1) = label*ones(512,1);
outBMP(:,512) = label*ones(512,1);
```

```
for i=2:512,
  for j=2:511,
    c = bmp(i,j);     %     d   a   e
    a = bmp(i-1,j); %   b   c
    b = bmp(i,j-1);
    d = bmp(i-1,j-1);  %  8-connected components
    e = bmp(i-1,j+1);  %
    y = (b==c);
    z = (a==c);
    if ~y & ~z,
      outBMP(i,j) = label;
      label = label +1;
    elseif ~z & y
      outBMP(i,j) = outBMP(i,j-1);
    elseif z & ~y
      outBMP(i,j) = outBMP(i-1,j);
    else
              outBMP(i,j) = outBMP(i-1,j);
      w = equivalences(outBMP(i,j-1));
      v = equivalences(outBMP(i-1,j));
      if v~=w
                 maxLabel = max([w v]);
         minLabel = min([w v]);
         ind = find(equivalences(1:label)==maxLabel);
                 equivalences(ind) = minLabel*ones(size(ind));
              end
            end
           end
end
numComponents = 1;
for i = 1:label,
  j= equivalences(i);
  if j==i
     equivalences(i) = numComponents;
     numComponents = numComponents+1;
  else
```

```
     equivalences(i) = equivalences(j);
  end
end
numComponents
for i=2:512,
 for j = 2:511,
           outBMP(i,j) = equivalences(outBMP(i,j));
 end
end

imagesc(data);
colormap('bone');
axis('image');
axis(axis);
hold on;
```

B.10 Table Look-up

```
%=============
% File Name: lookup.m
% by Meiqing He
% Last modified 6/7/1998
%-------------
%For a given integrated missing CT, find the apperture.
% a & b are parameters of the linear relation obtained by least square
% fitting the data from calibration experiment
%=============

function appt=lookup(integCT)
a = 1.398e5;
b = -194.5402;
appt = (integCT-b)/a;
```

B.11 Image Data Conversion

```
# =============
# File Name: roi_try.imp
# by Meiqing He
# Last modified 6/7/1998
#--------------
# script file for FPImage
# Convert the image file in IMG format to ascii format
# Input file name has to be in the fashion of 1978-01.img
# Supply the coordinates of upper-left and lower-right corner of the
# rectangular region which is to be clipped

input $fname "   enter first Input File:"
input $asc_file "Enter first asc output file"
input total "Enter total slices :"
input roi_x1  "  enter upper left X coordinate:"
input  roi_y1  "Enter upper left Y coordinate:"
input roi_x2  "Enter Lower Right X coordinate:"
input roi_y2  "Enter lower right Y coordinate:"

roi_type integer

*main
infile_a $fname
roi $asc_file

process begin
out = ina
process end
total = total - 1
if total < 1
exit
endif
nextfile $fname
nextfile $asc_file
goto main
exit
```

Appendix C

Programs Used for Calibration Purpose

- acc.m — calculate the integrated missing CT in an artificial fracture. The boundary is obtained from the Hough transform.
- houghBasalt.m — calculate the Hough transform of the edge map of the basalt core. The boundaries of the fracture can be estimated using this code.
- integBaslt.m — calculate the integrated missing CT in an artificial fracture. The boundary is obtained from the Active Contour method.
- lmsq.m — calculate the least square fit of the measured data having a linear relationship.
- snake4*.m — calculate the contours of the artificial fracture of different aperture size in basalt core.

The general procedure for calibration of the artificial fracture consists of following steps.

(1) Use houghBasalt.m to estimate the orientation of the fracture boundaries. This code can generate Figure 4.3.

(2) Modify snake4*.m to find the contour of the fracture. Figure 4.4 is one of the graphs resulting from this code.

(3) Use acc.m or integBaslt.m to calculate the integration of the missing CT over the gap. Figure 3.2 is one of the figures produced by acc.m. Code integBaslt.m integrates the missing CT based on the results of application of the Active Contour Model using snake4*.m.

(4) Use lmsq.m to do the least square fit to the measured relationship between integrated missing CT vs. aperture (see Figure 3.10).

C.1 Application of the Hough Transform to the Artificial Fracture

```
%================
% File Name: houghBasalt.m
% by Meiqing He
% Last modified 6/7/1998
%----------------
%calculate the Hough transform of the edge map of the basalt core.
%Select the two accumulator cells having largest value.
%Input data of the Hough transform is
%edge threshod chosen between 15 and 30
% parameters can be modified are:
% DATAPATH, filepath, thetaMin, thetaMax, center_x, center_y, radius
% numStep and numAccCell
%================

DATAPATH = '/wasson/home/geoth/meiqing/research/imagebasalt/';
        filepath = [ DATAPATH 'steph-01.txt'];
        fid = fopen(filepath,'r');

        data= fscanf(fid,'%g',[512 512]);
        fclose(fid);
data=data';

bmp=edge(data,20,'sobel');
```

```
imagesc(data);
colormap(bone);
colorbar;


%display the map of edge points
figure;
imagesc(128.*bmp);
colormap(bone);
axis('image');
colorbar;


figure;
i=find(bmp);
min_=min(min(data));
data(i)=min_.*ones(length(i),1);

imagesc(data);
colormap(bone);
axis('image')
colorbar;
%mask out the data outside the inner circle
center_x = 255;
center_y = 250;
radius = 60;
bmp_cut = CircMask(center_x, center_y, radius, bmp);

figure;
imagesc(128.*bmp_cut);
colormap(bone);
axis('image');
colorbar;

%hough transform
%thetaMin and thetaMax should be modified according the estimation of the
%range of the theta
```

```
thetaMin= -15;    %degree
thetaMax = 15;   %degree
numStep = 60;

points = find(bmp_cut);
y_ = floor(points./512)+1;
x_ = points-512.*(y_-1);    %row number
N= length(x_);

y = y_(1:2:N); % the sammpling rate 2 can be modified if the N is
       % not too big
x = x_(1:2:N);
N= length(x);

theta = linspace(thetaMin,thetaMax, numStep);
theta = theta*pi/180;

Rou = zeros(N,numStep);
for loop = 1:N,
Rou(loop,:) = x(loop)* cos(theta)+y(loop)*sin(theta);
end

RouMin = min(min(Rou))
RouMax = max(max(Rou))

figure;
for loop = 1:N,
axis('ij');
plot(theta, Rou(loop,:), 'w');
hold on;
end
title('Hough transform');
xlabel('$\theta$');
ylabel('rou');

numAccCell = 200
Acc = zeros(numAccCell,numStep);
```

```
resolution = (RouMax-RouMin)/(numAccCell-1);
for loop1 = 1:numStep,
  for loop2 = 1: N,

   index = floor((Rou(loop2,loop1)-RouMin)/resolution)+1;
   if index<=0 | index> numAccCell
    index
    loop1
    end
    Acc(index, loop1) = Acc(index, loop1)+1;
  end
end

idAll = find(Acc>30);
idAll_y = floor(idAll/numAccCell)+1
idAll_x = idAll-(idAll_y-1)*numAccCell

maxCount = max(max(Acc))
id = find(Acc==maxCount);
id_min = min(id);    %there may be two or more points aremaximal
id_y =  floor(id_min/numAccCell)+1;
id_x = id_min-numAccCell*(id_y-1);    %row number
theta1 = theta(id_y)
Rou1 = id_x*resolution+RouMin

Acc(id_min)=0;
%find second largest one
maxCount_ = max(max(Acc));
id = find(Acc==maxCount_);
id_min = min(id);
id_y = floor(id_min/numAccCell)+1;
id_x = id_min-numAccCell*(id_y-1);    %row number
theta2 = theta(id_y)
Rou2 = id_x*resolution+RouMin
```

C.2 Integration of the Missing CT

```
%===========
%File Name: acc.m
% by Meiqing He
% Last modified 6/7/1998
%---------------
%
%integrate the missing mass over the gap.
%using results from the Hough transform (theta1 and theta2)
%==============

%read and display the image
DATAPATH = '/wasson/home/geoth/meiqing/research/imagebasalt/';
        filepath = [ DATAPATH '6398-11.txt'];
        fid = fopen(filepath,'r');

        data= fscanf(fid,'%g',[512 512]);
        fclose(fid);
data=data';
imagesc(data);
colormap(bone);
axis('image')
colorbar;
title('(a)');
% perform the edge detection
bmp=edge(data,15,'sobel');
%display the edge map
figure;
imagesc(128*bmp);
colormap(bone);
axis('image');
colorbar;
title('(b)');

%j is between 1, 512
theta1 = 0.0754;
Rou1 = 303.6699;
```

```
theta2 = 0.0843;
Rou2 = 297.6210;

[i1,j1] = CalcLine(Rou1, theta1); %use most sure slope
ind = (j1-1)*512+i1;
% bmp(ind) = ones(512,1);

[i2,j2] = CalcLine(Rou2, theta2);
ind = (j2-1)*512+i2;

%plot edge map and two parallel lines
figure;
imagesc(128*bmp);
colormap(bone);
axis('image')
colorbar;
axis(axis);
hold on;
plot(j1,i1);
hold on;
plot(j2,i2);
title('(c)')

%set the Rou for the vertical line, can be modified to change its position
RouVertical = 230;
%calculate the orientation of the vertical line
if theta2>0
thetaVertical = theta2-pi/2;
else thetaVertical = theta2+pi/2;
end
[i,j] = CalcLine(RouVertical, thetaVertical);
%plot the vertical line
hold on;
plot(j,i);

ind = (j-1)*512+i;
```

```
perpenLine = data(ind);

figure;
hist(data(ind));

%plot the profile
figure;
tmp = 2200*ones(512,1);
plot(i,perpenLine,'w',i,tmp,'r');
title('(d)');

% calculate the intersect of two lines
xStart = round((RouVertical*sin(theta2)-Rou2*sin(thetaVertical))/
(cos(thetaVertical)*sin(theta2)-cos(theta2)*sin(thetaVertical)));
yStart = round((RouVertical*cos(theta2)-Rou2*cos(thetaVertical))/
(sin(thetaVertical)*cos(theta2)-sin(theta2)*cos(thetaVertical)));
startPValue = data(xStart, yStart)

xEnd = round((RouVertical*sin(theta2)-Rou1*sin(thetaVertical))/
(cos(thetaVertical)*sin(theta2)-cos(theta2)*sin(thetaVertical)));
yEnd = round((RouVertical*cos(theta2)-Rou1*cos(thetaVertical))/
(sin(thetaVertical)*cos(theta2)-sin(theta2)*cos(thetaVertical)));
endPValue = data(xEnd, yEnd)

[startPtOnPerpen,iStart] = min(abs(perpenLine-startPValue));
if length(iStart)>1
  startPt = min(iStart);
  endPt = max(iStart);
else
[startPtOnPerpen,iEnd] = min(abs(perpenLine-endPValue));
startPt = min([iStart iEnd]);
endPt = max([iStart iEnd]);
end
integral = sum(2200-perpenLine(startPt:endPt))
```

```
%==============
% File Name: integBasalt.m
% by Meiqing He
% Last modified 6/7/1998
%---------------
%Integrate the missing mass.
%Use the contour obtained from the active contour model
%Use global Hough transform obtained from houghBasalt.m
%The value of the pixel in boundary is used as reference density level in
%the process to calculate the missing CT.
%=================

DATAPATH = '/wasson/home/geoth/meiqing/research/imagebasalt/';
        filepath = [ DATAPATH 'steph-01.txt'];
        fid = fopen(filepath,'r');

        data= fscanf(fid,'%g',[512 512]);
        fclose(fid);
data=data';

%read the contour data
CTRPATH = '/wasson/home/geoth/meiqing/research/snake/';
filepath =[CTRPATH 'try.reslt'];
fid = fopen(filepath, 'r');
ctr = fscanf(fid, '%g',[2,inf]);
fclose(fid);
ctrJ = round(ctr(1,:));
ctrI = round(ctr(2,:));

[bmp,fnlCtrI, fnlCtrJ] = ctr2region(ctrI, ctrJ);

% rou and theta are obtained from the Hough transform
%This code is used especially for fracture with even aperture
%Two stright lines feature in intensity image

theta1 = 0.02;
theta2 = 0.02;
```

```
imagesc(data);
colormap(bone);
axis('image')
colorbar;
axis(axis);
hold on;
plot(fnlCtrJ,fnlCtrI,'r');

figure;
imagesc(bmp*128);
colormap(bone);
axis('image')
colorbar;
axis(axis);
hold on;
plot(fnlCtrJ,fnlCtrI);

RouVertical = -250; % pick a location of the profile

%calculate the thetaVertical
if theta2>0
thetaVertical = theta2-pi/2;
else thetaVertical = theta2+pi/2;
end

%calculate the profile line
[i,j] = CalcLine(RouVertical, thetaVertical);

hold on;
plot(j,i);

len = length(i);
integ =0;
maxDens = -1024;
numPixInGap=0;
for loop = 1:len,
```

```
  if bmp(i(loop),j(loop))==1
  integ = integ+data(i(loop),j(loop));
  numPixInGap = numPixInGap+1;
  if maxDens < data(i(loop),j(loop))
     maxDens = data(i(loop),j(loop));
     maxDens_i = i(loop);
     maxDens_j = j(loop);
   end
  end
end
%use maxDens as reference CT number in doing
integ = maxDens*numPixInGap-integ
```

C.3 Least Square Fitting

```
%================
% File Name: lmsq.m
% by Meiqing He
% Last modified 6/7/1998
%----------------
% x & y are measured data
%least mean square method to estimate the slope and intercect
%====================
x = [0.001 0.002 0.004 0.008 0.012 0.02 0.04];
y = [96 256 338 690 1404.3 2515 5501.3];
sumX = sum(x);
sumY = sum(y);

N= length(x)

sumX2 = sum(x.^2);
sumXY = sum(x.*y);
tmp = sumX * sumX - N*sumX2;

if ~(tmp==0)
a = (sumX*sumY-N*sumXY)/tmp
b = - (sumX2 * sumY-sumXY*sumX)/tmp
```

```
end

plot(x,y, '.');
hold on;
x1 = linspace(0.001, 0.04, 10);
y1 = a*x1+b;
plot(x1,y1);
legend('Measured data', 'LS approximation');
title('Integrated missing CT vs. aperture');
xlabel('aperture(in)');
ylabel('integrated missing CT');
y_ = a*x+b;
error = sqrt(sum((y-y_).^2))
```

C.4 Example of Setting Initial Contour and Solve the Snake

```
%================
%File:snake4_11.m
%-------------
%reading image of basalt core used in calibration and apply edge detection
%operator.
%call out_in to solve the contour
%===============
DATAPATH =  '/wasson/home/geoth/meiqing/research/imagebasalt/';
        filepath = [ DATAPATH '6398-11.txt'];
        fid = fopen(filepath,'r');

        data= fscanf(fid,'%g',[512 512]);
        fclose(fid);
data=data';
bmp=edge(data,20,'sobel');


x1 = 285;
y1 = 140;
x2 = 275;
y2 = 295;
[iInd1,jInd1] = CalcLineXY(x1,y1,x2,y2);
x3 = 285;
y3 = 300;
[iInd2, jInd2] = CalcLineXY(x2,y2,x3,y3);
x4 = 290;
y4 = 160;
[iInd3, jInd3] = CalcLineXY(x3,y3,x4,y4);
[iInd4, jInd4] = CalcLineXY(x4,y4,x1,y1);

jInd = [jInd1 jInd2 jInd3 jInd4];
iInd = [iInd1 iInd2 iInd3 iInd4];

%display the initial contour abainst the image and the edge map
```

```
imagesc(data);
colormap('bone');
axis('image');
colorbar;
axis(axis);
hold on;
plot(jInd,iInd,'r.');
title('(a) Initial contour');

figure;
imagesc(128.*bmp);
colormap('bone');
axis('image');
axis(axis);
hold on;
plot(jInd,iInd,'r.');


        [ctrI,ctrJ] = out_in(jInd', iInd', data);  %

figure;
imagesc(data);
colormap('bone');
axis('image');
colorbar;
axis(axis);
hold on;
plot([ctrJ;ctrJ(1)],[ctrI;ctrI(1)],'r.'), title('converged edge');

%display the edge map the the result of the Snake Model
figure;
imagesc(128.*bmp);
axis('image');
colormap('bone');
axis(axis);
hold on;
plot([ctrJ;ctrJ(1)],[ctrI;ctrI(1)],'r.');
```

```
%output the contour data
z = [ctrJ ctrI];
fid = fopen('contour11.reslt', 'w');
fprintf(fid,'%g \t %g \n', z');
fclose(fid);
```

Appendix D

Programs used for Geysers Sample

- dispwholectr.m — display all the detected contours against the core CT image of Geysers sample.
- snake3No*.m — calculate all the contours in the core CT image of Geysers sample one by one.
- integrateHgh.m — calculate the aperture of one natural fracture at a specific location.
- integHghSts.m — calculate the aperture of a fracture at each pixel on the contour.

The general procedure for estimating the aperture of the natural fracture includes following steps:

(1) Modify snake3No*.m to find the contour of a natural fracture. Figure 4.7(a) was obtained this way.

(2) Use integrateHgh.m to calculate the aperture at specific location of a fracture. Figure 4.7(b) illustrates the location of profile.

D.1 Calculation of Fracture Aperture in Geysers Sample

```
%==================
%File Name: integrateHgh.m
% by Meiqing He
% Last modified 6/7/1998
%---------------
%integrate the missing mass
%use the contour obtained from the active contour model
%Global Hough transform is used to estimate the orientation of the fracture
%The aperture is calculated at a location specified by RouVertical.
%===================

%read the image and contour data
load cleanCore1
edgeMap=edge(cleanCore,15,'sobel');

CTRPATH = '/wasson/home/geoth/meiqing/research/snake/';
filepath =[CTRPATH 'GeysCtr1.reslt'];

fid = fopen(filepath, 'r');
ctr = fscanf(fid, '%g',[2,inf]);
fclose(fid);

ctrJ = round(ctr(1,:));
ctrI = round(ctr(2,:));

[bmp, fnlCtrI, fnlCtrJ] = ctr2region(ctrI, ctrJ);

thetaMin = 30;
thetaMax = 60;
numTheta = 60;
numAccCell = 200;
[Acc, Rou ,theta] = hough(thetaMin, thetaMax, numTheta, numAccCell, ctrI, ctrJ);
```

```
% rou and theta are obtained from the Hough transform
%This code is used especially for fracture with relative uniform orientation


imagesc(cleanCore);
colormap(bone);
axis('image')
colorbar;
axis(axis);
hold on;
plot(fnlCtrJ,fnlCtrI);

figure;
imagesc(bmp*128);
colormap(bone);
axis('image')
colorbar;
axis(axis);
hold on;
plot(ctrJ,ctrI);
%statistical analysis on the profile
idAll = find(Acc>30);
idAll_y = floor(idAll/numAccCell)+1
idAll_x = idAll-(idAll_y-1)*numAccCell

maxCount = max(max(Acc))
id = find(Acc==maxCount);
id_min = min(id);    %there may be two or more points aremaximal
id_y =  floor(id_min/numAccCell)+1;
id_x = id_min-numAccCell*(id_y-1);    %row number
theta1 = theta(id_y)
% Rou1 = id_x*resolution+RouMin

RouVertical = 100; % pick a location of the profile
%calculate the thetaVertical
if theta1>0
thetaVertical = theta1-pi/2;
```

```
else thetaVertical = theta1+pi/2;
end

%calculate the profile line
[i,j] = CalcLine(RouVertical, thetaVertical);

hold on;
plot(j,i);

len = length(i);
integ =0;
maxDens = -1024;
numPixInGap=0;
for loop = 1:len,
  if i(loop)>0 & i(loop)<=512 &j(loop)>0 & j(loop)<=512
    if bmp(i(loop),j(loop))==1
    integ = integ+cleanCore(i(loop),j(loop));
    numPixInGap = numPixInGap+1;
      if maxDens < cleanCore(i(loop),j(loop))
       maxDens = cleanCore(i(loop),j(loop));
       maxDens_i = i(loop);
       maxDens_j = j(loop);
      end
    end
  end
end
%use maxDens as reference CT number in doing
integ = maxDens*numPixInGap-integ
appt = lookup(integ)
```

```
%====================
%File Name: integHghSts.m
% by Meiqing He
% Last modified 6/7/1998
%---------------
%integrate the missing mass at each pixel on contour
%Hough transform is local, the running length is 10~20
%use the contour obtained from the active contour model
%====================

%read the image and contour data
load cleanCore1
edgeMap=edge(cleanCore,15,'sobel');

CTRPATH = '/wasson/home/geoth/meiqing/research/snake/';
filepath =[CTRPATH 'try.reslt'];
fid = fopen(filepath, 'r');
ctr = fscanf(fid, '%g',[2,inf]);
fclose(fid);

ctrJ = round(ctr(1,:));
ctrI = round(ctr(2,:));

[bmp, fnlCtrI, fnlCtrJ] = ctr2region(ctrI, ctrJ);
imagesc(128*bmp);
colormap(bone);
axis('image')
colorbar;
axis(axis);
hold on;
plot(fnlCtrJ,fnlCtrI);

outFid = fopen('tryApp.dat','w');
thetaMin = -89;
thetaMax = 90;
numTheta = 180;
numAccCell = 100;
```

```
runLength = 5; % be sure to set runLength<lengthCtr
lengthCtr = length(fnlCtrI)
for loop =1:lengthCtr,
  tmpI = CircWindow(fnlCtrI,i,runLength);
  tmpJ = CircWindow(fnlCtrJ,i,runLength);
  [Acc, Rou ,theta] = hough(thetaMin, thetaMax, numTheta, numAccCell,tmpI,tmpJ);

% rou and theta are obtained from the Hough transform
% Using changing rou and theta to account for the changingaperture and
%orientation

%Select the best local slope at current pixel
  maxCount = max(max(Acc));
  id = find(Acc==maxCount);
  id_y = floor(id/numAccCell)+1;
  id_x = id-numAccCell*(id_y-1);    %row number
  slopeAtLoop = localSlope(fnlCtrI,fnlCtrJ,loop)
  length(id)
   if length(id)==1
      thetaBest = theta(id_y)
   else
errTheta=1e3;
for loop2=1:length(id),
         errTmp = abs(theta(id_y(loop2))-slopeAtLoop);

          if errTmp < errTheta,
     errTheta=errTmp;
             id_fnl = loop2;
           end
end
      thetaBest = theta(id_y(id_fnl))
   end

%calculate the thetaVertical
  if thetaBest>0
    thetaVertical = thetaBest-pi/2;
  else thetaVertical = thetaBest+pi/2;
```

```
  end

  RouVertical = fnlCtrI(loop)*cos(thetaVertical)
+fnlCtrJ(loop)*sin(thetaVertical); % pick a location of the profile

%calculate the profile line
  [i,j] = CalcLine(RouVertical, thetaVertical);

  if floor(loop/20)*20==loop,
    hold on;
    plot(j,i);
  end

  len = length(i);
  integ =0;
  maxDens = -1024;
  numPixInGap=0;
  for loop1 = 1:len,
    if i(loop1)>0 & i(loop1)<=512 &j(loop1)>0 & j(loop1)<=512
      if bmp(i(loop1),j(loop1))==1
        integ = integ+cleanCore(i(loop1),j(loop1));
        numPixInGap = numPixInGap+1;
        if maxDens < cleanCore(i(loop1),j(loop1))
          maxDens = cleanCore(i(loop1),j(loop1));
          maxDens_i = i(loop1);
          maxDens_j = j(loop1);
        end
      end
    end
  end
%use maxDens as reference CT number in doing
integ = maxDens*numPixInGap-integ;
appt = lookup(integ);
fprintf(outFid,'%g\n',appt);

end
```

D.2 Displaying the Contours

```
%================
%File Name: dispwholectr.m
% by Meiqing He
% Last modified 6/7/1998
%-----------------
%display the entire contour of the Geysers core against the original image
%================

load ../../matlab/cleanCore1



subplot(221);
imagesc(cleanCore);
colormap(bone);
axis('image');
colorbar;
title('(a)');

subplot(222);
bmp = edge(cleanCore, 20, 'sobel');
imagesc(128.*bmp);
axis('image');
colormap(bone);
title('(b)');

subplot(223);
imagesc(cleanCore);
colormap(bone);
axis('image');
colorbar;
axis(axis);
hold on;

for i=1:12,
fileNm=sprintf('GeysCtr%d.reslt',i);
fid = fopen(fileNm, 'r');
```

```
data = fscanf(fid, '%g', [2,inf]);
x0 = round(data(1,:));
y0 = round(data(2,:));
plot(x0,y0);
hold on;
end
title('(c)');
```