New Image Acquisition Algorithms

for

Atomic Force Microscope

by

Po Kin Cheng

A thesis
presented to University of Manitoba
in partial fulfillment of the
requirements for the degree of
Master of Science
in
Electrical Engineering

Winnipeg, Manitoba, 1994

ISBN 0-612-13027-4

Name CHENG PO KIN

Dissertation Abstracts International is arranged by broad, general subject categories. Please select the one subject which most nearly describes the content of your dissertation. Enter the corresponding four-digit code in the spaces provided.

ELECTRONICS and ELECTRICAL ENGINEERING
SUBJECT TERM

0544
SUBJECT CODE

U·M·I

Subject Categories

THE HUMANITIES AND SOCIAL SCIENCES

COMMUNICATIONS AND THE ARTS
Architecture 0729
Art History 0377
Cinema 0900
Dance 0378
Fine Arts 0357
Information Science 0723
Journalism 0391
Library Science 0399
Mass Communications 0708
Music 0413
Speech Communication 0459
Theater 0465

EDUCATION
General 0515
Administration 0514
Adult and Continuing 0516
Agricultural 0517
Art 0273
Bilingual and Multicultural 0282
Business 0688
Community College 0275
Curriculum and Instruction 0727
Early Childhood 0518
Elementary 0524
Finance 0277
Guidance and Counseling 0519
Health 0680
Higher 0745
History of 0520
Home Economics 0278
Industrial 0521
Language and Literature 0279
Mathematics 0280
Music 0522
Philosophy of 0998
Physical 0523
Psychology 0525
Reading 0535
Religious 0527
Sciences 0714
Secondary 0533
Social Sciences 0534
Sociology of 0340
Special 0529
Teacher Training 0530
Technology 0710
Tests and Measurements 0288
Vocational 0747

LANGUAGE, LITERATURE AND LINGUISTICS
Language
- General 0679
- Ancient 0289
- Linguistics 0290
- Modern 0291

Literature
- General 0401
- Classical 0294
- Comparative 0295
- Medieval 0297
- Modern 0298
- African 0316
- American 0591
- Asian 0305
- Canadian (English) 0352
- Canadian (French) 0355
- English 0593
- Germanic 0311
- Latin American 0312
- Middle Eastern 0315
- Romance 0313
- Slavic and East European 0314

PHILOSOPHY, RELIGION AND THEOLOGY
Philosophy 0422
Religion
- General 0318
- Biblical Studies 0321
- Clergy 0319
- History of 0320
- Philosophy of 0322

Theology 0469

SOCIAL SCIENCES
American Studies 0323
Anthropology
- Archaeology 0324
- Cultural 0326
- Physical 0327

Business Administration
- General 0310
- Accounting 0272
- Banking 0770
- Management 0454
- Marketing 0338

Canadian Studies 0385
Economics
- General 0501
- Agricultural 0503
- Commerce-Business 0505
- Finance 0508
- History 0509
- Labor 0510
- Theory 0511

Folklore 0358
Geography 0366
Gerontology 0351
History
- General 0578
- Ancient 0579
- Medieval 0581
- Modern 0582
- Black 0328
- African 0331
- Asia, Australia and Oceania 0332
- Canadian 0334
- European 0335
- Latin American 0336
- Middle Eastern 0333
- United States 0337

History of Science 0585
Law 0398
Political Science
- General 0615
- International Law and Relations 0616
- Public Administration 0617

Recreation 0814
Social Work 0452
Sociology
- General 0626
- Criminology and Penology 0627
- Demography 0938
- Ethnic and Racial Studies 0631
- Individual and Family Studies 0628
- Industrial and Labor Relations 0629
- Public and Social Welfare 0630
- Social Structure and Development 0700
- Theory and Methods 0344

Transportation 0709
Urban and Regional Planning 0999
Women's Studies 0453

THE SCIENCES AND ENGINEERING

BIOLOGICAL SCIENCES
Agriculture
- General 0473
- Agronomy 0285
- Animal Culture and Nutrition 0475
- Animal Pathology 0476
- Food Science and Technology 0359
- Forestry and Wildlife 0478
- Plant Culture 0479
- Plant Pathology 0480
- Plant Physiology 0817
- Range Management 0777
- Wood Technology 0746

Biology
- General 0306
- Anatomy 0287
- Biostatistics 0308
- Botany 0309
- Cell 0379
- Ecology 0329
- Entomology 0353
- Genetics 0369
- Limnology 0793
- Microbiology 0410
- Molecular 0307
- Neuroscience 0317
- Oceanography 0416
- Physiology 0433
- Radiation 0821
- Veterinary Science 0778
- Zoology 0472

Biophysics
- General 0786
- Medical 0760

EARTH SCIENCES
Biogeochemistry 0425
Geochemistry 0996
Geodesy 0370
Geology 0372
Geophysics 0373
Hydrology 0388
Mineralogy 0411
Paleobotany 0345
Paleoecology 0426
Paleontology 0418
Paleozoology 0985
Palynology 0427
Physical Geography 0368
Physical Oceanography 0415

HEALTH AND ENVIRONMENTAL SCIENCES
Environmental Sciences 0768
Health Sciences
- General 0566
- Audiology 0300
- Chemotherapy 0992
- Dentistry 0567
- Education 0350
- Hospital Management 0769
- Human Development 0758
- Immunology 0982
- Medicine and Surgery 0564
- Mental Health 0347
- Nursing 0569
- Nutrition 0570
- Obstetrics and Gynecology 0380
- Occupational Health and Therapy 0354
- Ophthalmology 0381
- Pathology 0571
- Pharmacology 0419
- Pharmacy 0572
- Physical Therapy 0382
- Public Health 0573
- Radiology 0574
- Recreation 0575
- Speech Pathology 0460
- Toxicology 0383

Home Economics 0386

PHYSICAL SCIENCES
Pure Sciences
Chemistry
- General 0485
- Agricultural 0749
- Analytical 0486
- Biochemistry 0487
- Inorganic 0488
- Nuclear 0738
- Organic 0490
- Pharmaceutical 0491
- Physical 0494
- Polymer 0495
- Radiation 0754

Mathematics 0405
Physics
- General 0605
- Acoustics 0986
- Astronomy and Astrophysics 0606
- Atmospheric Science 0608
- Atomic 0748
- Electronics and Electricity 0607
- Elementary Particles and High Energy 0798
- Fluid and Plasma 0759
- Molecular 0609
- Nuclear 0610
- Optics 0752
- Radiation 0756
- Solid State 0611

Statistics 0463

Applied Sciences
Applied Mechanics 0346
Computer Science 0984
Engineering
- General 0537
- Aerospace 0538
- Agricultural 0539
- Automotive 0540
- Biomedical 0541
- Chemical 0542
- Civil 0543
- Electronics and Electrical 0544
- Heat and Thermodynamics 0348
- Hydraulic 0545
- Industrial 0546
- Marine 0547
- Materials Science 0794
- Mechanical 0548
- Metallurgy 0743
- Mining 0551
- Nuclear 0552
- Packaging 0549
- Petroleum 0765
- Sanitary and Municipal 0554
- System Science 0790

Geotechnology 0428
Operations Research 0796
Plastics Technology 0795
Textile Technology 0994

PSYCHOLOGY
General 0621
Behavioral 0384
Clinical 0622
Developmental 0620
Experimental 0623
Industrial 0624
Personality 0625
Physiological 0989
Psychobiology 0349
Psychometrics 0632
Social 0451

NEW IMAGE ACQUISITION ALGORITHMS FOR ATOMIC FORCE MICROSCOPE

BY

PO KIN CHENG

A Thesis submitted to the Faculty of Graduate Studies of the University of Manitoba in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

I hereby declare that I am the sole author of this thesis.

I authorize the University of Manitoba to lend this thesis to other institutions or individuals for the purpose of scholarly research.

I further authorize the University of Manitoba to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

ABSTRACT

The behavior of the digital feedback control system of an atomic force microscope (AFM) and the development of novel AFM scan algorithms are described. A digital AFM feedback control system model is developed in which the system is constructed by cascading the AFM with a digital proportional-integral (PI) compensator. The PI compensator is composed by a proportional controller and an integral controller in parallel and is software implemented inside a digital signal processor, DSP 56001 from Motorola. The PI compensator manipulates the bicell detector signal received from the AFM and feeds back the manipulated signal to the AFM to adjust the vertical position of the sample accordingly via a piezo tube scanner. The rate of adjusting the vertical position of the sample can be increased by increasing the proportional gain and integral gain of the PI compensator. However, the system will become unstable if these two gains are too high. By adjusting the proportional gain and integral gain manually, we usually cannot find the value of these two gains that give the system the fastest and stable system response to adjust the vertical position of the sample. An automated scanner is developed which uses genetic algorithm to search for the optimal gains that gives the system the fastest stable system response. In addition, the automated scanner is able to vary the scan speed according to the surface roughness of the sample surface and the image data acquired has a finite known error margin. Attempts are made to find an improved alternative control algorithm to the proportional-integral-derivative (PID) compensator. Frequency response techniques and root-locus design techniques are used but they both do not produce an improved compensator. Instead, two alternative scan algorithms, predictor corrector and ramp scanner, are developed which improve the performance of the AFM system using PID compensator.

ACKNOWLEDGMENTS

First of all, I would like to thank my advisor, Dr. D.J. Thomson, for his kindly patience and encouragement throughout the past 30 months of my work on this thesis project. Also, I would like to thank Dr. G.C. McGonigal for his guidance and advice during the early days of the work in the AFM system enhancement. Special thanks are given to Dr. K.L. Westra for sharing his precious experience on the operation of the AFM and on the control system analysis. Finally, K. Yackoboski, R. Ng, S. Delorenzi, J. Nxumalo, and E.A. Lemus are the names that would not be forgotten for their helpful suggestions and efforts on this thesis project.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS

ADC : Analog-to-Digital Converter

AFM : Atomic Force Microscope / Atomic Force Microscopy

DAC : Digital-to-Analog Converter

DSP : Digital Signal Processor

GA : Genetic Algorithm

PI Compensator : Proportional-Integral Compensator

PID Compensator : Proportional-Integral-Derivative Compensator

STM : Scanning Tunneling Microscope / Scanning Tunneling Microscopy

CVD : Chemical Vapor Deposition

CHAPTER 1

INTRODUCTION

1.1 Prologue

Atomic force microscope (AFM) is a powerful tool for imaging surface structure of various samples. It was invented by G. Binnig, C. F. Quate, and the co-workers in 1986 [1]. The AFM is a stylus-like instrument and works like a record player. A tip is pressed against a sample surface and scanned over it. The force between the tip and the surface maps out the surface topography of the sample during scanning. This kind of AFM is later renamed as contact-mode AFM because non-contact mode AFM is developed afterward which uses longer-range force such as Van der Waals force as the detecting force [2,3]. However, only contact mode AFM is addressed in this thesis and the name AFM will be used for simplicity. Since force is used as the medium for detecting the surface contour, unlike its predecessor scanning tunneling microscope (STM), AFM is able to image insulators as well as conductors. Recently, researches attempt to use AFM to acquire images of biological systems such as DNA and living cells [4] which require small tracking force.

The success of STM and AFM has led to a massive hunting for different detection media besides tunneling current and atomic force in STM and AFM respectively. Various scanning microscopes have been developed and they use different detecting force such as Van der Waals forces[2,3], magnetic forces[5,6] and electrostatic forces[7].

1.1.1 About the AFM

The AFM is the most popular and well-developed scanning probe microscopy

besides STM. A very sharp tip mounted on a soft cantilever is pressed onto the sample surface. If the spring constant of the cantilever is small enough, the tip will follow the surface contour as it moves over the surface without damaging the surface. The movement of the tip is shown in figure 1.1 and the vertical displacement of the tip can be captured by a sensor.

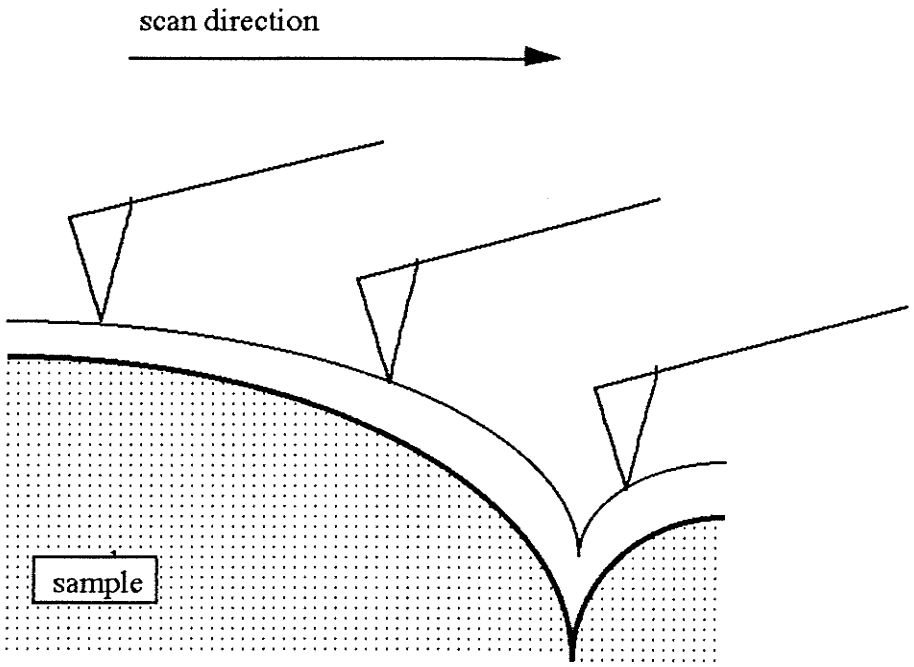

Figure 1.1 A simple diagram to show that the tip will follow the contour of the sample surface when the tip scan across the surface.

However, the sensor output is seldom used as the data for the surface topography image since the surface roughness of the sample may exceed the dynamic range of the sensor or the sensor may have nonlinear behavior. Instead, if the vertical position of the sample (which will be named 'sample height' for the rest of this thesis) can be adjusted to

follow the surface contour, then the tip will remain motionless in the vertical direction (or in other words, the sensor signal can be kept constant) during scanning as shown in figure 1.2. In figure 1.2, the tip moves from point 'a' to point 'b'. If the tip can be held at the same vertical level by adjusting the sample height accordingly, then the sample height is following the surface from the height 'Ha' at point 'a' to height 'Hb' at point 'b'. In addition, since the sample height is following the surface contour, the signal that controls the movement of the sample height can be used as the image data for the surface topography image. This is the idea how the AFM acquires an image of the sample surface topography.

Using a feedback control mechanism, the feedback system manipulates the sensor signal and adjusts the sample height accordingly so that the tip can be held motionless in the vertical direction during scanning. If the output of the feedback system is able to follow the surface contour, then the feedback output can be used to compose the image of the surface topography. Figure 1.3 shows the schematic diagram of a typical AFM system. Although there are many ways of building the AFM system, the following section gives a brief review of the AFM system using the schematic diagram shown in figure 1.3 as a model.

1.1.2 About the AFM system

As the schematic diagram in figure 1.3 reflects, the sample sits on a micropositioner that controls the sample movement in x, y, and z direction. The x and y direction movement sets the size of scanning (scanning is accomplished by moving the sample, instead of the tip, along x and y direction) while the z direction displacement adjusts the sample height. A tube scanner is used as the micropositioner in the system shown in figure 1.3. The tube scanner is a hollow cylinder-like device made from a piezoelectric ceramic transducer (it will be simply called "piezo" in rest of this thesis.) The piezo can be extended or contracted by varying the voltage applied to it. There are five electrodes on the scanner, four of them on the outside of the tube and the fifth one on the

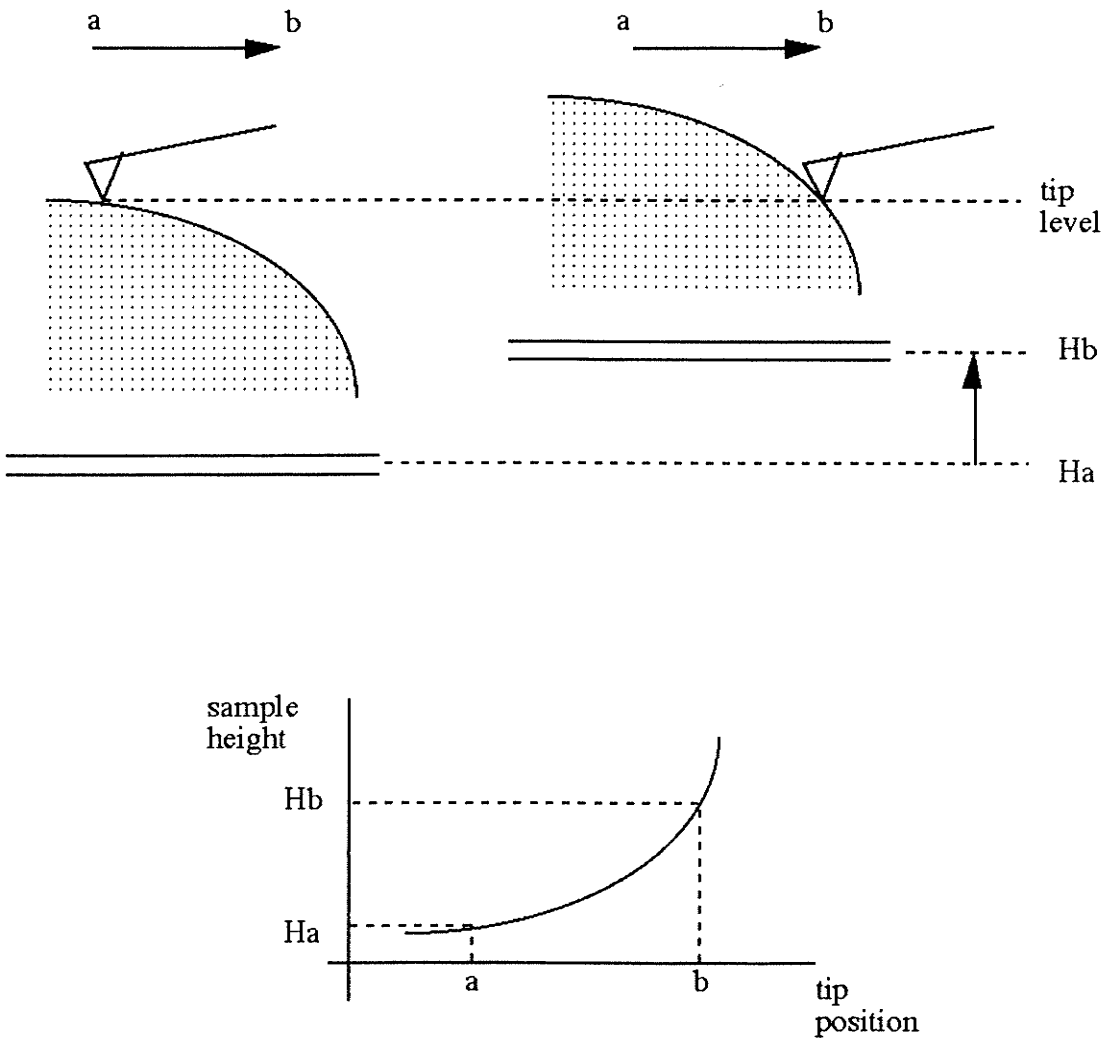

Figure 1.2 Diagrams that show how the AFM tracks the surface contour of a sample.

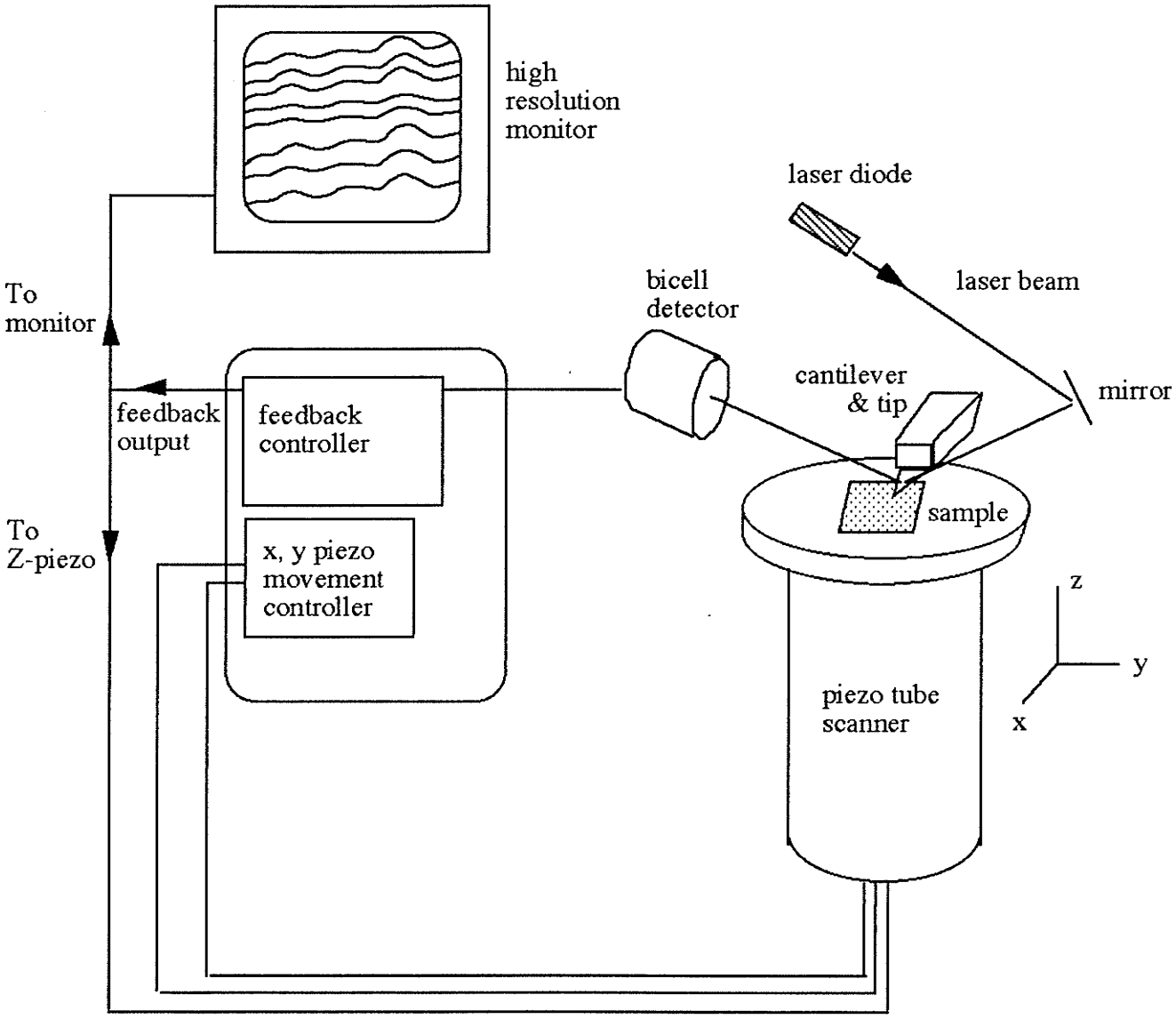

Figure 1.3 Schematic diagram of the AFM system

inner surface of the tube. The outer electrodes control the x and y movement and the inner electrode controls the z movement [8]. The inner electrode will be termed "Z-piezo" in the rest of this thesis.

The cantilever is a "V" shaped structure with a pyramid at the bottom of the front end. The apex of the pyramid is the tip that is used to scan over the surface. The cantilever and the tip are usually made from silicon nitride or similar materials using standard micromachining methods [9].

During scanning, the tip moves up and down following the surface contour and the vertical displacement of the tip deflects the cantilever by certain angle. The deflection of the cantilever reflects the surface contour and can be detected by various methods [10 - 12]. The one shown in figure 1.3 is called the beam bounce method. A laser beam is focused on the front end of the cantilever and is reflected to a bicell detector. The bicell detector is divided into two halves, and the difference of the output signal from these two halves of the detector are used. As the cantilever deflects, the position of the reflected beam falling on the detector drifts, and in turn, the signal difference changes accordingly. Before scanning, the tip is pressed onto the surface until the signal from the bicell detector equals a constant called 'set-point'. The 'set-point' is a reference point for scanning and represents the force that the tip presses onto the surface. During scanning, the difference between the bicell detector signal and the set-point, which is called 'error signal', is used for calculating the feedback output. The error signal is fed back to adjust the sample height via a feedback system to keep the error signal equal to zero (or in other words, the bicell signal equals the set point value). If the feedback system reacts fast enough, the tip will be virtually motionless in vertical direction and the feedback output represents the corrugation of the surface contour. The value of the feedback output is also sent to the computer for display as an image. The image acquired is the top view of the surface topography. There are two common image representations, one of them is the line plotting and the other is the gray-

level representation.

1.2 Origin of the thesis - motivations and deviations

The feedback system is one of the major components in the AFM system. The function of the feedback system is to compensate for the variation of the signal coming from the bicell detector and form the image of the sample topography. Almost all existing AFM systems use proportional-integral-derivative (PID) compensator in the feedback system since the PID compensator is supposed to provide fast response with zero steady-state error at the output. Because of the sensitivity to high frequency noise, the derivative controller is omitted and PI compensator is used in our system instead. This thesis project attempts to investigate alternative control algorithm(s) besides the PI compensator which may offer a faster and more stable system response.

Before starting the hunt for the alternative control algorithm(s), the system performance using the PI compensator was first examined. However, experiments show that optimal system performance depends on the gains of the PI compensation and the values of these gains are usually very difficult to locate. Generally, the rate of system response increases as the gains increase. But, the system will become unstable if the gains become too high. Besides the gains, the scan rate is also a critical parameter that determines the accuracy of the image acquired. Using too fast a scan rate, the feedback system might not have enough time to react to the input, but too slow a scan rate becomes impractical. Usually, an AFM operator chooses the gains and the scan rate using a "trial and error" method and by experience. Because of these difficulties, an automated AFM system was developed. This automated system requires no knowledge of control system for an AFM operator to acquire images using the AFM.

The investigation of the new AFM control algorithm(s) and the development of the

automated AFM system required altering the control algorithm of the AFM controller. This alteration of the control algorithm is made easier using a digital AFM controller in which the operational algorithms are software implemented. The following sections give an overview of the principle of an analog and digital AFM controllers. The digital controller built in our laboratory will be used as an example of implementing a digital controller.

1.3 The AFM Controller

The AFM controller is defined as a device that controls the operation of an AFM. Referring to the schematic diagram in figure 1.3, the controller includes everything in the figure except the microscope itself. When the AFM was first invented, an analog controller was used in which the controller was entirely built out of discrete electronic components. Although hardwired systems offer high speed operations, it requires hardware changes and rewiring for even a minor system modification. Recently, the advance of VLSI technology makes it possible to produce sophisticated digital signal processors (DSPs) at a reasonable price. DSPs open the gate for the AFM controller to enter the digital realm. Inside the digital controller, the control algorithm can be software coded and system modification can be done simply by rewriting code. This feature of the digital controller provides huge flexibility in system modification. The experiments carried out in this thesis are made possible using the digital controller since most of the experiments require altering the control algorithms of an operating AFM. The following two sections give a brief review of typical analog controller and digital controller.

1.3.1 Analog AFM Controller

An analog controller is mainly composed of discrete electronic components. A typical analog controller uses a triangle-wave generator to control the x and y direction movement of the sample. The frequency of the triangle wave determines the scan rate

along x direction. On the other hand, servo electronics are employed for the feedback system. In the very beginning, the feedback output is sent to a line plotter for plotting out the feedback output row by row. As the computer became sophisticated, analog-to-digital converter (ADC) was used to sample the feedback output so that images acquired can be displayed on the screen and stored on the disk for further image enhancement.

1.3.2 Digital AFM Controller

The inflexibility in modification of the analog controller system fueled the move to controlling in the digital realm. There are numerous ways of implementing the controller digitally. The idea behind a digital AFM controller is that some or all components inside the controller are implemented by software. For example, a partially digital controller can be implemented by using software counter to control the x and y direction movement of the piezo and leave the feedback system to servo electronics. Such system is exactly what happened before the digital signal processor became widely available [13]. An all-digital controller uses a digital control system to carry out the feedback mechanism inside the AFM system. Also, since digital signal processors support multiplication in a single instruction cycle, this feature allows real time image display while scanning. Various digital controllers have been designed and built by different laboratories [14 - 17] and commercial companies [18,19]. The following section uses the digital controller that was designed and built in the Scanning Tunneling Microscopy Laboratory of University of Manitoba as an example to demonstrate the principle of implementing a digital controller.

1.4 All Digital AFM Controller

The AFM used in our laboratory is from Park Instrument Inc. It uses tube scanner for positioning the sample and employs beam bounce technique to detect the deflection of the cantilever. The digital AFM controller was custom designed and built for the

specification of our AFM. Figure 1.4 shows the schematic diagram of the controller with the microscope. The system can be divided into four sections: the microscope itself, a DSP board, a host computer for the DSP board, and an analog/digital interface board. The following two sections provide an overview of the entire system. Interested readers can retrieve the detail from the thesis report "Design and Implementation of an All-Digital Atomic Force Microscope Controller" by Kerry Yackoboski [20] of University of Manitoba.

1.4.1 Hardware and Software Architecture

The digital controller was based on a digital signal processor (DSP56001) from Motorola. The DSP chip is mounted in an "evaluation board" with 64 kilobytes RAM and necessary interface circuitry (the term "DSP board" is used in the rest of the discussion to represent the DSP chip with the evaluation board). The DSP board is hosted by a 386 IBM-PC in which the PC directs the DSP to carry out particular AFM operations. The DSP board is an external board which has two communication links with the host PC. One of the links is for downloading and debugging the program on the DSP chip, and the other link is for data transmission.

The DSP chip and the host PC share the jobs of controlling the operation of the AFM. The DSP chip is responsible for all the calculations necessary in the system. The program running inside the DSP chip (called "DSP-program") is written in C/assembly language and is compiled and downloaded from the host PC [21]. The DSP-program is divided into many modules in which each module responsible for a single AFM operation such as accepting new scan parameters from the PC, sending bicell signal to the PC and controlling the stepper motor. The main subroutine of the DSP-program is called "feedback subroutine" which controls the feedback, updates the x and y movement of the piezo, and acquires images. These modules are written as interrupt subroutines which are

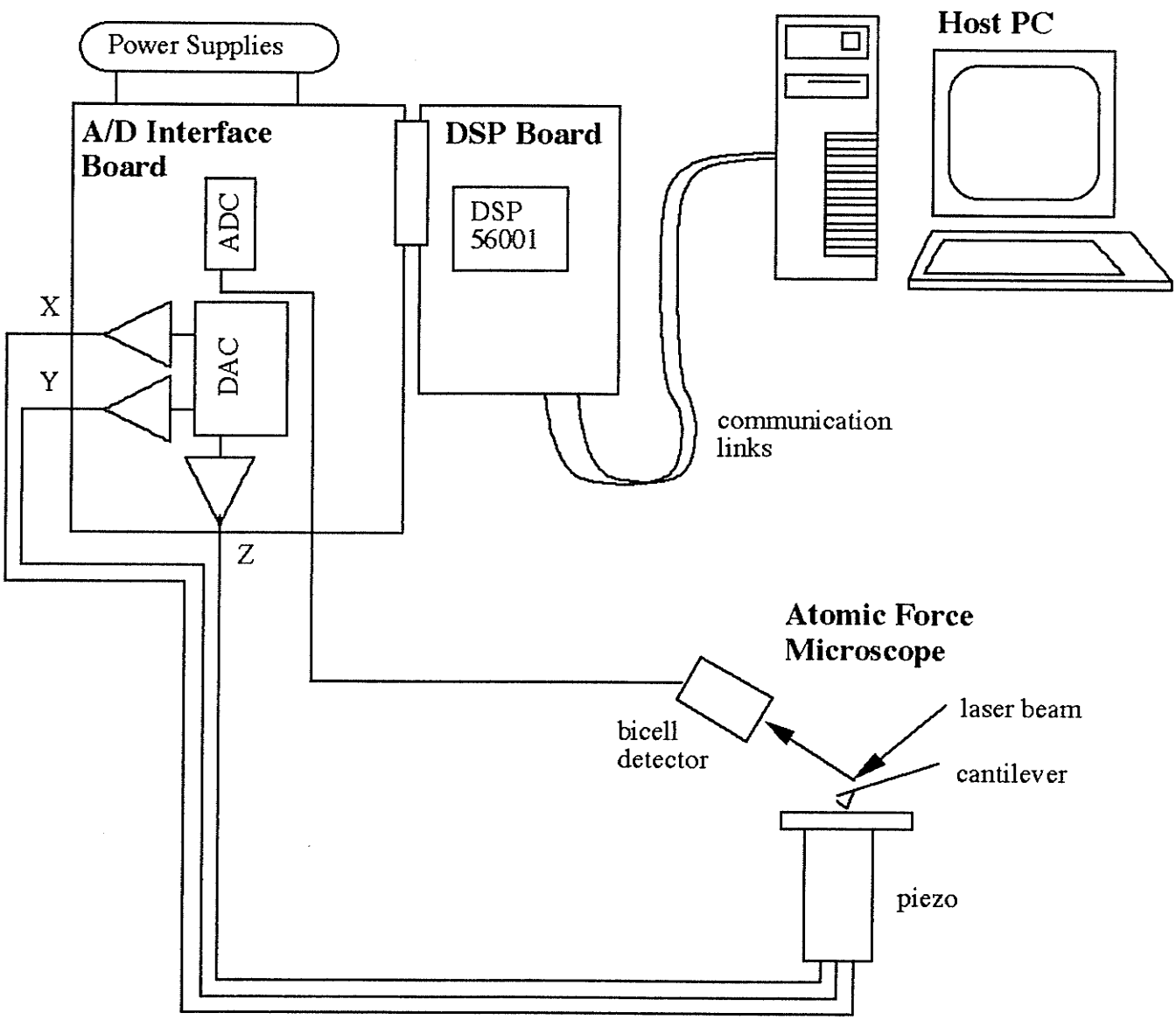

Figure 1.4 Physical hardware layout of the Digital AFM system

called for action by the host PC (the detail will be discussed in the following paragraphs).

The host-PC serves as a dual-purpose user graphical interface for displaying real time images during scanning and providing a menu driven control panel for the users to input control parameters. The program on the PC (called PC-program) is written in Turbo Pascal 6.0 by Borland International [22]. Because of the 640 kilobytes memory limitation of the MS-DOS, overlay programming technique is used in which only the portions of the program that are required at a given time are loaded into memory.

The DSP-program and the PC-program are running simultaneously and independently, and the data transmission between them is through hardware interrupts. After the DSP-program is executed, the DSP-program is running in an infinite "for" loop and waiting for the host PC to call the subroutines for action. For instance, if the scan size is changed by the user, the PC-program specifies the particular memory address of the interrupt subroutine for accepting scan size and puts the new scan size into the output register of the host PC. Then, the PC-program initiates an interrupt onto the DSP calling the particular subroutine to accept the new data, and then the host-PC sends the data via the Host Interface (HI) on the DSP. The subroutine retrieves the data from the input register of the DSP and assigns the data to the corresponding variables.

The PC-program starts the scanning by calling the feedback subroutine in the DSP-program and then runs into a "while" loop waiting for the DSP sending the image data to it. After a row of image data (fixed at 256 points) is collected, the DSP sets the address pointer to the start of the data array and sends out a signal to request a high-priority interrupt on the host PC. The interrupt causes the host PC to stop its current job and read in the data from the input register 256 times. As the PC-program is reading the data, the address pointer in the DSP-program automatically updates and new data is sent out for the host PC continuously. After receiving the entire row of data, the host PC displays the data

on the monitor screen.

The analog/digital interface board serves as a middle person between the controller and the microscope. The major components on the board are the analog-to-digital converters (ADCs), the digital-to-analog converters (DACs), and the high voltage op-amps. A 12 bit ADC with built-in sample and hold is used for digitizing the signal coming from the bicell detector. The input range and the conversion time of the ADC are +/- 5 volts and 6 μsec respectively. Original design does not include an anti-aliasing filter at the input of the ADC. Five DACs are used for controlling the piezo movement. Two pairs of 14 bit DACs with high voltage op-amps are employed for controlling the x and y direction movement of the piezo. The voltage range of the op-amps is 0 to 400 volts which gives the system view of field from 130 nm to 7000 nm. A single 16 bit DAC with high voltage op-amp range from -200 to 120 volts are used for controlling the Z-piezo. With the piezo sensitivity equal to 82 Å/V in the vertical direction, the resolution is 0.4 Å/bit.

1.4.2 Operational Algorithms

Two operational algorithms of the AFM system will be discussed in this section. One of them is the feedback control algorithm and the other is the scan algorithm.

A digital proportional-integral (PI) compensator is used inside the feedback loop of the system. The equation for calculating the feedback output is shown as follow.

$$Z_{out} = K_P * \text{error} + K_I * \sum_{n=0}^{N} \text{error} \qquad (1.1)$$

where error is the difference between the signal from the bicell detector and the set point value. The first and second terms on the right-hand-side of equation (1.1) are the proportional and the integral controllers of the PI compensator respectively. The proportional controller multiplies the current error signal with a constant called proportional gain, K_P, while the integral controller accumulates the error signal and multiplies the sum

with a constant called integral gain, K_I. Z_{out} is the sum of the output from the proportional and integral controllers and is sent to the piezoelectric via the analog/digital interface board. Equation (1.1) is software coded inside the feedback subroutine of the DSP-program.

The scan algorithm controls the way that the tip moves over the surface. First, the tip moves from left to right along the x direction over the surface. As the tip reaches the rightmost edge of the scan frame, it scans back along the same path. After the tip reaches the spot where it started, the tip steps downward or upward in the y direction depending on the direction the tip is scanning. When the tip is moving from left to right, the program carries out feedback and saves the feedback output as an image data of the surface topography. Since the image is composed of 256 by 256 pixels, 256 data points will be evenly collected along one row of scanning. However, since large step voltages excite the resonance of the piezo and causes the piezo ringing, instead of jumping from one image point to the other, feedback points are inserted between two image points. Figure 1.5 shows a simple diagram that the tip scans on the sample surface. As shown in figure 1.5, as the tip scans across the sample surface from point x1 to x4, the voltage supplied to the piezo is gradually stepping from v1 to v2, then v3 and v4 as shown in the graph in the middle of figure 1.5 (assuming there are two feedback points between two image points). If there are no feedback points between image points, the voltage will step from v1 to v4 directly as shown in the graph at the bottom of the figure. The difference between the image points and the feedback points is that the feedback output at the feedback points is not recorded. The number of the feedback points between two image points is related to the scan size. In the original design, a look-up table is used to tell the DSP-program the number of feedback points that should be used for a particular scan size. In addition, the scan speed determines the speed of the DSP-program where the PC-program calculates the interrupt frequency between the PC and the DSP chip to agree with the scan speed specified. The detail of the scan algorithm will be discussed in Chapter 2.

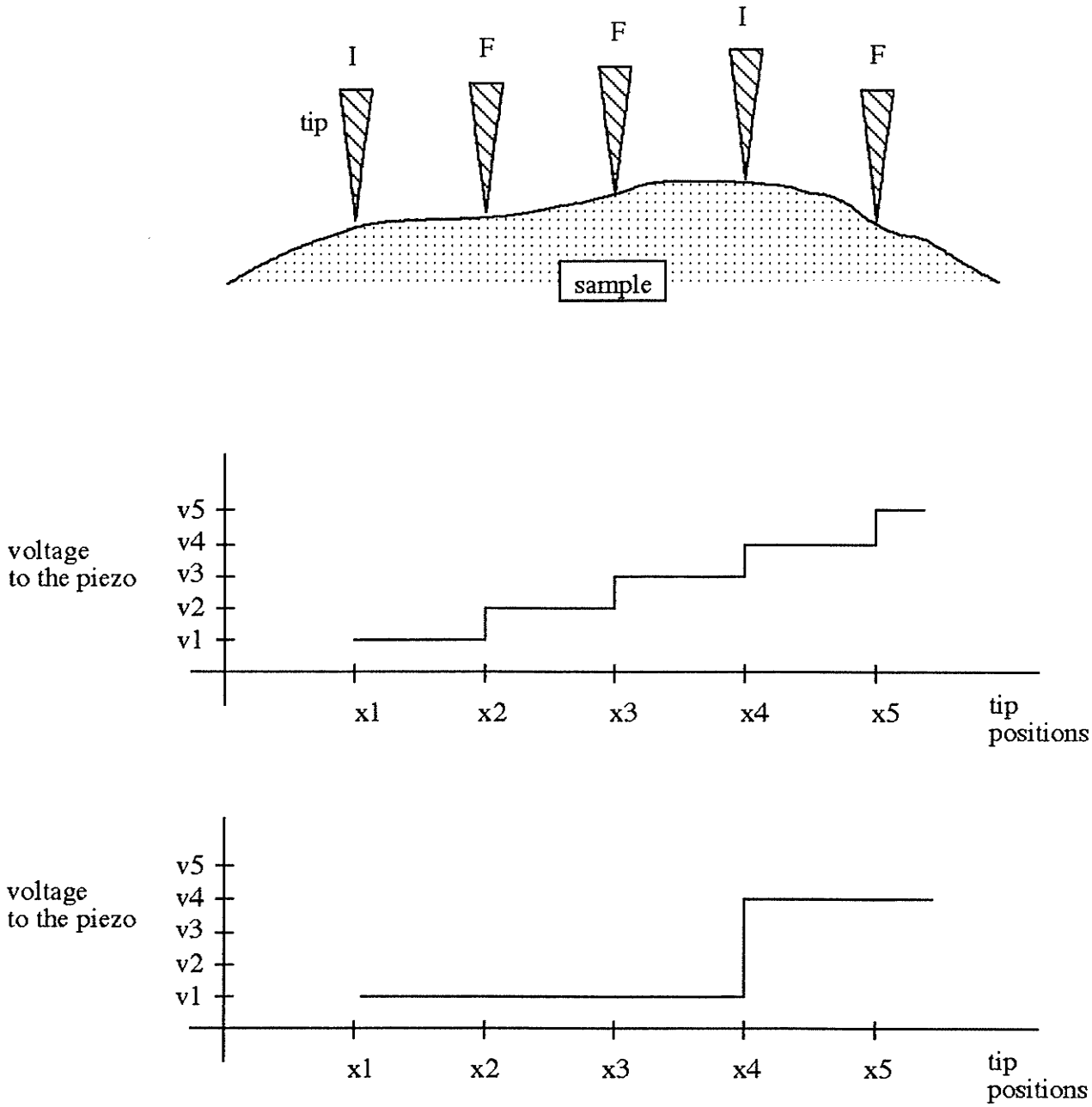

Figure 1.5 The picture on the top shows the tip scanning across the sample surface from point x1 to x4. Assuming there are two feedback points between two images points, the voltage supplies to the piezo is gradually stepping from v1 to v2, then v3 and v4 as shown in the graph in the middle. If there are no feedback point, the voltage will step from v1 to v4 as shown in the graph in the bottom.

1.5 Outline of this Thesis

After a brief introduction of the AFM and the all digital AFM controller given in this chapter, the next chapter, Chapter 2, discusses the various system enhancements of the all digital AFM controller. The enhancement of the digital controller includes the modification of the scanning algorithms, the hardware electronics, the functional options and the graphical user interface. Chapter 3 of this thesis analyses the digital control system of a PI compensated AFM system. A system model is developed to investigate the behavior of the system. A commercial control system design software called 'cc' is used to carry out all the simulations. From the control system analysis in the Chapter 3, the feedback gains of the PI controller are found to have a major effect to the AFM system performance and the optimal values of these feedback gains are difficult to achieve. In addition, the scan rate also plays a major role to the accuracy of the images acquired. Therefore, an automated scan algorithm is developed and is discussed in the Chapter 4. The automated scan algorithm is a two step algorithm. First, the system uses genetic algorithms to search for the optimal feedback gains. Then, using the optimal gains obtained, the system proceeds the scanning with an adaptive scanning algorithm. The adaptive scan algorithm keeps adjusting the sample height until the error signal is less than a constant called 'error-threshold'. Then, the images acquired would have a known finite error margin at every pixel. Finally, Chapter 5 carries out the feedback control design for the AFM system using the frequency response technique and the root-locus technique. In addition, two alternative scan algorithms are introduced which may improve the system performance of the AFM. The conclusions and recommendations are included in Chapter 6, the last chapter of this thesis.

Reference

[1] G. Binnig, C. F. Quate and Ch. Gerber "Atomic force microscope," *Phys. Rev. Lett.* **56**, 930 (1986).

[2] D. Sarid and V. Elings "Review of scanning force microscopy," *J. Vac. Sci. Technol.* **B 9**, 431 (1991).

[3] N. A. Burnham and R. J. Colton "Measuring the nanomechanical properties and surface forces of materials using an atomic force microscope." *J. Vac. Sci. Technol.* **A 7**, 2906 (1989).

[4] P. K. Hansma, V. B. Elings, O. Marti and C. E. Bracker "Scanning tunneling microscopy and atomic force microscopy : application to biology and technology," *Science* **242**, 209 (1988).

[5] D. Rugar, H. J. Mamin, R. Erlandsen, J. E. Stern and B. D. Terris "Force microscope using a fiber-optic displacement sensor," *Rev. Sci Instrum.* **59**, 2337 (1988).

[6] H. J. Mamin, D. Rugar, J. E. Stern, B. D. Terris and S. E. Lambert "Force microscopy of magnetization patterns in longitudinal recording media," *Appl. Phys. Lett.* **53**, 1563 (1988).

[7] J. E. Stern, B. D. Terris, H.J. Mamin and D. Rugar "Position and imaging of localized charge on insulator surfaces using force microscope," *Appl. Phys. Lett.* **53**, 2717 (1988).

[8] G. Binnig and D.P.E. Smith "Single-tube three-dimensional scanner for scanning tunneling microscopy," *Rev, Sci. Instrum.* **57**, 1688 (1986).

[9] T. R. Albrecht, S. Akamine, T. E. Carter and C. F. Quate "Microfabrication of cantilever styli for the atomic force microscope," *J. Vac. Sci. Technol.* **A 8**, 3386 (1990).

[10] S. Alexander, L. Hellemans, O. Marti, J. Schneir, V. Elings and P. K. Hansma "An atomic-resolution atomic-force microscope implemented using an optical lever," *J. Appl. Phys.* **65** (1), 164 (1989).

[11] S. Fujisawa, M. Ohta, T. Konishi, Y. Sugawara and S. Morita "Difference between the forces measured by an optical lever deflection and by an optical interferometer in an atomic force microscope," *Rev, Sci. Instrum.* **65** (3), 644 (1994).

[12] K. Takata "Whole electronic cantilever control for atomic force microscopy," *Rev, Sci. Instrum.* **64** (9), 2598 (1993).

[13] O. Marti, S. Gould and P. K. Hansma, "Control electronics for atomic force microscopy," *Rev, Sci. Instrum.* **59**, 836 (1988).

[14] S. M. Clark, D. R. Baselt, C. F. Spence, M. G. Youngquist and J. D. Baldeschwieler, "Hardware for digitally controlled scanned probe microscopes," *Rev, Sci. Instrum.* **63**, 4296 (1992).

[15] D. R. Baselt, S. M. Clark, M. G. Youngquist , C. F. Spence and J. D. Baldeschwieler, "Digital signal processor control of scanned probe microscopes," *Rev, Sci. Instrum.* **64**, 1874 (1993).

[16] T. L. Porter, " Scanning tunneling microscope data acquisition and control in visual basic," *Rev, Sci. Instrum.* **64**, 3530 (1993).

[17] P. Heuell, M. A. Kulakov, and B. Bullemer, "An adaptive scan generator for a scanning tunneling microscope," *Rev, Sci. Instrum.* **65**, 89 (1994).

[18] Digital Instruments, 520 E. Montecito Street, Santa Barbara, CA 93103.

[19] TopoMetrix, 1505 Wyatt Drive, Santa Clara, CA 95054.

[20] K. Yackoboski, "Design and Implementation of an All-Digital Atomic Force Microscope Controller," *MSc. Thesis*, University of Manitoba, 1992.

[21] Technical Staff, *DSP56000ADS Application Development System Reference Manual Version 2.00,* Motorola Inc., Austin, Texas, 1989.

[22] *Turbo Pascal Version 6.0, Turbo Assembler Version 2.0,* and *Turbo Debugger Version 2.0,* Borland International. Scotts Valley, California, 1990.

CHAPTER 2

SYSTEM ENHANCEMENT

This thesis started as a continuation project of implementing the all-digital AFM controller mentioned in chapter one. When this thesis began, the basic implementation of the controller was complete, in which the controller was able to drive the AFM for acquiring the sample topography and display the image on the monitor of the host PC. However, the ambitions of this work are to go beyond proving the feasibility of building a digital AFM controller and optimize the controller performance as well. This chapter describes various modifications that correspond to the different areas of system enhancement; they include scan algorithm modification, hardware enhancement, functional enhancement, and graphical user interface enhancement.

2.1 Scan Algorithm Modification

The first major system modification is on the scan algorithm. As mentioned in section 1.4, 256 image points are collected along each scan row in the x direction and feedback points are inserted between adjacent image points. The difference between image points and feedback points is that the feedback output at the feedback points is not recorded. The function of the feedback points is to prevent the piezo from ringing when the tip moves from one image point to the other by applying a step voltage to the piezo.

2.1.1 Original Scan Algorithm

In the original system, the number of feedback points between two image points

depends on the scan size. A look-up table is employed to find out the feedback points that should be used with particular scan size. However, before looking at the look-up table in Table 1, a description on how to digitize the scan size may be helpful in understanding the relationship between the scan size and the feedback points.

First of all, a 14 bit digital-to-analog converter (DAC) is used for controlling the x direction movement of the piezo. Since the 256 image points are evenly distributed along each scan row for the entire image, the minimum and maximum scan range are 256 bits and 64 * 256 bits (which equals 16384 or 2^{14} bit) respectively. For instance, if the maximum scan range is employed, there are 63 bits between two image points, or in other words, there can be, at most, 63 feedback points. However, the total number of feedback points along a scan row is kept under a reasonable number, which is 3000 in the original design and is arbitrary chosen, to keep the feedback rate high. Table 1 shows the look-up table used in the original system. Instead of using the total number of bits of the scan size along a scan row, the multiplier of 256 is used in Table 1 to simplify the explanation.

Table 1 Look-up table for the number of feedback points that should be used for a particular scan size.

scan size	number of bits between feedback points
0..11, 13, 17, 19, 23, 29, 31, 37, 41, 43, 47, 53, 59, 61	0
14, 22, 26, 34, 38, 46, 58, 62	1
12, 15, 18, 21, 24, 27, 30, 33, 39, 48, 51, 54, 57	2
16, 20, 28, 32, 36, 40, 44, 52, 56, 60	3
25, 35, 45, 50, 55	4
42, 49, 63	5

Also, the distance between two feedback points is used to represent the number of feedback points employed. For example, zero bit between two feedback points means all the bits between two images points are used as feedback points.

The scan sizes from 11 to 61 in the first row of Table 1 are prime numbers. Feedback points cannot be evenly distributed between two image points, except using all the bits between the image points as feedback points. A problem may arise for large scan sizes since the sum of total number of feedback points and image points will be very large which will slow the feedback rate.

The scan rate, which is the speed of the tip moves along x direction, is a significant parameter of scanning. The moving of the tip along the x direction is included in the feedback subroutine. The DSP moves the tip to the current point (image point /feedback point) and then acquires the error signal and sends out the feedback output to the Z-piezo. Then the DSP-program waits for the next interrupt call to run the feedback subroutine again. The PC-program calculates the timer interrupt, the time between interrupts, of the DSP needed to yield the scan rate specified. In other words, the PC-program calculates the frequency of interrupts needed from the total number of feedback points and image points in order to agree with the scan rate specified. Equation (2.1) expresses the mathematical representation to calculate the interrupt frequency.

$$\text{interrupt frequency} = (\text{scan rate specified}) * (\text{number of columns per row}) * (2\ \text{rows/cycle}) * (\text{number of raster points} / \text{number of image points}) \quad (2.1)$$

After reading the scan size and scan rate specified, procedure *ComputeEverything* in PC-program carries out this calculation before proceeding with scanning.

2.1.2 Current Scan Algorithm

The original scan algorithm requires a lot of calculation and approximation in determining the number of feedback points and the interrupt frequency needed. A novel scan algorithm is developed which is similar to the triangular wave generator that is used in the analog controller. The new scan algorithm uses the fastest interrupt frequency of the DSP chip and the PC-program calculates the number of feedback subroutines required to agree with the scan rate specified.

First, the time required for running the feedback subroutine once, called feedback time, is examined, which is about 30 μsec in our digital controller. Then, the PC-program uses equation (2.2) to determine the number of feedback subroutines needed across a scan row.

$$\text{number of feedback subroutine} = \frac{1}{(\text{feedback time}) * (\text{scan rate specified}) * 2} \qquad (2.2)$$

The result from the RHS of equation (2.2) is rounded up to nearest integer. Before sending the number of feedback subroutines to the DSP-program, the PC-program divides it by 256 in which the result is the number of feedback subroutines that will be carried out between two image points.

During scanning, the feedback subroutine in the DSP-program does two tasks simultaneously, which are updating the tip position along the x direction and keeping track of the feedback points and image points. The feedback subroutine uses a counter to step the tip along x direction during scanning. The step counter is incremented (or decremented, depending on the direction the tip is moving) by a factor called *xfrac*. The value of xfrac is calculated by equation (2.3) where *xrange* is the scan size and *feedvalue* is the number of

feedback subroutines received from the PC-program.

$$\text{xfrac} = \frac{\text{xrange}}{64 * 256 * \text{feedvalue}} \tag{2.3}$$

Every time when the feedback subroutine is called, the position of the tip along x direction is updated using equation (2.4). The variable *xout* and *xoffset* are the new tip position and the offset of the current position respectively. The product of 'xinc * xfrac * 64' will be rounded up to the nearest integer. Because of the rounding of the product 'xinc * xfrac * 64', the resulting xout may have the same value for several consecutive steps in which the tip will stay at the same spot. The variable *xinc* will be incremented or decremented by one when the subroutine is called.

$$\text{xout} = \text{xoffset} + \text{xinc} * \text{xfrac} * 64 \tag{2.4}$$

The implementation of the scan algorithm is best illustrated by an example as follows. For example, if the xrange is 16*256 bits and the feedvalue is 40, then, using equation (2.3), the value of xfrac is about 0.00625. Using equation (2.4) with xoffset set to zero for simplicity, Table 2 shows the movement of the tip.

Table 2 shows that during the second and third cycle, the value of xout is the same which means that the tip stays at the same spot and the program will adjust the sample height one more time.

On the other hand, the feedback subroutine uses a counter called *feedcounter* to keep track of the feedback and image points. If the feedcounter is less than the feedvalue, then the tip is on a feedback point and the program increments the feedcounter by one. Otherwise, if the feedcounter equals feedvalue, then the tip is on an image point and the feedback output is saved into the data array and the feedcounter is reset to zero.

Table 2. The movement of the tip along x direction with xfrac = 0.00625

xinc	xout
1	0.4 -> 0
2	0.8 -> 1
3	1.2 -> 1
4	1.6 -> 2
.	.
.	.
.	.
39	15.6 -> 16
40	16.0 -> 16

2.2 Hardware Enhancement

The DSP board and the analog/digital interface board were placed side by side into a metal case. Seven different power circuits were purchased and added into the metal case for voltage supplies to the ADC, DACs, high voltage op-amp, the stepper motor, the bicell detector, and the laser diode inside the AFM head. Only two minor hardware modifications have been carried out on the system and will be discussed as follow

2.2.1 Anti-aliasing Filter

When a signal is sampled, any frequency component higher than the Nyquist frequency, which is half the sampling frequency, will be folded into the low frequency region. This phenomenon is called *aliasing* [1]. Since a signal that suffers from aliasing

during sampling is not able to be accurately reconstructed, the input signal must be strictly bandlimited to a bandwidth of less than the Nyquist frequency before sampling. A simple lowpass filter with cutoff frequency at the Nyquist frequency can suppress the aliasing components and this lowpass filter is called anti-aliasing filter.

Although the ADC in our digital controller has 100 kHz sampling rate, the actual sampling rate is the frequency of the feedback subroutine that acquires the signal from the bicell detector, which is at about 33 kHz. Therefore, an anti-aliasing filter with cut-off frequency at 16 kHz is employed which is a simple first order RC lowpass filter composed of a 100 Ω resistor and a 0.1 μF capacitor.

2.2.2 LowPass Filter at the Input of the Piezo

Since the piezo has a high Q at resonance (the characteristic of the piezo will be discussed in the next chapter), adding a resistor at the input of the Z-piezo can reduce this resonant peak. Since the piezo in our AFM has capacitance about 10 nF, the combination of the resistor and the piezo was primary modeled as a lowpass filter. A better analysis of the effect of this extra resistor to the piezo will be discussed in the Appendix A. Since the interested frequency range of the AFM system is from 0 to 16 kHz, the cutoff frequency of the lowpass filter is chosen at 8 kHz which gives a 7 dB suppression of gain at 16 kHz.

2.3 Functional Enhancement

2.3.1 Incrementing / Decrementing the Feedback Gains

In the original design, the feedback gains are incremented or decremented by one each time when the right mouse button is clicked. A new function has been added so that the feedback gains can be increased or decreased by twenty percent. The feedback gain button on the screen is divided into two halves, the upper portion of the button is for

increasing the gains while the lower portion is for decreasing the gains. When the cursor is on the upper portion of the button, clicking the right mouse button will increase the gain by twenty percent while clicking the left mouse button will increase the gain by one. On the other hand, when the cursor is on the lower portion of the button, clicking the right and left mouse button will decrease the gain by twenty percent and by one respectively.

2.3.2 Plane Subtraction Function Added

A flat plane subtraction feature is added as one of the options among the option buttons on the screen. This feature subtracts a plane from the image that is caused by scanning an unleveled sample. The option button can be activated only after the original image is loaded and displayed on the screen. After the button is activated, the original image is erased and the plane subtracted image will be displayed.

2.4 Graphical User Interface Enhancement

2.4.1 Error Signal Display

The error signal is the difference between the signal coming from the bicell detector and the set-point value. This signal reflects the difference between the desired output and the actual output that represents the real surface and the feedback output to the Z-piezo respectively. Besides demonstrating the tracking ability of the system, the error signal also reflects the stability of the system. Using the proportional-integral (PI) compensator, the system has faster response as the gain of the feedback system increases. However, the system will become unstable if the gains are too high (details of the digital control system will be discussed in the next chapter). The AFM operator can keep track of the system stability by monitoring the error signal while adjusting the gains.

In the original design, an integer array with size 256 is declared in the DSP-

program to store the image data. This integer array is extended to 512 in size to store another 256 data for the error signal. During scanning, when an image point is encountered, the error signal is captured and stored into the array after the feedback output is sent to the Z-piezo. Therefore, the image data and the error data are stored alternatively in the data array. After finishing one row of scanning, the DSP-program sends an interrupt request to the host PC and asks for receiving the data as usual.

On the host-PC side, the PC-program reads in 512 data points and stores the image data and the error data into two different arrays. A new scope-like window is inserted onto the user screen to display the error signal. During scanning, the gray-level topography, the cross-section, and the error of the current scan row will be displayed simultaneously on the screen.

2.4.2 Mouse Droppings Problem

When the cursor is overlapped on an area, such as a button, where the area is supposed to change color, the portions that the mouse is sitting on do not change to the color specified but change to another color instead. The original PC-program uses Pascal built-in function *PutImage* to erase the previous mouse and draw the current mouse during updating mouse position. However, when the area is redrawn with color changes, the mouse loses track of what color should be redrawn. To solve this problem, the previous mouse is erased before the area starts being redrawn and changing color. After the area is completely redrawn, the current mouse is put back on the screen again.

2.4.3 Size Bar on the Image

A size bar with one quarter of the image width is displayed with the image on the screen when the image is reloaded. The corresponding length of the size bar is shown underneath of the bar in nanometer scale.

Reference

[1] L. B. Jackson, *Digital Filters and Signal Processing*, Kluwer Academic Publishers, MA. pp. 69-72, 1986.

CHAPTER 3

DIGITAL PI COMPENSATED AFM CONTROL SYSTEM ANALYSIS

The AFM uses the force between a sharp tip and the sample surface to map out the surface topography of the sample. By pressing the tip onto the sample surface, the tip will follow the surface contour as it scans over the surface. The vertical displacement of the tip can be captured by the bicell detector using the beam bounce technique as in our AFM system. However, because of the nonlinearity of the bicell detector, the output signal from the bicell detector is not suitable to be used as the image data of the surface topography. Instead, a closed loop feedback control system is employed which manipulates the detector signal and feeds back the signal to the Z-piezo to adjust the sample height. The function of the feedback system is to keep the tip stationary in the vertical position (or in other words, keep the detector signal constant) by adjusting the sample height accordingly. If the tip can be held motionless during scanning, then the feedback output is following the sample contour and can be used as the image data to compose the surface topography.

The control algorithm employed inside the feedback control system has a significant effect on the performance of the AFM since the feedback output adjusts the sample height and composes the image of the sample surface topography. Our system uses proportional-integral (PI) control algorithm to manipulate the error signal and feed the output back to the piezo. This chapter attempts to investigate the behavior of a PI compensated AFM system. Among various components inside the AFM system, the piezo tube scanner and the PI compensator dominate the behavior of the system. The first section of this chapter reviews the characteristics of these two major components of the system. Then, the overall transfer function of the system is developed from modeling the physical hardware. Afterward, the

system behavior is reviewed using the transfer function developed. All the simulated results in this chapter are done by a commercial control analysis software package called 'cc' [1].

3.1 Characteristics of the Piezo Tube Scanner and the PI Compensator

The response of the feedback loop inside the AFM system is primarily determined by the dynamic behavior of the piezo tube scanner and the PI compensator [2]. At resonance, the phase and gain of the piezo may drive the control system into net positive feedback and the system becomes unstable [3]. The functions of the PI compensator are to eliminate the unsatisfactory effects of the piezo tube scanner and improve the system performance such as the rate of the system response and stability. This section reviews the frequency and transient response of the system with the piezo alone and the effect of the PI compensator to the system. The interest of this section is not on the behavior of the complete physical AFM control system, but on understanding the relationship between the piezo and the PI compensator in the system.

3.1.1 Characteristic of the Piezo

According to the analysis of D. Pohl [4], only the lowest resonant frequencies of the piezo are significant in the microscope application. Assuming linear behavior, the piezo tube scanner is modeled as a damped harmonic oscillator with the transfer function shown as follows

$$G'(s) = K_{dc}\, G(s)$$

$$\text{where } G(s) = \frac{\omega_n^2}{s^2 + \frac{\omega_n}{Q}s + \omega_n^2} \tag{3.1}$$

In equation (3.1), K_{dc} is the DC gain of the piezo while the ω_n and Q are the resonant frequency and the quality factor at resonance respectively. Thus, G(s) is the normalized function of G'(s). The value of DC gain, K_{dc}, is 82 Å/V as provided by the manufacturer. The resonant frequency, ω_n, ranges from 16 kHz to 30 kHz depending on the mass of the sample put on top of the piezo. The heavier the sample placed onto the piezo, the lower resonant frequency the piezo will have. In the following analysis, 16 kHz will be used as the resonant frequency of the piezo and the quality factor equals 20 as measured. Both the measurements of the resonant frequency and the quality factor are taken by K. Westra [5].

Figure 3.1 shows the block diagram of a simplified closed-loop AFM control system with the piezo alone. The input and output of this block diagram are the desired and actual sample height respectively. This block diagram neglects the details of the system such as the bicell detector and the electronic components required. For simplicity, normalized transfer function, G(s), is used inside the block diagram with the loop gain of the system, K. The block diagram attempts to provide a general overview of the transient response of the closed-loop AFM system with the effect of the piezo tube scanner alone.

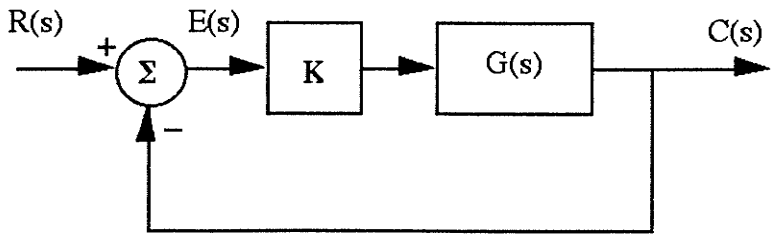

Figure 1. Block diagram of the closed loop AFM system with the piezo tube scanner alone

Since G(s) has no pole at the origin of the complex s-plane, the system always has finite steady state error, e_{ss} [6]. The value of the steady state error can be obtained from the final value theorem of Laplace transform analysis shown as follows

$$e_{ss} = \lim_{s \to 0} sE(s)$$

$$= \lim_{s \to 0} s \frac{R(s)}{1 + KG(s)} \qquad (3.2)$$

where E(s) and R(s) are the Laplace transform of the error signal and the input signal respectively. If unity step input is used, then the steady state error can be calculated by replacing R(s) with $\frac{1}{s}$ shown as follow

$$e_{ss} = \lim_{s \to 0} \frac{1}{1 + KG(s)}$$

$$= \frac{1}{1 + K} \qquad (3.3)$$

As shown from equation (3.3), the steady state error for a unity step input can be reduced by increasing the system loop gain, K. However, the transient response of the closed loop system shown in figure 3.2 shows that the system is highly underdamped and the ringing at the output settles after 2 msec. Increasing the system loop gain will reduce the steady state error as well as increase the magnitude of the ringing.

In order to reduce the steady state error and the ringing at the output of the system, a control compensator is inserted into the feedback loop of the AFM system. The proportional-integral-derivative (PID) compensator is a common control compensator that is used in many control system applications and almost all existing AFM systems use a PID compensator to implement the feedback loop of the system. Since the derivative operator amplifies the high frequency noise, the derivative operator is omitted and only the PI compensator is employed in our system.

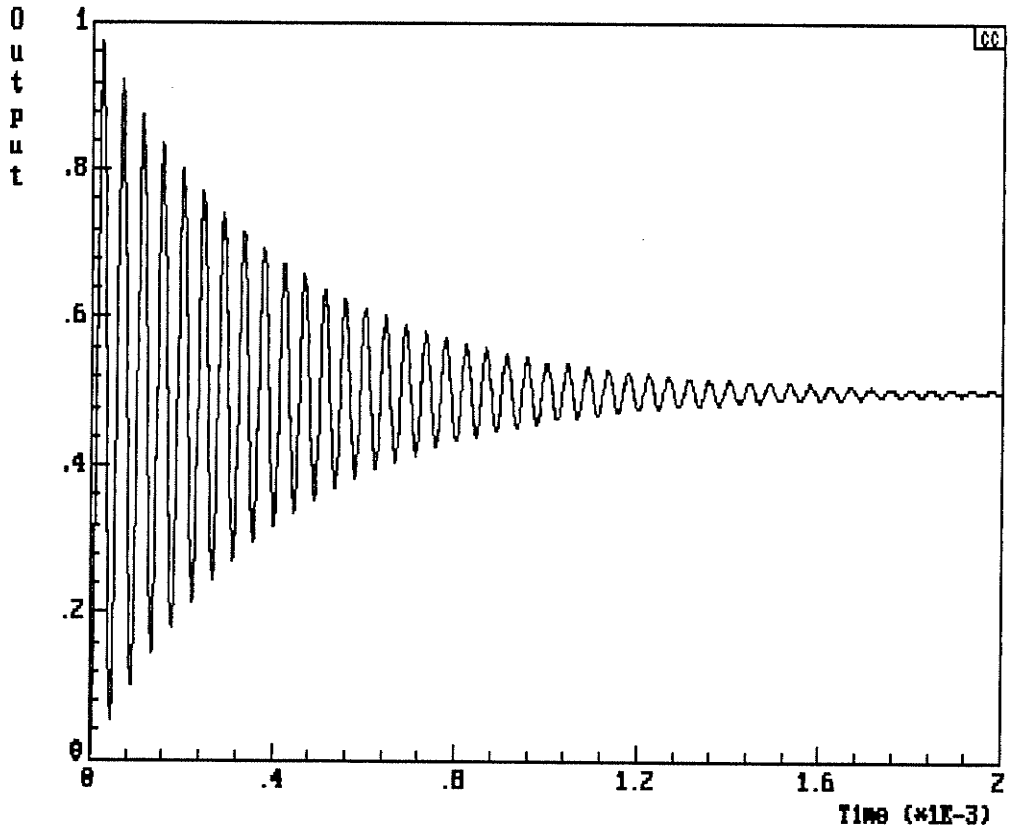

Figure 3.2 Transient step response of the closed loop AFM system with the piezo tube scanner alone.

3.1.2 Proportional-Integral (PI) Compensator

Proportional-Integral (PI) compensator is a combination of a proportional controller and an integrator. These two controllers are placed in parallel as shown in figure 3.3.

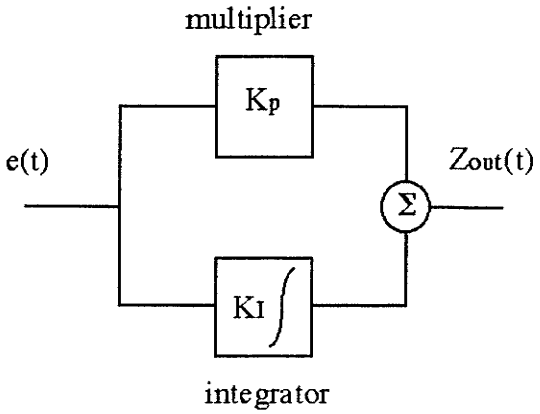

Figure 3.3 Simple block diagram of a PI compensator

The proportional controller is simply a multiplier that multiplies the error signal e(t) by a constant called proportional gain, K_p, and sends the product to the process plant, which is the piezo in this case. In other words, the proportional controller varies the loop gain of the feedback system by manipulating the error signal. On the other hand, the integrator keeps adding the error signal and sending the sum to the piezo continuously. Since the integrating effect stops only when the error signal is zero, the integrator is able to achieve the zero steady state error for a step function input. Usually, the output of the integrator is multiplied by a constant called integral gain, K_I, to increase the rate of integrating effect. Using Laplace transformation, the transfer function of the PI compensator, D(s), is shown as follows

$$D(s) = K_p + \frac{K_I}{s}$$

$$= K_p\left(\frac{s+\frac{K_I}{K_p}}{s}\right) \tag{3.4}$$

3.1.3 Behavior of the Compensated Piezo

Cascading the PI compensator with the piezo, the open loop transfer function of the system is

$$D(s)\,G'(s) = K_pK_{dc}\left(\frac{\left(s+\frac{K_I}{K_p}\right)\omega_n^2}{s\left(s^2+\frac{\omega_n}{Q}s+\omega_n^2\right)}\right) \tag{3.5}$$

The open loop transfer function of the system has a pole at the origin of the complex s-plane, and therefore, the system has zero steady state error for step input. From equation (3.5), the zero of the open loop transfer function depends on the ratio of the integral gain and the proportional gain. In order to simply the explanation, the proportional controller is omitted in the following discussion. Figure 3.4 shows the Bode diagram of the frequency response of the open loop AFM system that is compensated by the integrator only. The system has high gain at low frequency so that the system output has zero steady state error for a step input. However, the integrator reduces the bandwidth of the system which, in turn, reduces the speed of the system response. Although the system bandwidth can be increased by increasing the open loop gain of the system, the resonant peak of the system must be kept below 0 dB or the system will become unstable.

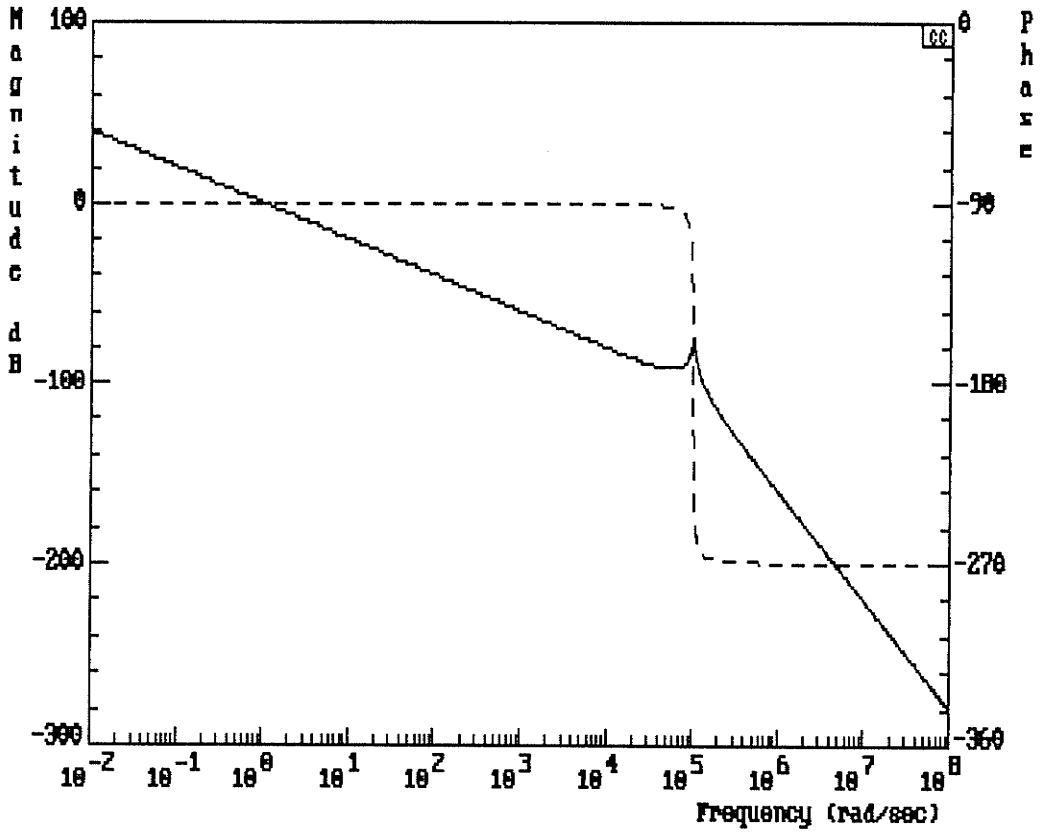

Figure 3.4 Frequency response of the open loop PI compensated AFM system.

3.2 Modeling of the Physical AFM Control System

In order to use numerical and graphical analysis to describe the behavior of the AFM control system, the physical hardware of the system is modeled into a mathematical expression. A block diagram is used to provide a pictorial representation of the system and each block represents a separate functional component inside the system. Figure 3.5 (a) and (b) show the physical hardware and the corresponding block diagram of the AFM control system respectively. The system input corresponds to the height of the features on the sample surface while the system output is the output of the compensator. The compensator output is used to adjust the sample height via the piezo and is used as the image data for the surface topography of the sample. Unlike the block diagram shown in the previous section, the piezo is placed on the feedback path of the system. The following sections discuss the functional component of each block in the block diagram.

3.2.1 Components on the Feedback Path

The compensator manipulates the error signal and feeds back the manipulated signal to the piezo via the feedback path of the system. As shown in the block diagram in the figure 3.5 (b), the feedback path converts the calculated value (in bits) from the compensator to the actual sample height (in angstroms) at the piezo. The functional components on the feedback path include a digital-to-analog converter (DAC), a high voltage op-amp, a lowpass filter and the piezo.

The compensator is software implemented inside the DSP-program and the output of the compensator is converted to an analog signal via a DAC before it is sent to the piezo. A 16-bit DAC with -5 to +5 voltage supplies is used in our system. The dc conversion factor of the DAC is

$$K_{DAC} = \frac{10}{2^{16}} \left(\frac{volt}{bit}\right) \qquad (3.6)$$

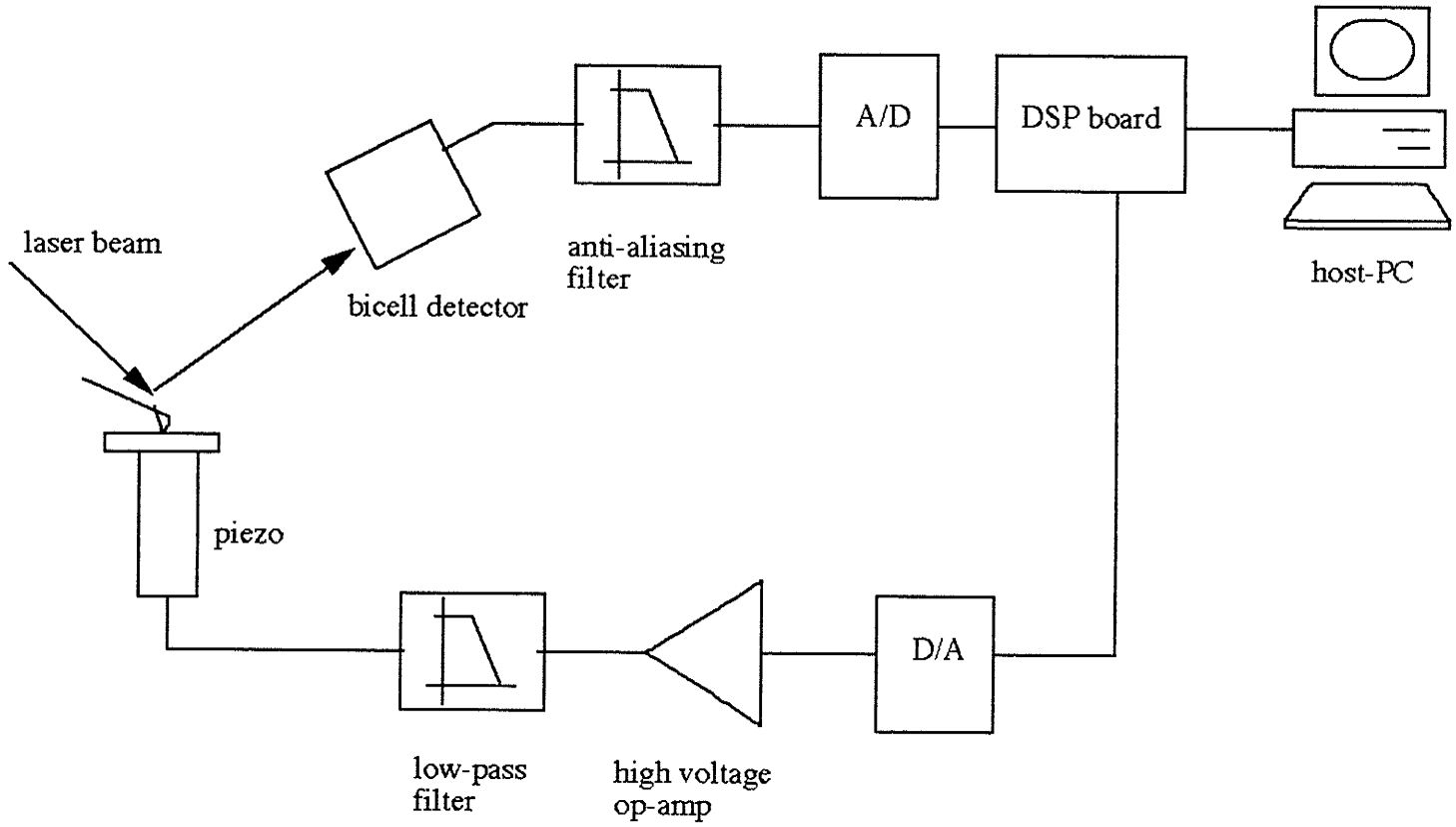

Figure 3.5a Physical hardware of the AFM control system

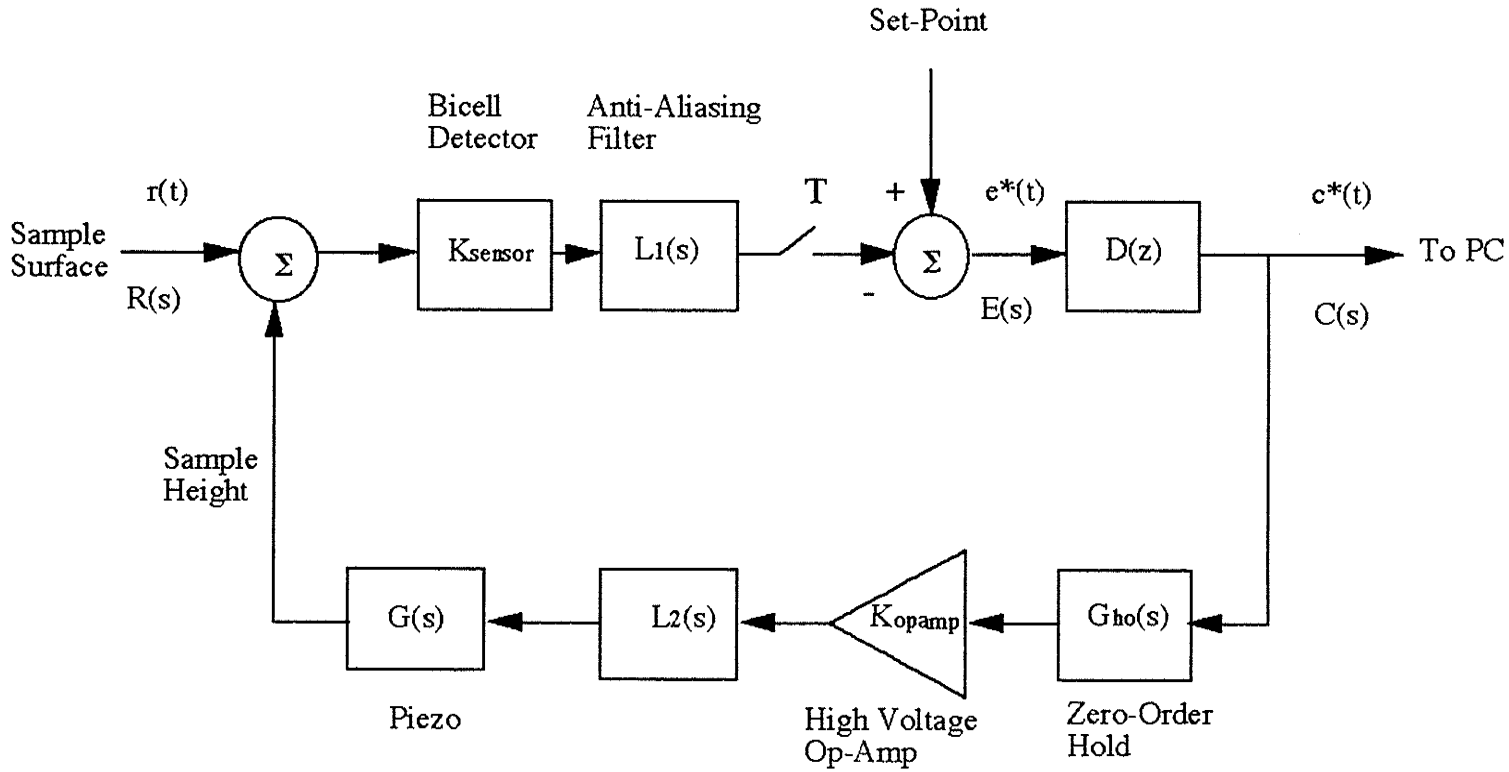

Figure 3.5b Block diagram of the AFM control system

The DAC is modeled as a zero-order (ZOH) in our system. The ZOH holds the output at a constant value during the sampling interval and the value equals to that at the preceding sampling instant. The transfer function of the ZOH in the s-domain is

$$G_{ho}(s) = \frac{1 - e^{-Ts}}{s} \tag{3.7}$$

The output of the DAC is amplified to the working range of the piezo via a high voltage op-amp. The voltage supplies of the op-amp are -200 and +120 volt. Since the output of the DAC ranges from -5 to +5 volt, therefore the gain of the op-amp, K_{amp}, is

$$K_{amp} = \frac{120 - (-200)}{5 - (-5)} = 32$$

The output of the high voltage op-amp is then sent to the piezo.

The characteristic of the piezo tube scanner has been described in the previous section. A resistor is added at the input of the piezo in order to reduce the resonant peak of the piezo. Since the piezo has 10 nF capacitance, the combination of the resistor and the piezo was primary modeled as a RC lowpass filter. A better model which describes the effect of the resistor to the piezo will be described in Appendix A. The cut off frequency of the lowpass filter is chosen as 8 kHz which gives a 7 dB suppression of gain at the resonance of the piezo. A 2 kΩ resistor is placed at the input of the Z-piezo and the transfer function of the filter is

$$L_2(s) = \frac{\omega_c}{s + \omega_c} \tag{3.8}$$

where ω_c is the cut off frequency of the filter and is 50,000 rad/sec.

In all, the transfer functions of the functional blocks on the feedback path are

combined together and named $H'(s)$. The transfer function of $H'(s)$ and its normalized function $H(s)$ are shown in equation (3.9) as follow

$$H'(s) = K_{amp}\ K_{DAC}\ K_{dc}\ G_{ho}(s)\,H(s)$$

$$\text{where } H(s) = L_2(s)\ G(s) \qquad (3.9)$$

3.2.2 Bicell Detector and Anti-Aliasing Filter on the Feedforward Path

On the feedforward path, the input is the desired sample height, such that the force between the tip and the sample remains constant. The difference between the desired and actual sample height is captured by the bicell detector as described in Chapter 1. The output signal of the detector is sampled by the ADC and the sampled value is subtracted from the set-point inside the DSP-program. The result from the subtraction is named error signal and is used by the compensator to calculate the feedback output. Since the set-point is a constant and remains unchanged during scanning, it is omitted in the following calculation. The feedforward path converts sample height difference (in angstroms) to the feedback signal (in bits) in which the feedback signal is sent to the piezo via the feedback path and to the host PC for image display.

The gain of the bicell detector is measured by supplying a 5 volt peak to peak sinusoidal function to the piezo and observed the signal changes from the detector. The signal at the output of the bicell detector is a sinusoidal waveform with 0.15 volt peak to peak in magnitude. Since the piezo has 82 Å/V sensitivity, the gain of the photo-detector can be calculated as follow

$$K_{sensor} = \frac{0.15\ V}{82\ Å/V * 5\ V} \qquad (3.10)$$

$$= 3 * 10^{-4}\ V/Å$$

The output signal of the bicell detector is converted to digital signal before it is sent to the DSP for feedback calculation. According to the theory of digital signal processing, the signal must be bandlimited to the half of the sampling frequency of the system before it is digitized or the signal cannot be accurately reconstructed. Therefore a lowpass filter is placed at the front end of the ADC. This lowpass filter is called an anti-aliasing filter and is used to eliminate the possible aliasing components (high frequency components that exist above the sample frequency) inside the signal from the bicell detector. The details of the anti-aliasing filter have been discussed in Chapter 2, and the transfer function of the filter is

$$L_1(s) = \frac{10^5}{s + 10^5} \tag{3.11}$$

Since the anti-aliasing filter has little effect on the frequency range of interest of the signal, which is from 0 Hz to about 16 kHz, it is neglected from the overall system transfer function.

Same as the DAC discussed in the previous section, the resolution of the ADC has to be considered in the development of the system transfer function. Since a 12 bit ADC is employed with -5 V to +5 V voltage supplies, the conversion factor, named K_{ADC}, is

$$K_{ADC} = \frac{2^{12}}{10} \left(\frac{\text{bit}}{\text{volt}}\right) \tag{3.12}$$

Typical digital control system analysis models an ADC as a switch, or called sampler, with sampling period, T. The sampler can be mathematically represented as a unit impulse train with the following transfer function as

$$S(t) = \sum_{n=-\infty}^{\infty} \delta(t - nT) \tag{3.13}$$

where T is the sampling period of the sampler. A sampled signal is obtained by multiplying the continuous signal with S(t). For instance, the error signal e(t) in figure 3.6 is sampled and becomes $e^*(t)$ as

$$e^*(t) = \sum_{n=0}^{\infty} e(nT)\,\delta(t - nT) \tag{3.14}$$

In the above transfer function, only the positive side is considered, i.e., n is greater than or equal to zero. And the Laplace transform of the signal $e^*(t)$ is

$$E^*(s) = \sum_{n=0}^{\infty} e(nT)\,\varepsilon^{-nTs} \tag{3.15}$$

Letting $z = \varepsilon^{Ts}$, the $E^*(s)$ can be z transformed to

$$E(z) = \sum_{n=0}^{\infty} e(nT)\,z^{-n} \tag{3.16}$$

$$E(z) = e(0T) + e(T)\,z^{-1} + e(2T)\,z^{-2} + \ldots \tag{3.17}$$

Equation (3.21) shows that z transform of a sampled signal represents the sequence of the signal at each sample, T, and each (z^{-1}) term is one sample time delay of the sequence. Therefore, the e(0T) is the present error signal and e(T) is signal one sample period ago, so on and so on.

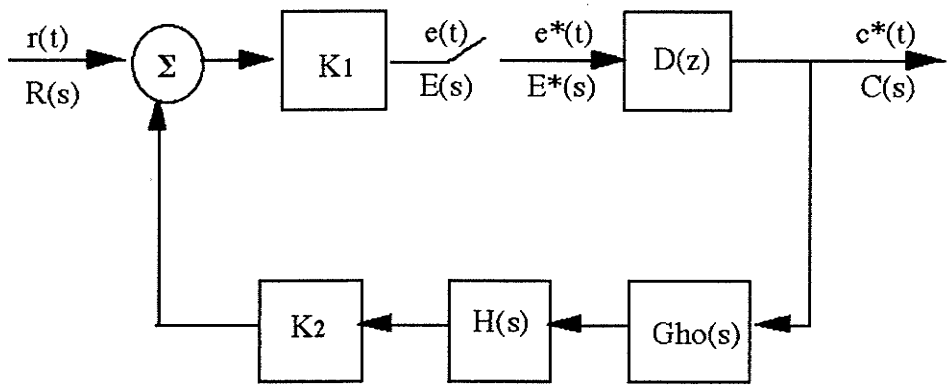

Figure 3.6 Simplified block diagram of the AFM control system

3.2.3 Digital PI compensator

Since the PI compensator is software coded inside the DSP-program, the numerical approximation of the compensator algorithm is required. The z transform, an analogous transform technique of Laplace transform in the continuous system, is employed which provides a mathematical representation for digital control algorithms and sampled signals. The z transform of a digital algorithm is usually derived from the difference equations of the system while the z transform of the discrete signal can be obtained from the closed form of the signal time sequence. The z-transform of the PI compensator is shown below which is derived from right-side rectangular rule approximation method

$$D(z) = K_p + \frac{K_I T}{1 - z^{-1}} \qquad (3.18a)$$

$$\text{or} \qquad D(z) = (K_p + K_I T)\frac{\left(z - \dfrac{K_p}{K_p + K_I T}\right)}{(z - 1)} \qquad (3.18b)$$

The z-transform of the PI compensator can be found in most digital control text books [7]. Appendix B in this thesis derives the equation (3.18) using the right-side rectangular rule. Equation (3.18b) shows that the PI compensator has a pole at 'z' equals one and the zero position depends on the value of K_P and K_I.

3.2.4 Transfer Function of the Closed Loop System

Figure 3.6 is a simplified block diagram of figure 3.5 (b) where the blocks on the feedback path are grouped together. The system includes analog and digital components with both continuous and digital signals flowing inside the loop. In order to link these two different domains together to develop an overall system transfer function, a synchronous fictitious sampler is used to produce a mathematical sampling effect to the analog components and continuous signals so that only the values at the samples are considered. Pulse transfer functions [7,8] of the analog components and the continuous signal are used to represent the output of the fictitious sampler.

The development of the transfer function starts from the error signal E(s) as follow

$$E(s) = K_1 R(s) - K_1 K_2 E^*(s) D(z) G_{ho}(s) H(s) \qquad (3.19)$$

where $K_1 = K_{sensor}\ K_{ADC}$ and $K_2 = K_{amp}\ K_{DAC}\ K_{dc}$.

The analog component, $G_{ho}(s)\ H(s)$, and the continuous signals, E(s) and R(s), are first starred transformed as shown previously and are notated by '*'. Then equation (3.19) is rewritten as

$$E^*(s) = K_1 R^*(s) - K_1 K_2 E^*(s) D(z) \overline{G_{ho}(s) H(s)}^* \qquad (3.20)$$

$$E^*(s) = \frac{K_1 R^*(s)}{1 + K_1 K_2 D(z) \overline{G_{ho}(s) H(s)}^*} \qquad (3.21)$$

The output, C(z), equals the sampled error signal $E^*(s)$ multiplied by the digital

compensator D(z) shown as follow

$$C(z) = E^*(s)\, D(z) \tag{3.22}$$

Combining the equations (3.26) and (3.27), the overall system transfer function is

$$\frac{C^*(s)}{R^*(s)} = \frac{K_1\, D(z)}{1 + K_1 K_2\, D(z)\, \overline{G_{ho}(s)\, H(s)}^*} \tag{3.23}$$

Letting z equal ε^{Ts}, equation (3.28) can be rewritten as

$$T(z) = \frac{K_1\, D(z)}{1 + K_1 K_2\, D(z)\, H(z)} \tag{3.24}$$

where H(z) equals the pulse transfer function of the product $G_{ho}(s)\, H(s)$.

3.3 System Analysis

The primary requirement for the feedback system in AFM application is to have zero steady state error at the output with step input function. This requirement can be accomplished by cascading the AFM system with an integrator since the compensated system has high gain at low frequency. Besides zero steady state error, the swiftness of the feedback system response and the stability of the feedback system are two significant factors that affect the performance of the AFM system. A fast and stable feedback system allows the AFM to scan at a higher scan rate which reduces the time to acquire an image.

This section discusses the behavior of the feedback system using the transfer function obtained from the previous section. The root locus of the characteristic equation of the system will be used to determine the limit of the gain for which the system remains stable. The swiftness of the system response can be observed from the transient step response with different feedback gains. Also, the step response of the system reveals the

stability of the feedback system. The system behavior from simulation and experiments is obtained and compared.

3.3.1 Root Locus Diagram of the Open Loop Transfer Function

As shown in the block diagram in figure 3.6, the input of the closed loop AFM system is the height of the features on the sample surface and, therefore, the unit of the input is in 'angstrom'. On the other hand, the output of the system is the output of the digital compensator and the unit of the output is in 'bit'. Since the input and output of the system have different units, the block diagram in figure 3.6 is slightly changed in order to simplify the simulation. The dc gain, K_2 on the feedback path is moved to the feedforward path so that the input and output of the block diagram have the same unit, angstrom. A modified closed loop system block diagram for simulation is shown in figure 3.7 and the transfer function becomes

$$T(z) = \frac{K\, D(z)}{1 + K\, D(z)\, H(z)}$$

$$\text{where } K = K_{sensor}\, K_{amp}\, K_{dc}\, K_{ADC}\, K_{DAC}\, (K_P + K_I T) \qquad (3.25)$$

D(z) is the transfer function of the digital PI compensator and H(z) is the normalized pulse transfer function of the analog components on the feedback path. K is the open loop gain of the system. The gain of the digital PI compensator is excluded from the transfer function, D(z), and put into the open loop gain, K.

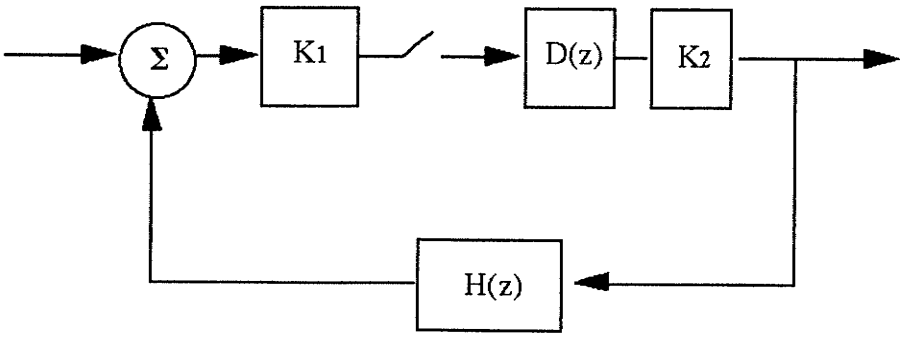

Figure 3.7 Modified block diagram of the AFM control system for simulation

Numerical values are substituted into the variables inside the transfer function in order to carry out the simulation. For the digital controller developed in our lab, the sampling frequency of the system is determined by the cycle time of the feedback subroutine inside the DSP-program, which is 33 kHz. Using the software 'cc', the pulse transfer function, H(z) is obtained and shown as follows:

$$H(z) = \frac{0.93\,(z^2 + 1.59\,z + 0.49)}{(z^2 + 1.84\,z + 0.87)\,(z - 0.22)} \tag{3.26}$$

and the characteristic equation of T(z) is

$$1 + K\,D(z)\,H(z) = 1 + K\,\frac{0.93\,(z^2 + 1.59\,z + 0.49)\,(z - \frac{K_P}{K_P + K_I T})}{(z^2 + 1.84\,z + 0.87)\,(z - 0.22)\,(z - 1)}$$

$$\text{where } K = (3*10^{-4})\,(32)\,(82)\,(\frac{2^{12}}{10})\,(\frac{10}{2^{16}})(K_P + K_I T)$$

$$= 0.05\,(K_P + K_I T) \tag{3.27}$$

Equation (3.27) shows that the K_P and K_I determine the zero position of the open loop

transfer function. Also the open loop gain equals the product of a constant, 0.05 and the sum of K_P and $K_I T$. The following system analysis omits the proportional controller by letting K_P equals zero.

Figure 3.8 shows the root locus plot of the characteristic equation of the compensated system using the software package 'cc'. According to typical digital control theory, the root of the characteristic equation must be within the unit circle or the system will become unstable. From the root locus plot, the gain at which the root loci cross the unit circle is about 0.6. Since the system starts oscillate when the root locus approaches the circumference of the unit circle, the open loop gain, K, must be less than 0.6 for a stable system.

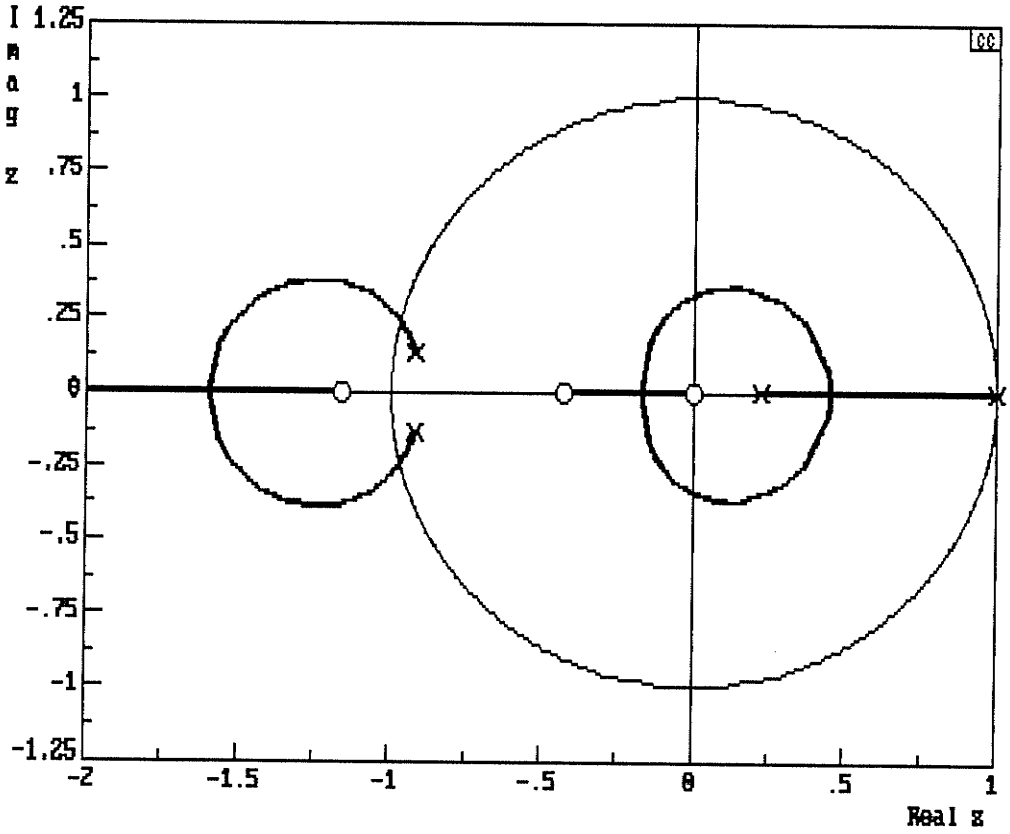

Figure 3.8 Root locus diagram of the characteristic equation 1 + KD(z)H(z) where D(z) is the transfer function of the integrator.

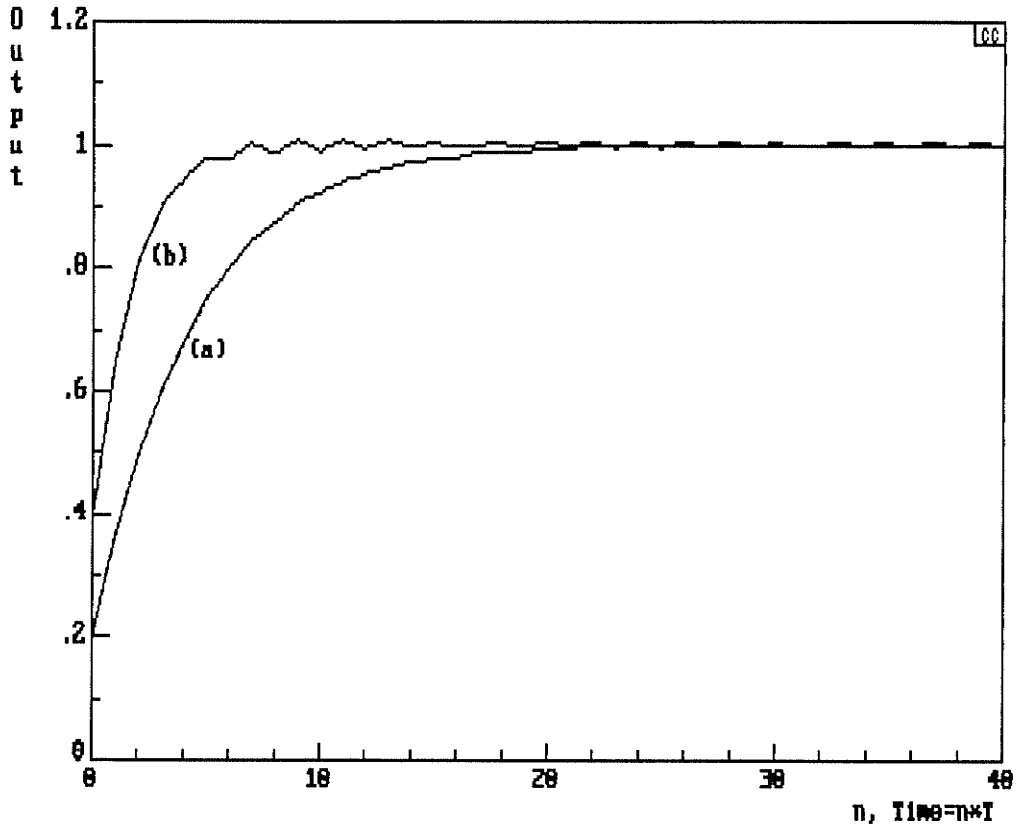

Figure 3.9 Simulated step response of the integral compensated AFM system with the open loop gain equals (a) 0.2, (b) 0.4.

3.3.2 Transient Response of the Compensated System

The transient step response of the system is acquired using the closed loop transfer function in equation (3.25). Figure 3.9 shows two step response curves that use loop gain equal 0.2 and 0.5 respectively. Ringing exists in the response curve with gain equals 0.5, which indicates that the gain is approaching the maximum limit.

The simulated result gives an approximation of the open loop gain that gives the fastest and stable response. The real system response is acquired to verified the correctness of the model developed. The step input is generated by changing the 'set-point' with a constant, and then the feedback system will adjust the sample height until the signal from the bicell detector equals the set-point. The input of the system is no longer the desired sample height as in figure 3.7, but is the set-point. The revised block diagram is shown in figure 3.10.

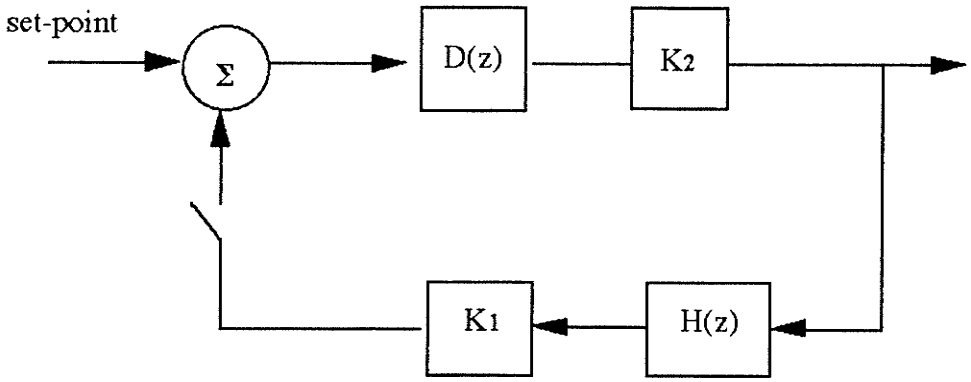

Figure 3.10 Modified block diagram of the physical AFM control system for acquiring step response of the system

The digital PI compensator is software coded inside the feedback subroutine of the DSP-program as follow

$$sum = sum + error;$$

$$Z_{out} = K_p * error + K_I' * sum; \qquad (3.28)$$

The variable *error* is the difference between the signal from the bicell detector and the *set-point* and the variable Z_{out} is the value output to the DAC. The variable error is accumulated and put into the variable *sum* during the scanning. The sum is then multiplied with a variable K_I', which is the integral gain of the PI compensator. The variable K_I' is the one tenth of the value input by the AFM user from the keyboard or from the mouse. The integral gain K_I' is equivalent to the value $(K_I T)$ in the mathematical expression shown in equation (3.25) or can be named effective integral gain. The relationship between the value K_I' and $(K_I T)$ can be shown as

$$\begin{aligned} Z_{out} &= K_p * error + K_I * \Sigma\ (error * T) \\ &= K_p * error + (K_I T) * \Sigma\ error \\ &= K_p * error + K_I' * sum \end{aligned}$$

Three step response curves are shown in figure 3.11 which corresponds to different integral gains: 2, 4, and 8. The rate of response increases as the gains increases. In order to compare the experimental and simulation step response, curve (b) in figure 3.11 and curve (a) in figure 3.9 are put together in figure 3.12 since they have the same open loop gain. The curve (b) in figure 3.11 is acquired using integral gain at 4 which corresponds to open loop gain 0.2 using equation (3.27) and, therefore, is the same as that of curve (a) in figure 3.9. Figure 3.12 shows that these two curves almost overlap each other and both have rise time at about 10 sample cycles. In addition, figure 3.13 compares the rise times of the step response curves from the simulated and physical system with different integral gains. The rise time is in term of the number of samples the systems require to reach 90 percent of the steady state level.

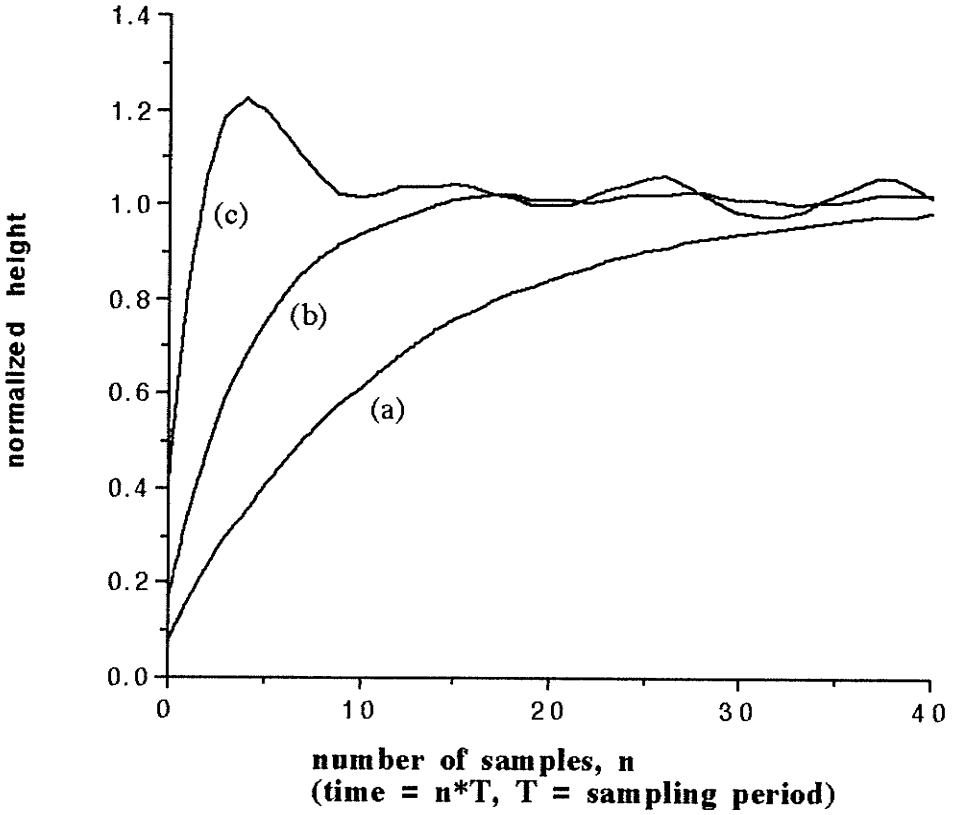

Figure 3.11 Step response curves of the physical AFM system with integral gain equals (a) 2, (b) 4 and (c) 8. The sampling period of the system is 30 μsec.

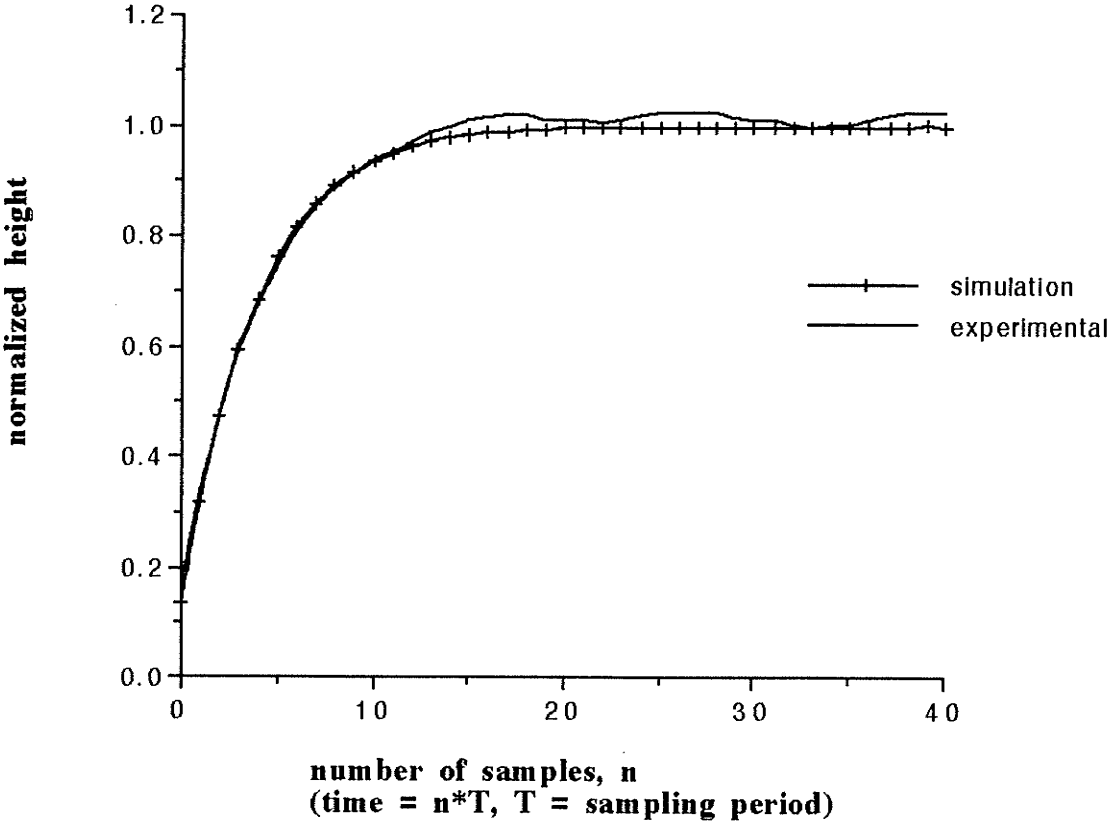

Figure 3.12 Step response curves from simulation and physical AFM system with integral gain equals 4.

In addition, figure 3.13 shows that the results from the model are very close to that from the physical system since the step response curve has similar rise time using the same integral gain. Figure 3.12 and 3.13 show that the model developed in this chapter is able to estimate the behavior of the compensated system with the integrator when the system is stable. However, the curve (c) in figure 3.11 has 20 percent overshoot which does not exist in the simulated step response curves in figure 3.9. The overshoot may be accounted for the phase lag from the anti-aliasing filter that has been omitted in the system transfer function. Furthermore, the step response from the physical system has 3 kHz ringing at steady state. This ringing is believed to be coupled from the mechanical resonance of the microscope stage and appears when the tip is off the center of the piezo tube scanner.

As the proportional controller comes into effect, the limit of the open loop gain changes as the position of the zero varies. It will be a trial and error to locate the proper value of proportional and integral gains which give the best system performance. The next chapter introduces a method which uses a genetic algorithm to locate the proportional gain and the integral gain that give optimal system performance.

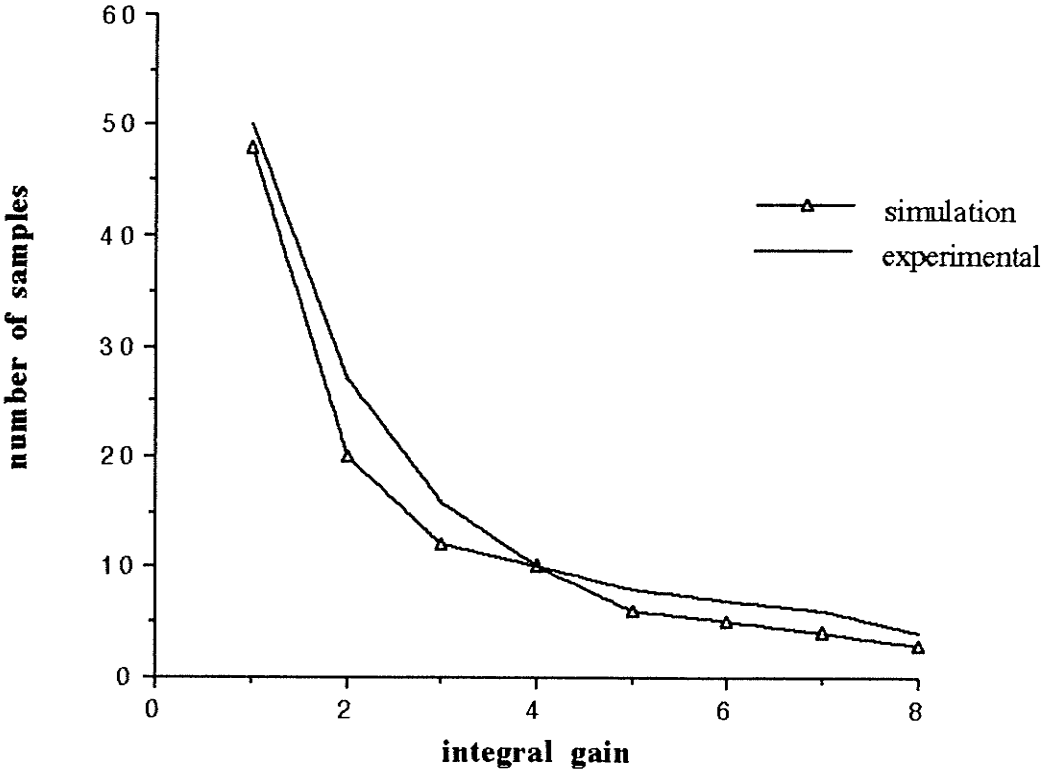

Figure 3.13 Comparsion of the rise time of the step response curves from simulated and physical AFM system versus different intergal gains used. The rise time is in term of the number of samples required in which each sample equals 30 μ sec

3.4 Chapter Summary

The behavior of the closed loop AFM control system is dominated by the piezo tube scanner and the proportional-integral (PI) compensator. The closed loop AFM system with the piezo tube scanner alone has finite steady state error at the output to the step input function. In addition, the uncompensated AFM system is highly underdamped in which the oscillation at the output settles after 2 msec. Cascading the AFM system with the PI compensator, the compensated AFM system has zero steady state error at the output for a step input function. However, the PI compensator reduces the bandwidth of the system which reduces the rate of the system response to the input. The rate of the system response can be increased by increasing the feedback gains of the PI compensator. However, the resonant peak of the piezo must be kept below 0 dB or the system will become unstable.

A model of the digital PI compensated AFM system is developed from the physical hardware system. Comparison of the step response from the simulation and experiments shows that the model is able to estimate the behavior of the AFM system. Using the integral controller only, simulation and experimental results show that the system has the fastest step response with integral gain of 0.4; the rise time is about 10 sampling cycles. The sampling period of our system is about 30 μsec.

Reference

[1] Program cc, Systems Technology Inc.

[2] S. Grafstrom, J. Kowalski and R. Neumann "Design and detailed analysis of a scanning tunnelling microscope," *Meas. Sci. Technol.* **1**, 139 (1990).

[3] L. Tao and J. Maps "Simple determination of the frequency response of piezoelectric tubes," *Rev. Sci. Instrum.* **64** (5), 1367 (1993).

[4] D. Pohl "Some design criteria in scanning tunneling microscopy," *IBM J. Res. Develop*, **30** (4), 417, (1986).

[5] K. Westra "Atomic force microscopy of thin films," *PhD thesis*, University of Manitoba, 1994.

[6] C. L. Philips and H. T. Nagle, Jr. *Digital Control System Analysis and Design*, Prentice-Hall, N.J., pp. 328 - 340, 1984.

[7] C. L. Philips and H. T. Nagle, Jr. *Digital Control System Analysis and Design*, Prentice-Hall, N.J., pp. 548, 1984.

CHAPTER 4

AUTOMATED AFM SCANNER

The same as users of any other instruments, AFM users usually have a desire to operate the system at its optimal performance. As discussed in the previous chapters, the AFM makes use of a feedback control system to track the surface contour and acquire the topographic images of the samples. If the feedback system has a fast and stable response to the input (which is the height of the features on the sample surface), then the AFM system is able to acquire the whole topographic image in the minimum period of time. As shown in the previous chapter, if a PI compensator is used in the feedback system, then the swiftness and stability of the system are determined by the feedback gains of the compensator. Although the speed of system response can be increased by raising the feedback gains, the system will become unstable if the feedback gains are too high.

Besides the feedback gains, the scan speed is another factor that affects the performance of an AFM. In the conventional AFM, the piezo is moving along x-direction with constant speed during scanning. This scan speed must be chosen very carefully so that the feedback system has enough time to adjust the sample height. Otherwise, the image obtained will become unreal [1] since the output of the feedback system is not able to track the surface. There are two major factors that affect the tracking ability of the system, which are the scan speed and the size of the tip as compared to the size of the features [1]. The effect of the tip size is beyond the scope of this thesis and the discussion in this chapter will focus on the scan speed only and assume the apex of the tip is small enough to track the features.

Usually, the AFM users use "trial and error" methods to search for the proportional

gain and integral gain that give the fastest system response. They increase both the proportional and integral gains until the system becomes unstable. Then, they decrease both gains slowly until the system regains stability and starts scanning. On the other hand, the choice of the scan speed largely depends on the experience of the AFM operators. It is believed that the scan speed is related to the number of features across a scan row [1]. When many features exist along a scan row, slow scan speed should be used in order to allow the feedback system to have enough time to compensate for the sample height.

I have developed an automated AFM scanner which is able to search for the gains that give a fast and stable feedback response and vary the scan speed according to the surface roughness during scanning. The automated scanner requires virtually no knowledge about the control system from an AFM user to proceed scanning at optimal system performance. This chapter discusses the development and implementation of this automated scanner and the discussion is divided into two halves. The first half of the chapter describes the concept and implementation of the searching algorithm that is employed for searching for the optimal gains. The second half of the chapter discusses how the adaptive scanner changes the scan speed according to the surface roughness and maintains a finite error margin at the output.

4.1 Searching for Optimal Feedback Gains

Chapter 3 has mentioned that the behavior of a PI compensated AFM system varies with different combinations of the proportional gain and the integral gain. Using the root locus diagram to find the feedback gains that give the fastest stable system is largely based on trial and error. Instead, a combinatorial searching algorithm can be used to search for these feedback gains and these gains are named optimal gains. A measuring index, named performance index, is defined in order to determine the performance of the system. Since a

stable system with the fastest system response has the smallest difference between the desired and actual output as observed from the shape of the step response curves, the performance index can be defined as the sum of the difference between the desired and actual output response at each sample along a step response curve. 256 data points are collected for each step response curve which corresponds to 7 msec running time in our experiments. The calculation of the performance index will be further discussed in detail in section 4.3.

The surface plot in figure 4.1(a) shows the relationship between the performance index and the feedback gains. The x and y axes are the proportional gain and integral gain respectively, ranging from 0.2 to 10, while the z-axis is the normalized performance index. Figure 4.1(b) is a zoom in of the surface plot to show the detail inside the valley. However, it is very difficult to perceive the relationship between the performance index and the variations of the feedback gains from the 3-D graphs shown in figure 4.1 (a) and (b). The cross sections of the surface plot shown in figure 4.2 (a) and (b) attempt to illustrate a general relationship between the performance index and one of the feedback gains while holding the other feedback gain constant. Figure 4.2 (a) is the cross section of the surface plot at integral gain equal to 3.4. The performance index gradually increases from about 200 to 450 as the proportional gain increases from 0.2 to 10. On the other hand, figure 4.2 (b) shows the cross section of the surface plot at proportional gain equal to 0.8. The performance index is very high for small integral gain and the value of the index drops when the integral gain increases. However, the performance index increases again when the integral gain keeps increasing. Recall the analysis in the previous chapter, the system bandwidth is small for small integral gain in which the system has a large rise time in step response. As the integral gain increases, the rise time of the step response increases with the system bandwidth. However, when the integral gain becomes too high, the system become unstable and the difference between the desired and actual output increases.

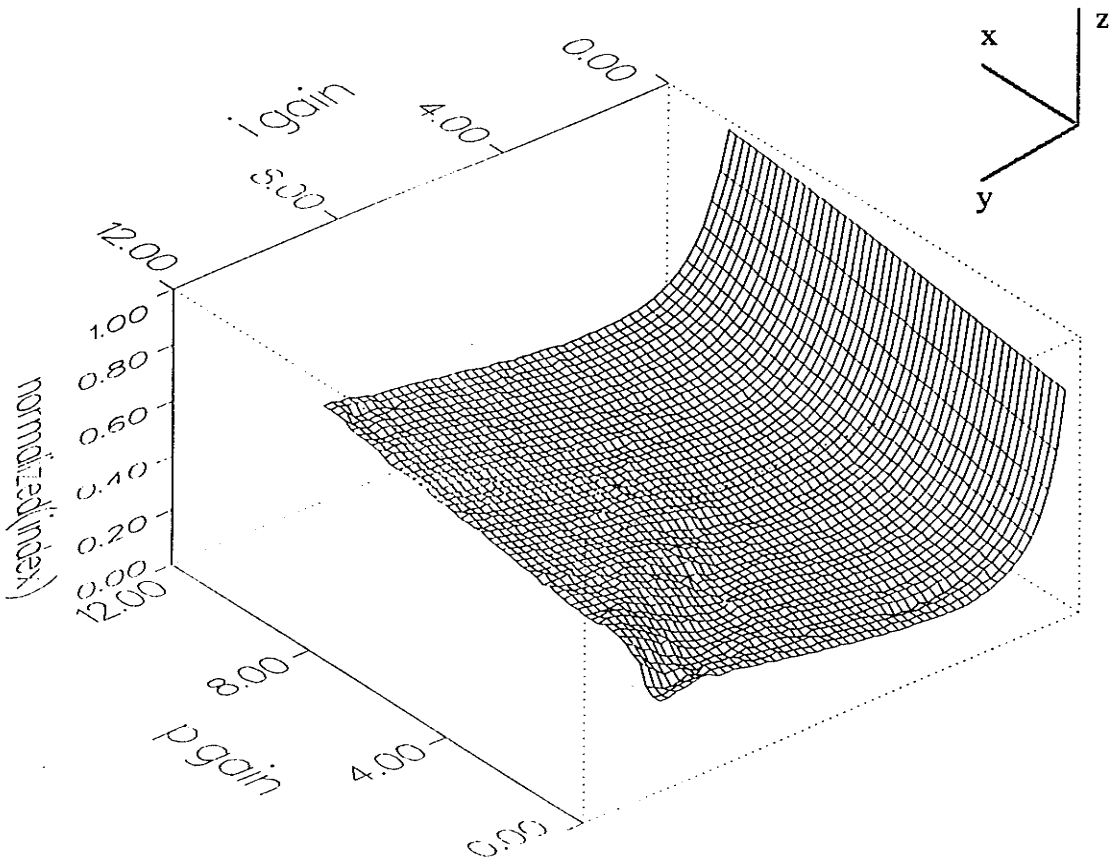

Figure 4.1 (a) The relationship between the performance index and the feedback gains is shown as a 3-D surface plot. The horizontal axes, x and y, are the proportional gain and the integral gain respectively while the vertical axis, z, is the performance index. The feedback gains range from 0.2 to 10, with a step of 0.2.

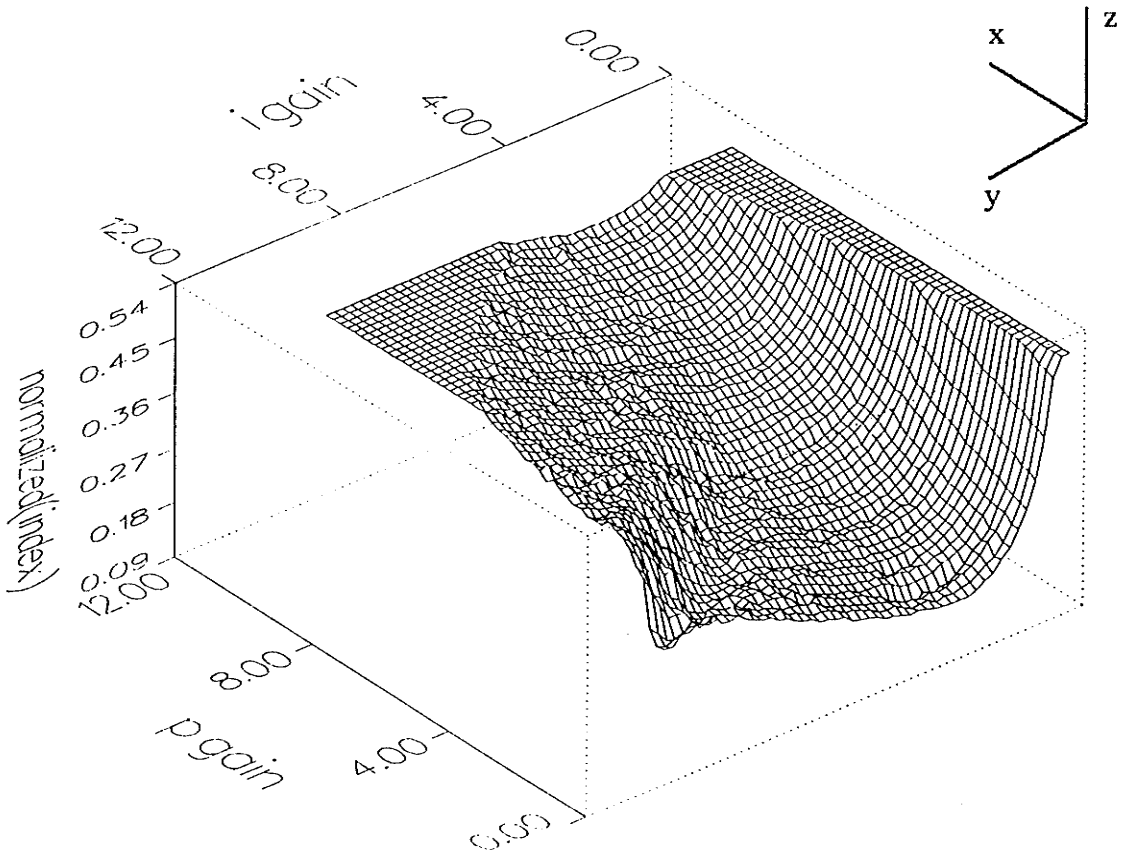

Figure 4.1 (b) Zoom in of the surface plot in figure 4.1 (a) to show the detail inside the valley. Normalized performance index values larger than 0.5 are clipped.

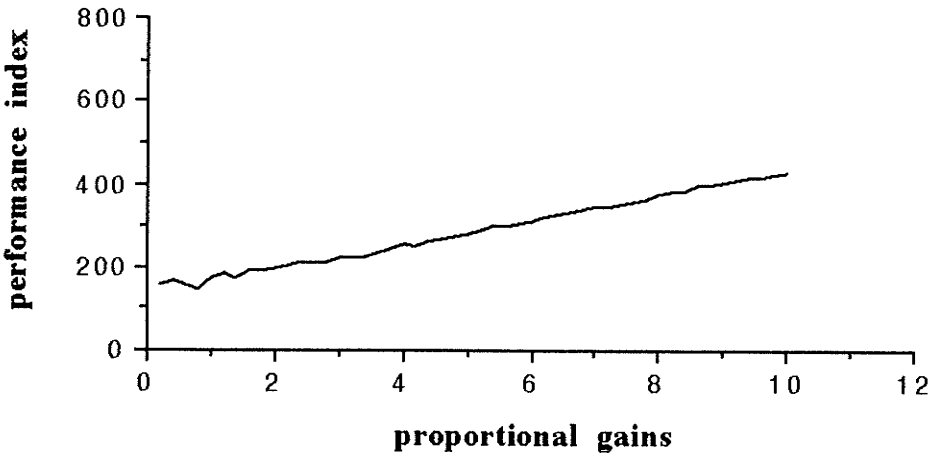

Figure 4.2 (a) Cross section of the surface plot in figure 4.1 at integral gain equal to 3.4.

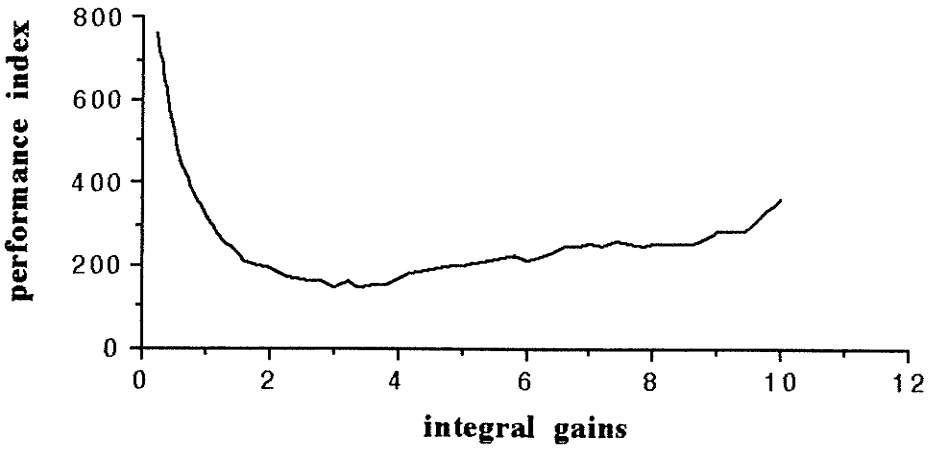

Figure 4.2 (b) Cross section of the surface plot in figure 4.1 at proportional gain equal to 0.8.

The surface plots in figure 4.1 (a) and (b) show that there are many local minima on the surface. The optimal gains are located at the global minimum of the plot. In addition, the function of the surface plot is unknown and varies with sample and AFM system. Given these two problems, the searching method selected must be able to locate the global minimum of the surface without being trapped by the local minima, and must be able to perform without much knowledge of the problem itself. Among various optimization algorithms, a genetic algorithm was selected for the following two reasons: 1) a genetic algorithm requires little knowledge of the problem itself; 2) the problem and solution in this case can be formulated into genetic configuration [2].

Baselt et al. have developed a general-purpose "evolutionary" algorithm in which the algorithm repeatedly searches for the control parameters that give the AFM optimal performance [3]. The algorithm that Baselt et al. have developed is very similar to a genetic algorithm except that the algorithm they developed gives a single solution every iteration. A genetic algorithm gives a set of the best solutions obtained in the current iteration and uses this set of solutions to find better solutions in the next iteration. The following sections give an overview of the genetic algorithm and the implementation of the genetic algorithm on optimal gains searching. Experimental results will be presented to show the performance of algorithms in this application.

4.2 An Overview of Genetic algorithm

Genetic algorithms, originally developed by Holland [4], have recently been used in solving various optimization problems such as standard cell placement [5], traveling salesman problem [6], and groundwater monitoring problems [7]. The algorithm is analogous to the biological evolution in which species evolve to better adapt to the

environment throughout generations. Most of the terms using in genetic algorithm are taken from the biological counterpart.

The algorithm starts with an initial set of solutions called the population. This initial population can be randomly generated or selectively chosen. Each solution in the population is called an individual and is represented by a string of symbols called genes. These genes are the building blocks or segments of the solution and can be altered to generate better solution for the problem.

The genetic algorithm is an iteration process and each iteration is called a generation. In the beginning of every iteration, all the individuals in the population are put into a fitness test to examine their fitness to the environment. In other words, this fitness test examines how good these solutions are to the problem. Those individuals having higher degree of fitness are selected to be parents for producing new individuals for the next generation. Usually, three operators are used for producing offspring and they are called crossover, mutation, and inversion operators.

1) crossover operator:

First, two parents are selected at a time and put into this operator. The operator duplicates and exchanges their genes to produce new individuals. The new individuals contain some genes from each of their parents. The amount of crossover among parents is controlled by a constant called crossover rate and this crossover rate is selected by the user.

2) mutation operator:

After the crossover operator, the new individuals are put into the mutation operator. This operator causes a random changes of the genes in each new individual so that new genes can be introduced. New genes are significant to the existence of the species since better genes that do not exist in the current generation can be generated. A control variable

called 'mutation rate' is used to control the amount of variations of the gene. This control variable must be carefully selected so that useful genes can be created without losing the good inherited from their parents.

3) inversion operator:

Inversion operator alters the sequence of the gene inside an individual. This operator is useful for the problem in which the individual has different adaptation ability to the environment for different gene orientations.

After these three operators, the new individuals are called offspring of the current generation. Afterward, both the parents and the offspring are put into the fitness test again and those individuals having better fitness are selected and form the next generation. A fixed number of individuals are selected so that the size of the population remains the same for all generations. The iterations carry on until some termination constraints are reached such as the number of generations. Theoretically, after a number of generations, 'bad' genes are eliminated and 'good' genes remain. All the individuals in the final generation have good genes and are best adapted to the current environment. In all, genetic algorithm can be summarized as follow

Generate the initial population
Loop for a number of generations
    Fitness Test for every individual in the population;
    Choose Individuals for mating;
    Crossover Operator- duplicates and interchanges the genes among parents;
    Mutation Operator - mutation of genes of the new individuals;
    Inversion Operator - changes the orientation of genes inside the new individuals;
    Fitness Test for all individuals;
    Choose Individuals for the Next Generation;
End Loop.
Final Result.

4.3 Implementation of Genetic Algorithm on Optimal Gains Searching

The first step of the implementation is to code the problem into a genetic configuration. The goal of the genetic algorithm is to find the smallest performance index of the system; the control parameters are the proportional and integral gains. Therefore, each individual is represented by a combination of the proportional and integral gains and these two gains are the only genes in each individual. The population is composed by a fixed number of these individuals. The population and each individual can be represented as follow

$$P(t) = \{X_1^t, \ldots, X_i^t, \ldots, X_n^t\} \tag{4.1}$$

$$X_i^t = \langle P_i^t, I_i^t \rangle \tag{4.2}$$

where P(t) is the population of generation t, and n is the size of the population. The size of population is twenty in our experiments. X_i^t represents the individuals inside the population and each individual contains two genes P_i^t and I_i^t, which is the proportional and integral gain respectively.

In order to generate the initial population, a pair of initial proportional and integral gains are selected. According to the algorithm, the value of the initial proportional and integral gains selected will not affected the final result of the search. However, this initial gains should be low enough so that the system remain stable. This selected gain pair passes itself through the mutation operator nineteen times (the implementation of the mutation operator will be discussed later) to produce the rest of individuals in the initial population. After the initial population is created, the program proceeds with the iterations.

The fitness measurement of the individual is defined by the stability and swiftness

of the step response of the system in which those individuals having smaller performance index have better fitness. The generation of the step input is achieved by changing the set-point value with a constant called 'delta'. Before running the genetic algorithm, the program raises the Z-piezo by 420 Å. The signal change in the photo-detector is recorded and assigned to the constant 'delta'. As the set-point changes by 'delta', the feedback system will adjust the Z-piezo by 420 Å to returns the error signal to zero (the signal coming from the photo-detector equals the set-point value). Using the method mentioned above, M step response curves are generated for each individual and N data points are recorded for each curve. The data points, ε, are the absolute value of the difference between the desire and actual feedback output at each sample. All N data points of each curve are summed up and average giving the performance index of a single curve. Then, the performance index of all M curves obtained are added and averaged giving the performance index of the corresponding individual. The mathematical formula for calculating the performance index of each individual is shown as follow

$$\text{performance index} = \frac{1}{M}\sum_{n=0}^{M-1}\sum_{t=0}^{N-1}\frac{|\varepsilon|_t^n}{N} \qquad (4.3)$$

where ε_t^n is the error at data point t of curve n. In the experiments, twenty step response curves were generated and 256 data points (which is correspond to running for 7 msec) were collected for each curve. Therefore, the value of M and N are 20 and 256 respectively.

After the fitness test, the best five individuals are selected as parents. Among these five parents, two of them are selected at a time and put into the crossover operator. The operator duplicates and exchanges genes among the parents. For instance, if the two selected parents are X_i^t and X_j^t and they can be represented as follows

$$X_i^t = \langle P_i^t , I_i^t \rangle$$
$$X_j^t = \langle P_j^t , I_j^t \rangle \qquad (4.4)$$

After crossover, two new individuals Y_l^t and Y_k^t are produced and can be represented as

$$Y_l^t = \langle P_i^t , I_j^t \rangle$$
$$Y_k^t = \langle P_i^t , I_j^t \rangle \qquad (4.5)$$

Since there are only two genes in each individual, the new individuals have one gene coming from each parent. In other words, the crossover rate is 50 percent in this case. In all, twenty new individuals are created in one generation with no self-crossover occurring among the parents.

In the mutation operator, every gene in each new individual is multiplied by a random number, R. This random number is limited by the mutation rate, p, and is defined by the inequality shown as follow:

$$\frac{1}{(1+p)} \le R \le (1+p) \qquad (4.6)$$

where p is between zero and one. Different random numbers are generated for each gene. After the mutation operator, the genes inside the new individuals Y_l^t and Y_k^t are changed as follows

$$Y_l^t = \langle P_i^t * R_a , I_j^t * R_b \rangle$$
$$= \langle P_l^t , I_l^t \rangle \qquad (4.7)$$

$$Y_k^t = \langle P_i^t * R_c , I_j^t * R_d \rangle$$
$$= \langle P_k^t , I_k^t \rangle \qquad (4.8)$$

where R_a, R_b, R_c, and R_d are the four different random numbers for each gene.

The function of the mutation operator is to allow the individual to jump out of the local minima of the surface plot shown in figure 4.1. The value of the mutation rate, p, is a critical parameter in a genetic algorithm. If the mutation rate is too small, the search will be easily trapped into local minima or it will take a lot of iteration cycles to reach the global minimum. On the other hand, if the mutation rate is too high, the information from the previous iteration will be lost and the algorithm becomes a random search. *J. Grefenstette* proposed using a genetic algorithm to search for the best parameters [8]. Their system has two genetic algorithm loops running simultaneously. The outside genetic algorithm loop (or called meta-genetic algorithm) searches for the optimal parameters and the inside genetic algorithm loop searches for the optimal solution of the particular problem. However, this method required a lot of calculations and iterations which are not appropriate to the AFM application. In the following experiment, 'trial and error' method is used to look for the best mutation rate.

Since there is no sequence of the genes inside each individual, the inversion operator is omitted. After the crossover and mutation operator, the new offspring are called offspring of the current generation.

Finally, all individuals, both old and new, are put into the fitness test and the best twenty individuals are selected to form the population of the next generation as follows:

$$P(t+1) = \{X_1^{t+1}, \ldots, X_i^{t+1}, \ldots, X_n^{t+1}\}. \qquad (4.9)$$

The best individual of each generation is the one that has the smallest performance index within the population. The iteration continues until a fixed number of generations are reached. In the final generation, all the individuals that remain have good genes for surviving in the current environment and the best individual within the generation contains the optimal gains for the system. The program subroutine that implements this algorithm is shown in Appendix C.

4.4 Experimental Procedures, Results and Discussions

Before running the experiments to investigate the performance of the algorithm on optimal gain search, tests were carried out to look for a proper value of mutation rate. The tests were simply done by running the actual optimal gains search and recording the number of generations required to locate the optimal gains. Small initial proportional gain and integral gain were chose to clearly demonstrate the variations of the performance index throughout the generation and these initial gains were set at 0.1 for the tests. The choice of mutation rates were from 0.1 to 0.6. Figure 4.3 shows the results of the test and the mutation rate at 0.4 required the least number of generations to search for the optimal gains. This mutation rate was used for the experiments that followed.

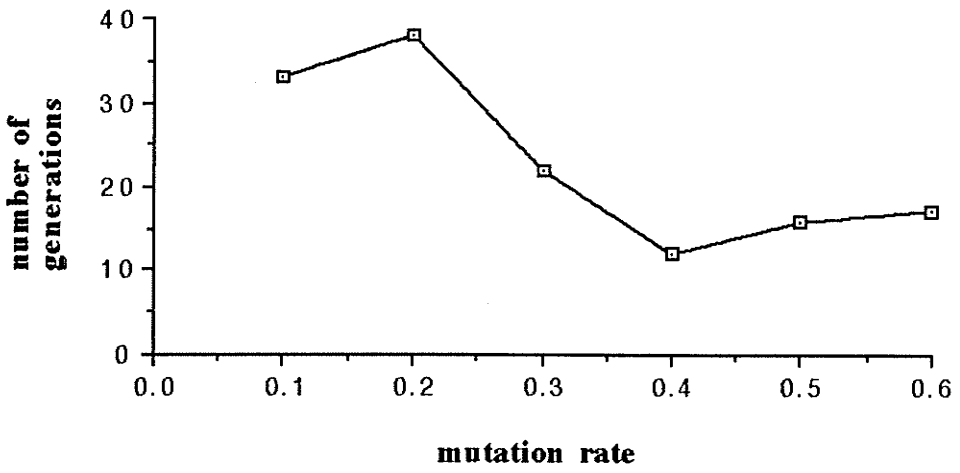

Figure 4.3 Number of generations required to locate the optimal gains at different mutation rates

After the proper mutation rate was found, the experiments were carried out for the optimal gain search. The experiments were started with three different sets of initial proportional gain and integral gain and they were (0.1,0.1), (1,1), and (5,5). Figure 4.4 and 4.5 show the variations of the proportional gain and integral gain throughout forty generations respectively. These two plots show that the search came to the about the same final value in which these final values were supposed to be the optimal gains of the system. From figure 4.4 and 4.5, the optimal proportional gain was about 0.8 and the optimal integral gain was about three for the system. In addition, the two plots show that the number of generations required to locate the optimal gains depended on the initial value. The search required fewer generations to find the optimal gains if the initial gains were closer to the optimal gains. For instance, starting with initial gains at 0.1, the search took fifteen generations to locate the optimal gains while the search took about ten generations for initial gains equal to one.

Recall that for the surface plot in figure 4.1, cross sections were made at the optimal gains obtained in the experiments using the genetic algorithm. Figure 4.2 (a) and 4.2 (b) show the cross sections of the surface plot at proportional gain equal to 0.8 and integral gain equal to 3.4 respectively. The minimum of the curve in figure 4.2 (a) is the optimal integral gain, 3.4, while the minimum of the curve in figure 4.2 (b) is the optimal proportional gain, 0.8.

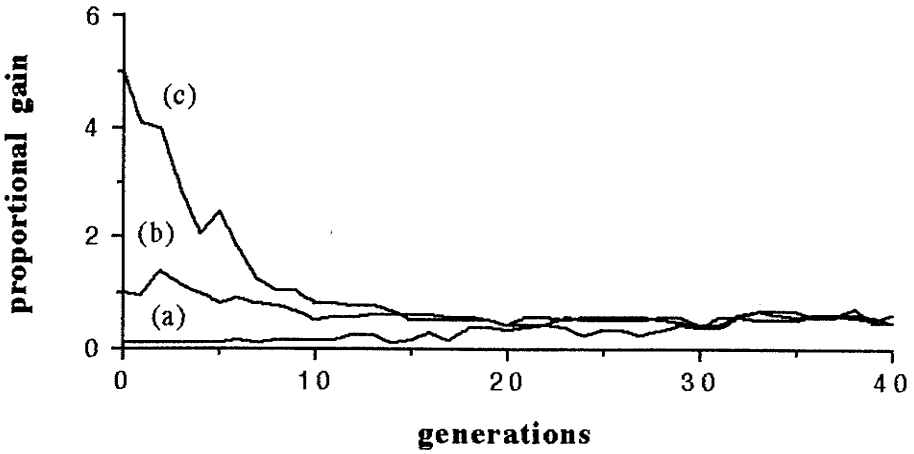

Figure 4.4 Variations of the proportional gain with initial value a) 0.1, b) 1 and c) 5 in 40 generatios.

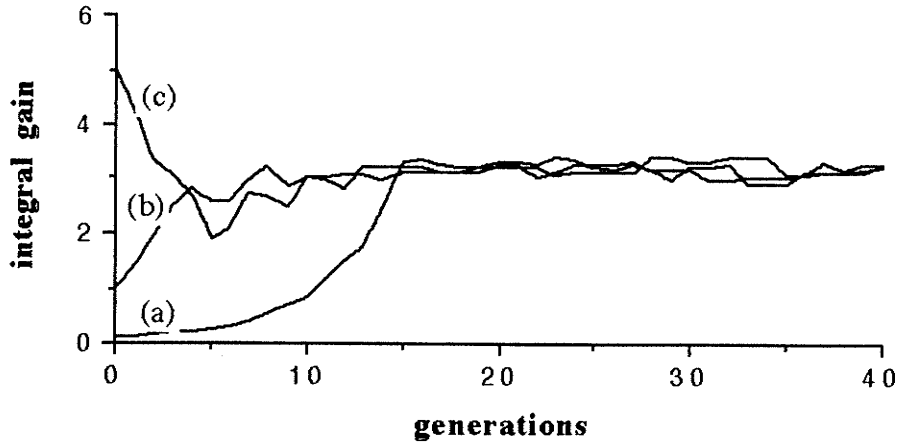

Figure 4.5 Variations of the integral gain with initial value a) 0.1, b) 1, and c) 5 in 40 generations.

4.5 Adaptive Scanner

Most existing AFM control systems move the tip along the x and y direction with constant speed and therefore can be called constant speed scanner. The choice of the scan speed usually depends on the surface roughness of the sample. As the number of features along a scan row increases, scan speed is slower in order to allow the feedback system to have enough time to compensate for the sample height. *P. Heuell, et al.* have developed an adaptive scan generator for the scanning tunneling microscope [9]. Their scan generator enables/disables the x direction movement of the piezoelectric according to the tunneling current to avoid the tip crash into the surface. Based on the idea of this scan generator, an adaptive scanner for the AFM is developed. The adaptive scanner is able to change the scan speed according to the surface roughness of the sample and the images acquired have a finite error margin.

The feedback system is a significant component in the AFM system. Since the feedback system has a finite response time to the input, the tracking ability of the feedback system decreases as the scan speed increases. However, too slow a scan speed would become impractical since the scan time would become unnecessarily long.

The adaptive scanner varies the scan speed according to the surface roughness of the sample. By varying the scan speed, the adaptive scanner is able to ensure that each pixel of the image acquired is within a known finite error margin. The next section discusses the concept of the constant speed scanner and the adaptive scanner. Then, the scanning results obtained from these two scanners will be compared and discussed.

4.6 Descriptions of the Constant Speed Scanner and the Adaptive Scanner

In typical analog controllers, the x and y direction movement of the piezoelectric is driven by a triangle-wave generator while the vertical displacement is controlled by an analog feedback circuit [10]. The period of the triangular wave determines the scan speed. For the digital controller used for this thesis, the only control parameter for scanning is the number of Z-feedbacks as discussed in Chapter 2. The term Z-feedback stands for one iteration that the feedback subroutine inside the dsp-program that calculates the sample height according to the error signal. The error signal is the difference between the signal coming from the photo detector and the set point. The tracking ability of the feedback system and the time required to acquire an image is proportional to the number of Z-feedbacks carried out at each pixel. Figure 4.6 repeats the step response curve shown in Chapter 3. The curve shows that the feedback output is getting closer to the input as the number of Z-feedbacks increases, or in other words, the percentage of error at the feedback output decreases as the number of Z-feedbacks increases.

4.6.1 Constant Speed Scanner

Constant speed scanner is used in most existing AFM controllers. The idea of the scanning algorithm is taken from the typical analog controllers, which use a triangle-wave generator to control the x and y direction movement of the piezoelectric. Using this scanner, the system takes the same amount of time to acquire every row of data for the entire image and the scan time depends on the scan speed selected.

The constant speed scanner implemented in our system runs a fixed number of Z-feedbacks at each pixel point. Since both the number of pixel points across a row and the time required to run a single Z-feedback are constant, the total time taken to scan a row will be fixed for the entire image. Thus, the total scan time for a row can be calculated by

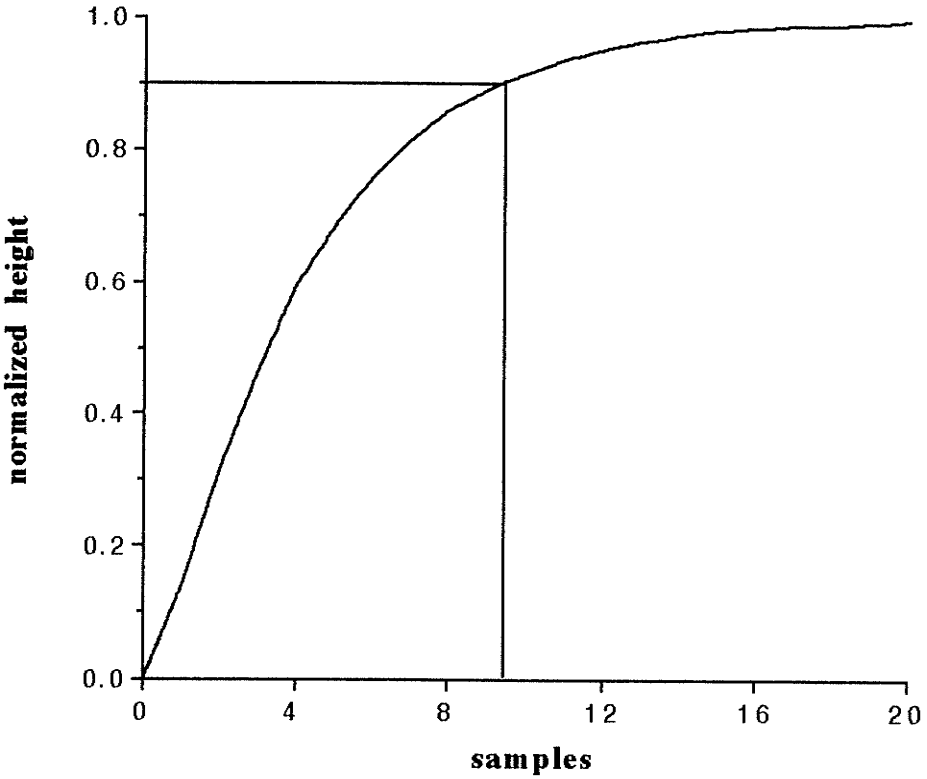

Figure 4.6 Simulated step response from figure 3.12. The straight lines indicate that the output has 90 percent accuracy after running for 9 samples.

multiplying the number of the pixel points in a row, the number of Z-feedbacks at each pixel point, and the time required for one Z-feedback:

total time = number of pixels * the number of Z-feedback
* time required per Z-feedback.

Figure 4.6 shows that using a particular number of Z-feedbacks, the feedback output reaches a corresponding percentage of the actual feature height. In other words, an image has the same percentage of error at each pixel for a particular scan speed. This scanning algorithm has two major drawbacks:

1) Since every pixel has the same percentage of error for a particular scan speed, the absolute error of the pixel can be very large if the feature has a steep slope. For example, figure 4.6 shows that 12 Z-feedback gives 5% of error at the output. If the feature has height equal to 100 Å, then the error at the output will be 5 Å; however, if the feature is 1000 Å in height, then the pixel will have error as high as 50 Å.

2) On the other hand, if the sample surface is smooth, there can be too many Z-feedbacks at the pixel without any significant gain in accuracy at the output.

4.6.2 Adaptive Scanner

Instead of having a fixed number of Z-feedback at each pixel point as in the constant speed scanner, the adaptive scanner is developed which keeps on carrying out the Z-feedback at each pixel point until the error signal is less than a constant called 'error-threshold'. Figure 4.7 shows the flow chart of this scanner. Since the actual feature height is unknown prior scanning, the adaptive scanner makes use of the relationship between the error signal and the feedback output. First, the error signal decreases as the feedback output is getting closer to the actual feature height. Also, the value of the error signal reflects the difference between the feedback output and the actual feature height. By

specifying the error-threshold at the error signal, the adaptive scanner is able to ensure that the feedback output has achieved a finite error margin before it proceeds to the next pixel.

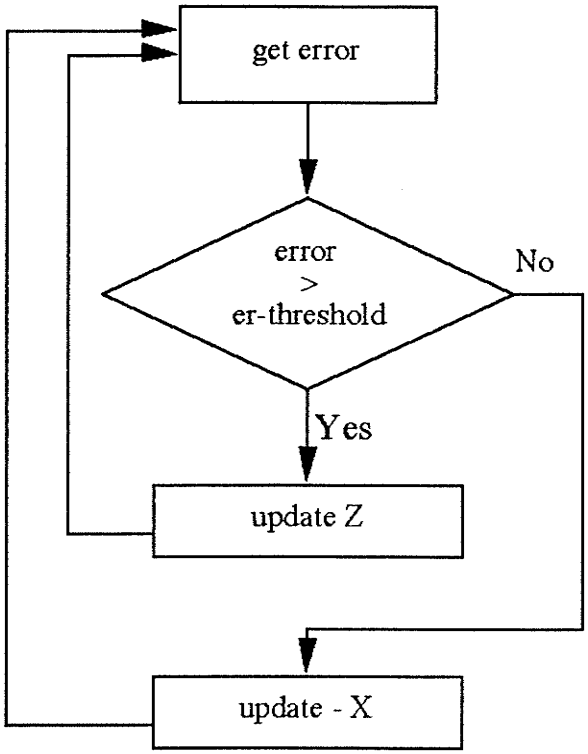

Figure 4.7. Flow chart of the adaptive scanner

For instance, the error signal changes by 1 bit when the sample height varies by 8 Å in our system. If the error signal captured is 10 bits when the tip moves to a new point, then that means the feature is about 80 Å in height with respect to the previous pixel. If the error-threshold is set at 2 bits, then the scanner will keep on adjusting the sample height until the magnitude of the error signal equals 1 bit. The actual sample height is then different from the desired one by less than or equal to 8 Å. Also, since the feedback output is also used for image formation, the pixels on the image acquired have a finite error

margin, which is 8 Å. However, in order to obtain the same absolute value of the error signal at each pixel, the scanner has to run more Z-feedbacks for tall features than that of the short one.

The adaptive scanner has three major advantages: 1) any feature that causes the bicell detector to vary by more than the error-threshold will be captured; 2) the image data acquired at the pixel points has a finite value of error margin; 3) the scan time is less when the surface is smooth since less Z-feedback is needed to acquire image data within the error margin.

4.7 Experimental Results from the Scanners

A diffraction grating sample which has both a smooth surface and steep slope features was chosen in order to demonstrate the pros and cons of the two scanners. Besides the surface topography, the error at each pixel along a row of data was also recorded since the error value is significant in illustrating and comparing the performance of the two scanners. The unit of the error is in bits since the signal is digitized by the ADC in the digital controller when it is being recorded, and one bit corresponds to about 8Å variation in the sample height.

4.7.1 Results from the Constant Speed Scanner

Figure 4.8 shows the surface topography and the cross-section of the sample taken at 2 Hz scan speed. The surface topography shows that the sample looks like a checker board. The cross-section shows that the flat plateau has features about 100 Å high and the highest feature is about 600 Å in height. Figure 4.9 shows the error of each pixel along single row scanning at 2 Hz. The plus and minus sign of the error correspond to the features that have negative and positive slope respectively. The average value of this error

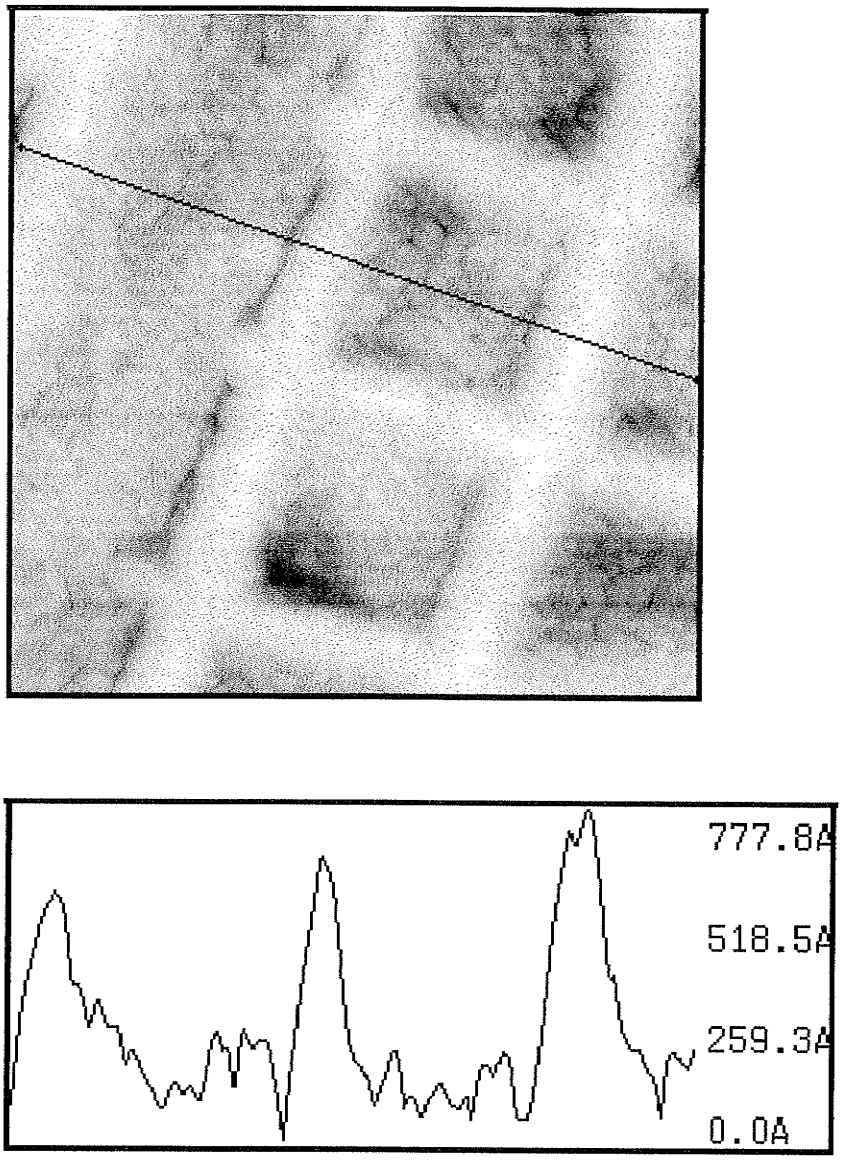

Figure 4.8 Surface topography of the grating sample and the cross section of the image. The image was taken by the constant speed scanner with scan rate equals 2 Hz. The scan size of the image is 1 μm by 1 μm.

profile is 1.2 bits and the maximum error is 4 bits in magnitude. Putting the cross-section of the image and the error profile together shows that the error spikes appear when the feature is steeply sloped. As the scan speed increases to 6 Hz, the error profile of the single line scan has average error and maximum error equal to 2.6 bits and 9 bits respectively.

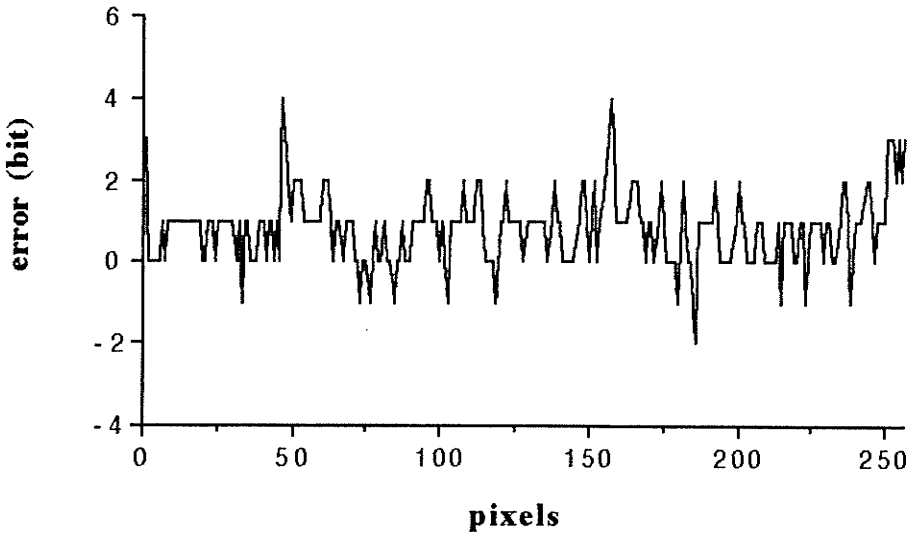

Figure 4.9 The error exists along a row of image data scanning at 2 Hz.

4.7.2 Results from the Adaptive Scanner

Figure 4.10 shows the error profile of the single row scanning with error-threshold equal to 2, 4 and 6 bit. These error profiles show that the errors are bounded by the corresponding error-threshold. Figure 4.11(a) and 4.12 (a) show the cross-sections of the

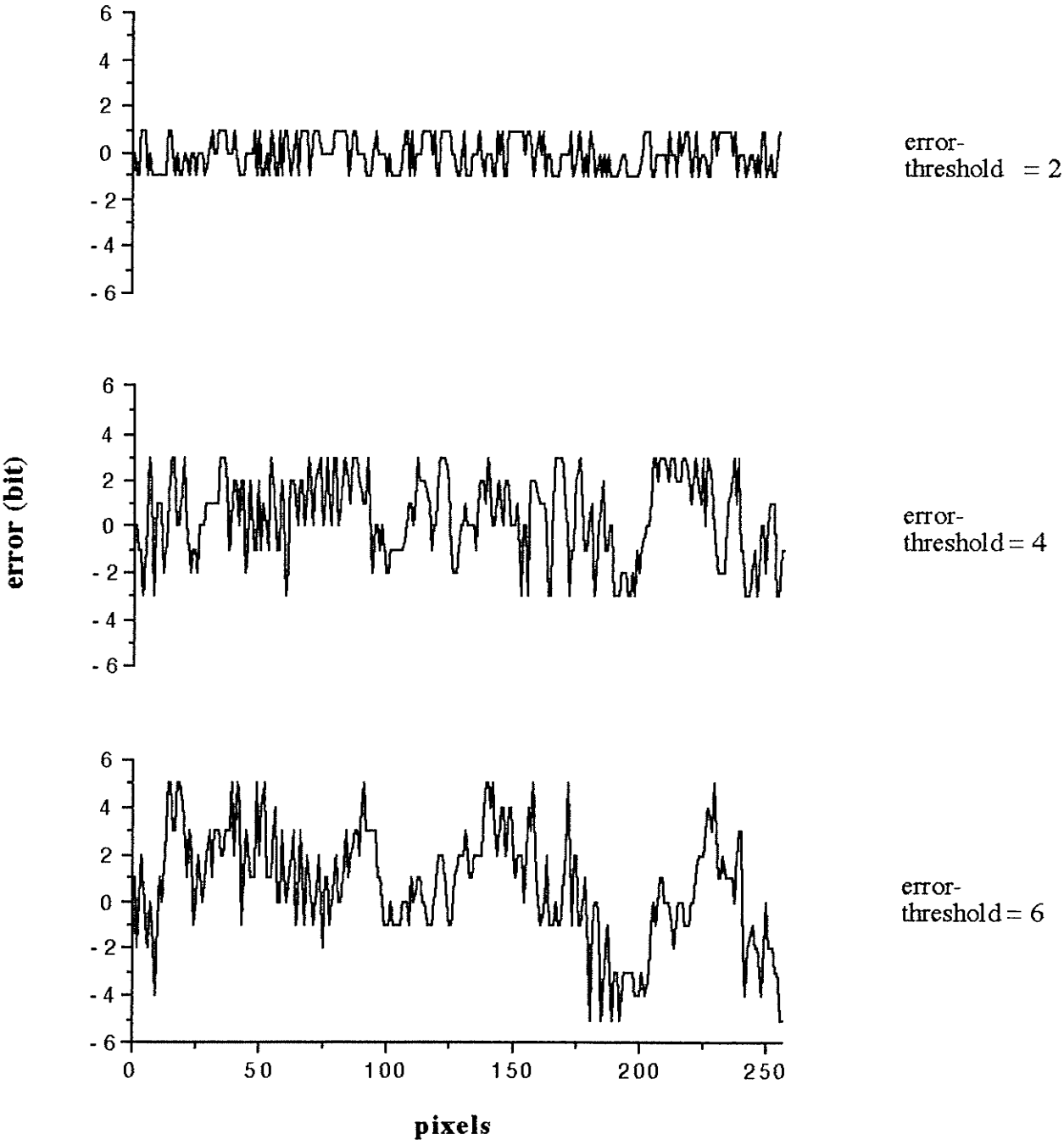

Figure 4.10 Error exists along a scan row with error-threshold equals 2, 4, and 6. These error plots show that the error are bounded by the corresponding error-threshold.

images acquired with error-threshold equals 2 and 6 respectively. The cross-sections show that detail are lost as the error-threshold increases. Also, the cross sections reveal that the error-threshold defines the smallest size of a feature that can be captured.

Figure 4.11(b) and 4.12(b) show the number of Z-feedbacks carried out at each pixel point. However, there are zero Z-feedback at some pixels. For the scanner that is implemented, instead of jumping the tip from one image point to the next, feedback points are placed between two image points in order to avoid the oscillation of the piezoelectric due to the large voltage step. The difference between the feedback and image point is that the feedback output at the feedback point is not recorded. Therefore, some pixels may not need any Z-feedback to adjust the sample height if the height difference between the pixel and the previous feedback point is smaller than the error-threshold. Nevertheless, figure 4.11(b) and 4.12(b) attempt to show the relationship between the number of Z-feedbacks and the features. These two plots show that more Z-feedbacks are carried out where the features have a steep slope.

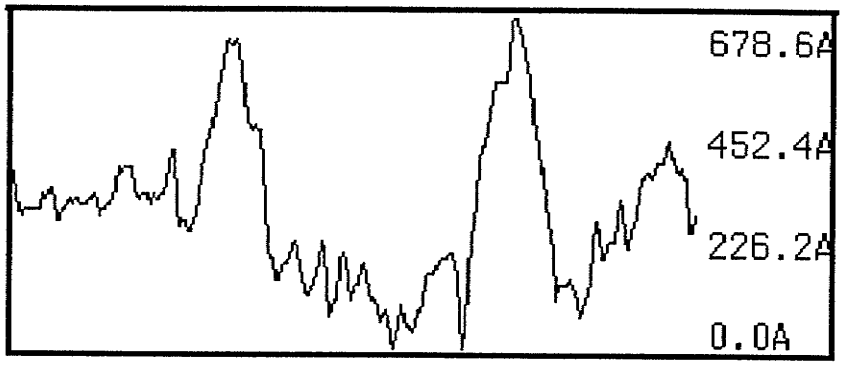

Figure 4.11 (a) The cross section of the one line scan of the grating sample with the error-threshold equal to 2-bits. The scan size is 1 μm

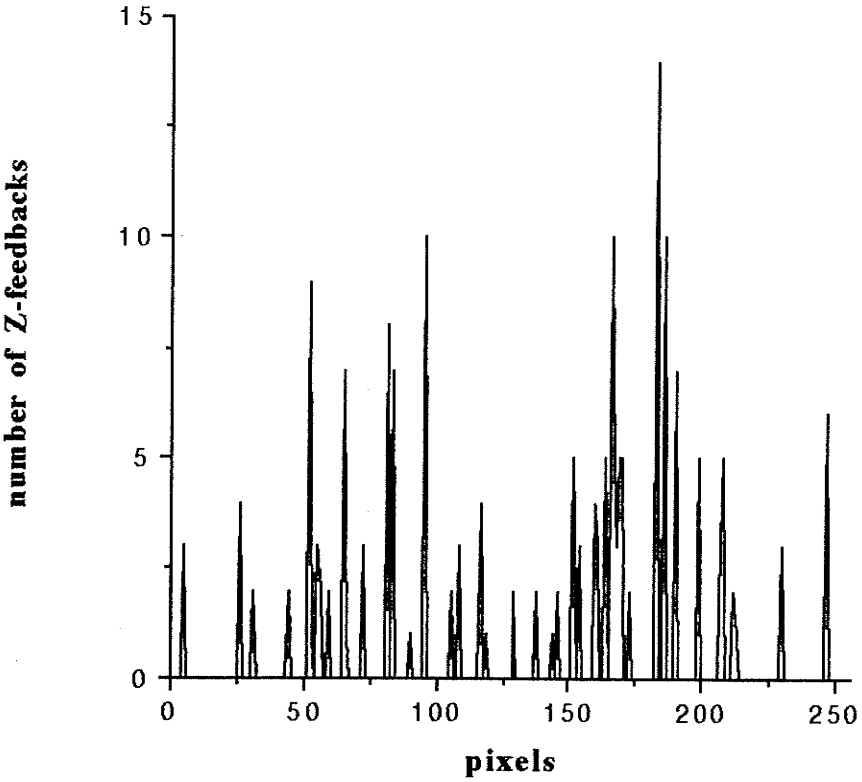

Figure 4.11 (b) The number of Z-feedbacks carried out at each pixel along a scan row with error-threshold equal to 2.

Figure 4.12 (a) The cross section of the one line scan of the grating sample with error-threshold equal to 6 bits. The scan size is 1 μm.

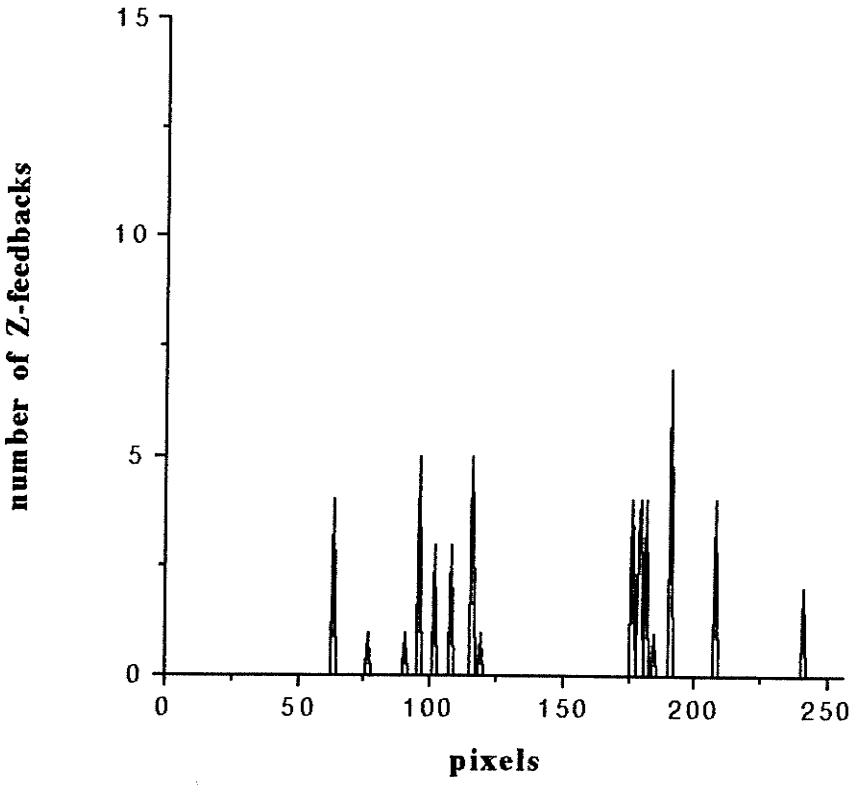

Figure 4.12 (b) The number of Z-feedbacks carried out at each pixel along a scan row with error-threshold equal to 6.

4.8 Discussion on the Performance of the Adaptive Scanner

From the experimental results shown in the previous section, the two scanners have a significant difference in the error profile of the image acquired. The accuracy of an image acquired can be defined by the error at each pixel and the error depends on the scan time. This section attempts to compare the performance of the two scanners by illustrating the relationship between the RMS (root-mean-square) error and the maximum error of the images versus scan time. However, there is a difficulty in comparing these two scanners since they have different independent variables and their dependent variables have different relationships with their independent variables. The independent variable of the constant scanner is the scan speed while the dependent variables are the scan time, maximum error, and the RMS error, while the scan time equals the inverse of the corresponding scan speed. On the other hand, the independent variable of the adaptive scanner is the error-threshold and the dependent variables are the scan time, maximum error, and the RMS error, in which the maximum error equals the corresponding error-threshold minus one. In order to put the data together and compare the performance of these two scanners, the discussion in this section considers an AFM operator who wants to acquire an image of the diffraction grating, the test sample that was used in the experiments discussed in the previous section, with a specific maximum error and RMS error. The discussion is based on the scan time needed to achieve the specified requirements.

Figure 4.13 shows the relationship between the maximum error within the entire image versus the scan time. The graph shows that the images taken by the constant speed scanner have larger maximum error and longer scan time than the images taken by the adaptive scanner. For example, the constant speed scanning requires 256 seconds (which corresponds to 1 Hz scan speed) to acquire an image with maximum error of 5 bits, but the

adaptive scanner took only 50 seconds to acquire the image with the same maximum error. If further less maximum error is required, the adaptive scanner requires even shorter scan time than the constant speed scanner.

Figure 4.14 shows the relationship between the RMS error of the entire image and the scan time of the images taken by both scanners. The graph shows that the scan time of the adaptive scanning is about 50% less than that of the constant speed scanning for small RMS error. As the RMS error increases, the difference in scan time required decreases. Recall the descriptions mentioned in previous section; the major difference between the two scanners is in capturing features with a steep slope. As illustrated by the error-profile in figure 4.9 and 4.10, the constant speed scanner lacks the ability to capture features with steep slopes accurately. The pixels would have large error when steeply sloped features are encountered, as compared to the adaptive scanner. Therefore, the constant speed scanning has to scan at very slow speed in order to acquire an image with small RMS error. On the other hand, although the adaptive scanner spends more Z-feedbacks on capturing the steeply sloped feature (as shown in Figure 11(b) and 12(b)) in order to keep the error within the specified error margin, but it saves Z-feedback when scanning the smooth region. Therefore, the total scan time of the adaptive scanning is faster than that of the constant speed scanning for the small RMS error. However, for larger RMS error, both scanners lack ability in tracking the surface in which case both of them reach the same the scan speed.

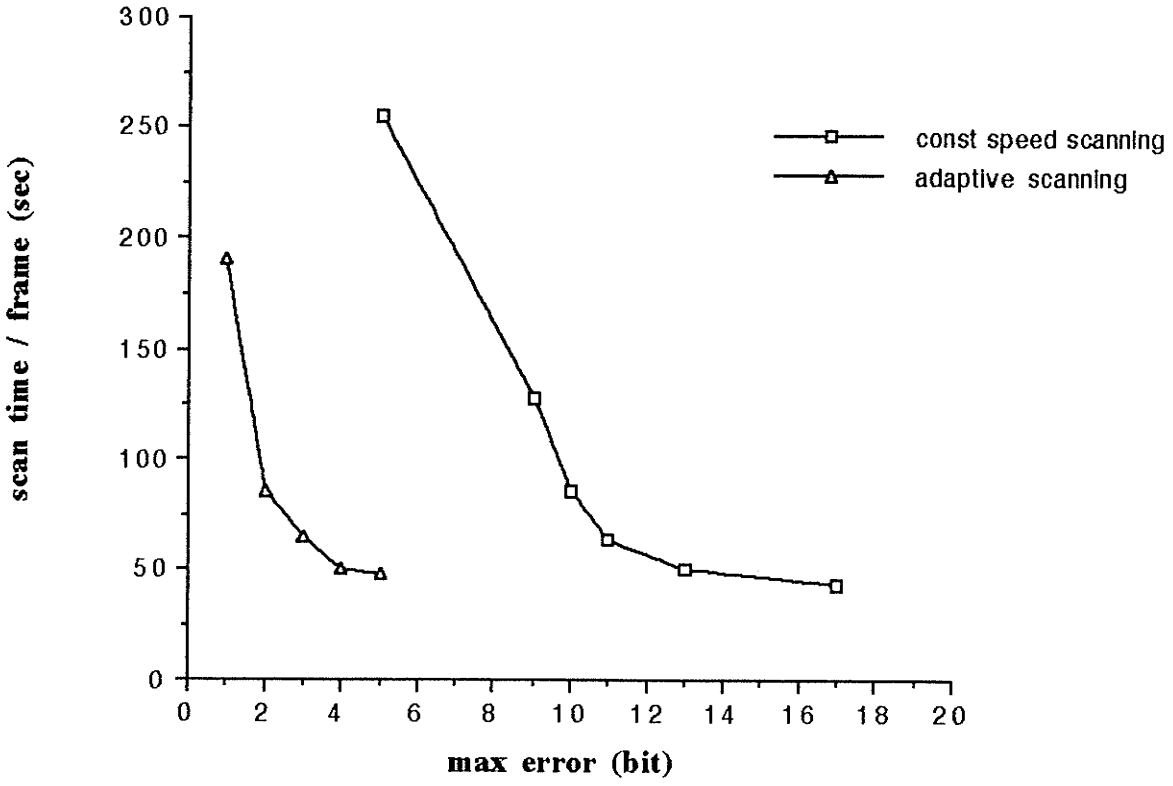

Figure 4.13 The scan time versus maximum error of the images taken by the constant speed scanner and adaptive scanner.

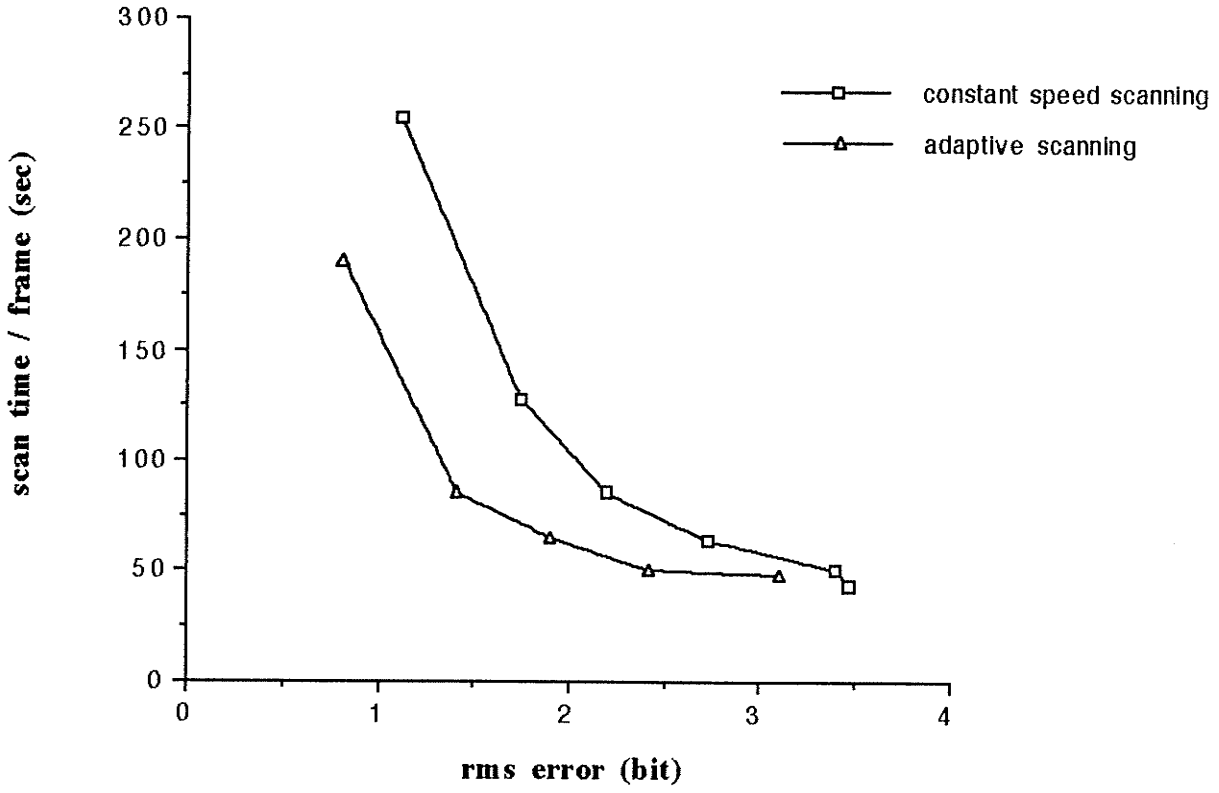

Figure 4.14 The scan time versus RMS error of the images taken by the constant speed scanner and the adaptive scanner.

4.9 Chapter Summary

The combination of the genetic search algorithm and the adaptive scanner makes up the automated AFM scanner. When the AFM operator wants to acquire images of a sample, he/she only has to input the scan size and the value of the error-threshold. The operator does not even have to understand the meaning of the error-threshold, but only know that smaller error-threshold gives more accurate images with longer scan time. As the operator executes the program, the scanner will search for the optimal gains using the genetic algorithm and then carries out scanning afterward with the adaptive scanner.

However, these two components can be handled separately. The system can be configured by using the adaptive scanner only. The adaptive scanner does not care about the value of the proportional gain and the integral gain selected as long as the system is stable, the adaptive scanner can acquire an image with finite known error margin. The effect of the value of the feedback gains is only the scan time needed. If the feedback gains are less than the optimal gains, then the system requires longer scan time to acquire images using the same error-threshold. In addition, the concept of the error-threshold can be further utilized to develop other scan algorithms besides the adaptive scanner and these alternative scanners will be discussed in the next chapter.

Reference

[1] K. L. Westra "Atomic force microscope on thin films," *PhD thesis*, University of Manitoba, 1994.

[2] Z. Michalewicz, J. B. Krawczyk, M. Kazemi, C. Z. Janikow "Genetic Algorithms and Optimal Control Problems," IEEE Proc. 29th. Conf. on Decision and Control, 1664 (1990).

[3] D. R. Baselt, S. M. Clark, M. G. Youngquist , C. F. Spence and J. D. Baldeschwieler "Digital signal processor control of scanned probe microscopes," *Rev, Sci. Instrum.* **64**, 1874 (1993).

[4] J.H. Holland *Adaptation in Natural and Artificial Systems*, Ann Arbor, MI: University of Michigan, 1972.

[5] K. Shahookar and P. Mazumder "A genetic approach to standard cell placement using meta-genetic parameter optimization," *IEEE Trans. Computer-Aided Design*, vol **9**, no. 5, 500 (1990).

[6] J. J. Grefenstette, R. Gopal, B. Rosmaita, and D. Van Gucht "Genetic algorithms for the traveling salesman problem," *Proc. Int. Conf. on Genetic Algorithms and Their Applications*, pp. 160-168, 1985.

[7] J. W. Eheart, S. E. Cieniawski, and S. Ranjithan "Genetic-algorithm-based design of groundwater quality monitoring system," *WRC Research Report* no. 218, University of Illinois, (1993).

[8] J.J. Grefenstette "Optimization of control parameters for genetic algorithms," *IEEE Trans. Systems, Man, Cybernet..*, vol. **SMC-16** (1), pp. 122 (1986).

[9] P. Heuell, M.A. Kulakov, and B. Bullemer "An adaptive scan generator for a scanning tunneling microscope," *Rev. Sci. Instrum.* **65** (1), pp. 89 1994.

[10] R. Piner and R. Reifenberger "Computer control of the tunnel width for the scanning tunneling microscope," *Rev. Sci. Instrum.* **60** (10), Oct. 1989.

CHAPTER 5

AFM CONTROL SYSTEM DESIGN AND ALTERNATIVE AFM SCAN ALGORITHMS

The feedback control system and the scan algorithm are the two factors that affect the performance of a digital AFM system. As discussed in the previous chapters, the feedback control system determines the rate of the system response to the input in which a fast and stable system response with zero steady state error is defined as the optimal AFM system performance. On the other hand, the scan algorithm controls the movement of the tip from one pixel to the other and, in turn, controls the length of time that the feedback system responds to the input. Unlike the typical analog AFM system, the scanner in a digital system can be modified to aid the feedback control system and improve the overall system performance. The automated scanner discussed in Chapter 4 provides an example to show the relationship between the feedback control system and the scanner, and their effects on the performance of the AFM system.

5.1 AFM Control System Design

The AFM works like a record player in which a sharp tip is pressed onto the sample surface and scanned over it. During scanning, the tip will follow the surface contour due to the force between the tip and the sample surface. The vertical displacement of the tip is captured and subtracted from the set-point value and the result is called the error signal. By adjusting the sample height accordingly, the force between the tip and the sample surface

can be kept constant and the error signal will remain zero during scanning. In other words, the signal used to adjust the sample height follows the surface contour and therefore, can be used as the image data for composing the topographic image of the surface. The function of the feedback system is to keep the error signal at zero by adjusting the sample height accordingly so that the feedback output can follow the surface contour accurately. If the response of the feedback system is fast enough, the feedback output can adjust the sample height instantaneously when the error signal varies. Then, the tip will be virtually motionless in the vertical direction and the error signal always equals zero.

As discussed in the Chapter 3, the AFM requires a closed loop feedback control system to acquire topographic image of sample because of the nonlinearity behavior of the bicell detector. However, the closed loop AFM system with the piezo alone has a finite steady state error at the output and the system is highly underdamped with the settling time at about 2 msec. A compensator is required to compensate the drawbacks of the piezo and improve the performance of the AFM system. There are two major specifications for a compensated AFM system:

1) zero steady state error - the system output must have zero steady state error for the step input function so that the system can track the surface accurately;

2) fast and stable system response - stability is actually a primary specification in all system design and the AFM system is no exception. A fast and stable system response can reduce the scan time of the AFM in acquiring an image.

Bound by these two specifications, the following discussion attempts to design a compensator that brings the system into better performance.

5.1.1 Frequency Response Design Technique

Frequency response design technique is a classical control system design method

that has been widely used for years [1]. Using this technique, the design is carried out on the Bode diagram of the open loop transfer function of system. The designer varies the gain, phase, and the cut off frequency of the system curves on the Bode diagram to meet the design specifications such as gain margins, phase margins, bandwidth, and steady state accuracy. After a satisfaction design is achieved, the transfer function of the compensated system is derived from the curves on the Bode plot. The transfer function of the compensator can be obtained by extracting the transfer function of the plant from that of the compensated system.

There are different compensation methods available such as series compensation, minor loop compensation and feedforward compensation [1,2]. However, because of the hardware limitation of the AFM system, only the series compensation is appropriate for the AFM application. Referring to the physical hardware and the block diagram of the AFM system in figure 3.5 in Chapter 3, the input and output of the system are the feature height and the output of the compensator respectively. The piezo, which is the plant of the system, is placed in the feedback path. The output of the piezo is the motion of the piezo in vertical direction and this piezo movement cannot be extracted directly as an input data for the compensator. In addition, the system input, which is the feature height, is unknown prior to scanning and is also not accessible for compensation calculation. The only known parameter is the difference between the feature height and the piezo output and this difference is detected by the bicell detector. Therefore, the AFM acts like a black box in which the bicell detector signal is the only information available to the compensator and the only control signal sent to the AFM is the input voltage to the piezo. This constraint makes the series compensation the only compensation method that is appropriate to the AFM application.

Phase-lag and phase-lead compensation are two common series controllers that are available in frequency response design and these two compensation methods share the

same form of transfer function. The first order transfer function of the compensator is shown in equation (5.1) where ω_p and ω_o are the pole and zero location of the compensator in ω plane respectively, and K_d is the gain of the compensator. If $\omega_o > \omega_p$, the compensation is called phase lag. On the other hand, if $\omega_o < \omega_p$, the compensation is called phase lead.

$$D(j\omega) = K_d \frac{1 + j\frac{\omega}{\omega_o}}{1 + j\frac{\omega}{\omega_p}} \qquad (5.1)$$

The design of the AFM control system using these two compensation methods will be discussed in the following paragraphs.

a) phase-lag compensation

The function of the phase-lag compensator is to increases the loop gain of the system at lower frequency and/or decreases the gain at higher frequency. Figure 5.1 (a) shows the frequency characteristic of the phase-lag compensator. The gain of the phase-lag compensator below the frequency, ω_p, is unity and above the frequency, ω_o, equals $20 \log \frac{\omega_p}{\omega_o}$ if K_d equal to one. The magnitude of the maximum phase lag, Φ_m, of the compensator equals $-\sin^{-1} \frac{1 - \frac{\omega_p}{\omega_o}}{1 + \frac{\omega_p}{\omega_o}}$ and this maximum phase lag occurs at frequency, $\omega_m = \sqrt{\omega_p \omega_o}$. Cascading the phase-lag compensator with the plant, the system gain above ω_o will be reduced by $20 \log \frac{\omega_p}{\omega_o}$ dB because of the negative slope between ω_p and ω_o in the compensator. Varying the open loop gain of the system shifts the gain curve up and down in which can increase the gain at low frequency range or further decreases the gain at high frequency range respectively. On the other hand, although the system bandwidth will usually be reduced because of the reduction in gain at higher frequency, the bandwidth can be increased by raising the open loop gain of the system. The designer chooses the proper

value for ω_p, ω_o, and K_d so that the compensated system meet the design specifications.

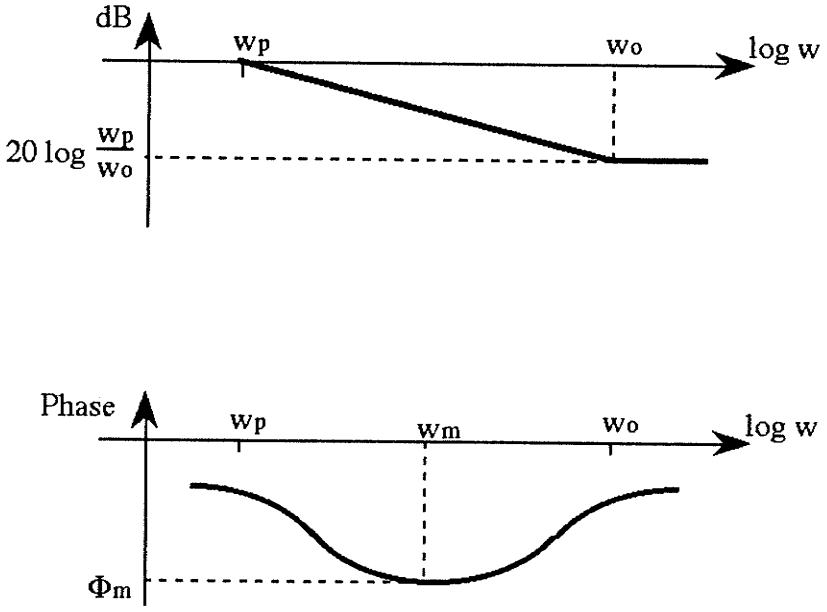

Figure 5.1 (a) Frequency characteristic of the phase-lag compensator

As mentioned in the previous paragraphs, the AFM system requires zero steady state error at the output so that the system can track the surface accurately. Equation (5.2) repeats the transfer function of the piezo, G'(s), shown in Chapter 3 where K_{dc} is the DC gain of the piezo. G'(s) is a type zero system where the function has no pole at the origin of the complex s-plane. Therefore, the closed loop system would always have a finite steady state positional error and this error is proportional to $\frac{1}{1+K}$ where K is the open loop gain of the system.

$$G'(s) = K_{dc}\, G(s)$$

$$\text{where } G(s) = \frac{\omega_n^2}{s^2 + \frac{\omega_n}{Q}s + \omega_n^2} \qquad (5.2)$$

Cascading the piezo with a phase-lag compensator, the compensated system can have small steady state error by moving the pole, ω_p, of the phase-lag compensator to the left (reducing the value of the ω_p) since the system will have higher gain at the low frequency range by increasing the open loop gain. Setting ω_p equals zero, the compensated system will have zero steady state error for the step input function and the phase-lag compensator becomes a PI compensator.

The PI compensator is a special type of phase-lag compensator with transfer function as follow

$$D(j\omega) = K_d \frac{1 + j\frac{\omega}{\omega_o}}{j\omega}$$

The PI compensator is a combination of a proportional controller and an integrator and these two controllers are placed in parallel in the compensator. The proportional controller varies the loop gain of the system by multiplying the error with a constant called proportional gain. On the other hand, the integrator continuously adds the error signal and the sum is multiplied by a constant called integral gain in order to increase the integrating effect. The result of these controllers are added up and sent to the piezo. Since the integrating effect stops only when the error is zero, the integrator is able to achieve zero steady state error at the output with step input function. Equation (5.2) can be rewritten using the z transformation and is shown as

$$D(z) = (K_p + K_I T)\frac{\left(z - \frac{K_p}{K_p + K_I T}\right)}{(z - 1)} \qquad (5.3)$$

The zero position and the gain depend on the proportional gain and integral gain. The detail

of the PI compensated AFM system has been discussed in Chapter 3. Using genetic algorithms to search for the optimal value of these two gains, the proportional gain and the integral gain are 0.8 and 3.4 respectively as obtained from the experimental results presented in Chapter 4.

b) phase-lead compensation

The characteristic of the phase-lead compensator is opposite to that of the phase-lag compensator where the phase-lead compensator decreases the gain at lower frequency and/or increases the gain at higher frequency. Figure 5.1 (b) shows the frequency characteristic of the phase-lead compensator. The significance of the phase-lead compensator is that the positive slope of the gain between ω_o and ω_p can increase the system bandwidth and, in turn, increases the rate of system response. In addition, the positive phase within the effective frequency range increases the stability of the system.

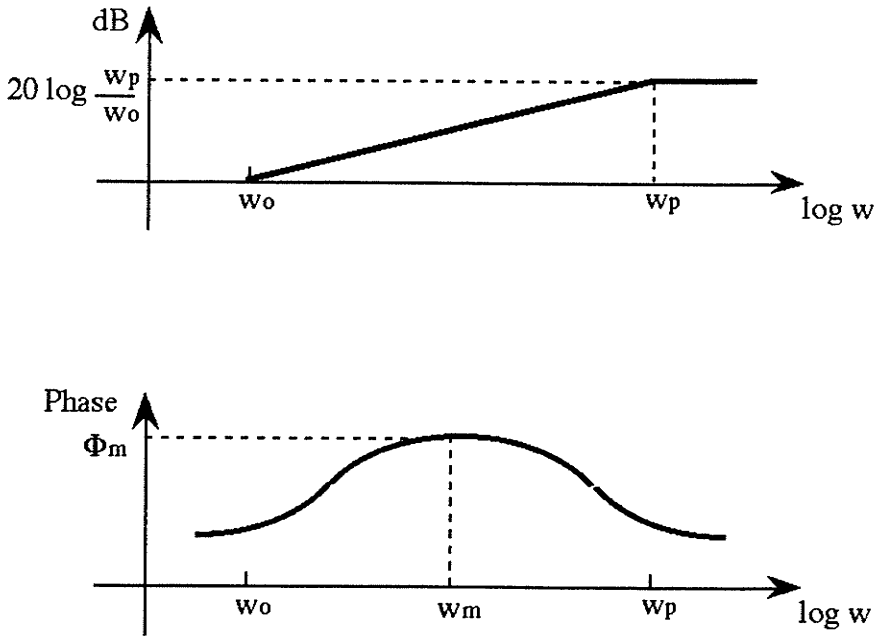

Figure 5.1 (b) Frequency characteristic of the phase-lead compensator

From the discussion in the previous paragraphs, the compensator must include the integrator so that the compensated system will have zero steady state error at the output. Figure 5.2 shows the Bode plot of the open loop system that is compensated by the integrator only. The function of the phase-lead compensator is to increase the system bandwidth by adding a positive slope gain to the system. However, cascading the phase-lead compensator with the integral compensated system, the phase-lead compensator not only increases the system bandwidth, but also raises the resonance peak of the piezo. Since the resonance peak of the piezo must be kept below 0 dB or the system will become unstable, the system may have less gain at the low frequency range than that of the integral compensated system in order to keep the system in stability. Since a proper choice of ω_p and ω_o of the phase-lead compensator is largely trial and error, a proportional-integral-derivative (PID) compensator is used instead which is a special type of a lead-lag compensator.

A lead-lag compensator is a cascade of a phase-lag and a phase-lag compensator. Since an integrator is essential in compensating the piezo, cascading a phase-lead compensator with a PI compensator is equivalent to cascading a phase-lag and a phase-lead compensator. The PID compensator is a cascade of a phase-lag compensator with ω_p equal to zero and a phase-lead compensator with ω_o equal to infinity. The following sections describe the PID compensator and the genetic algorithm is used to tune the PID compensator to find the optimal feedback gains.

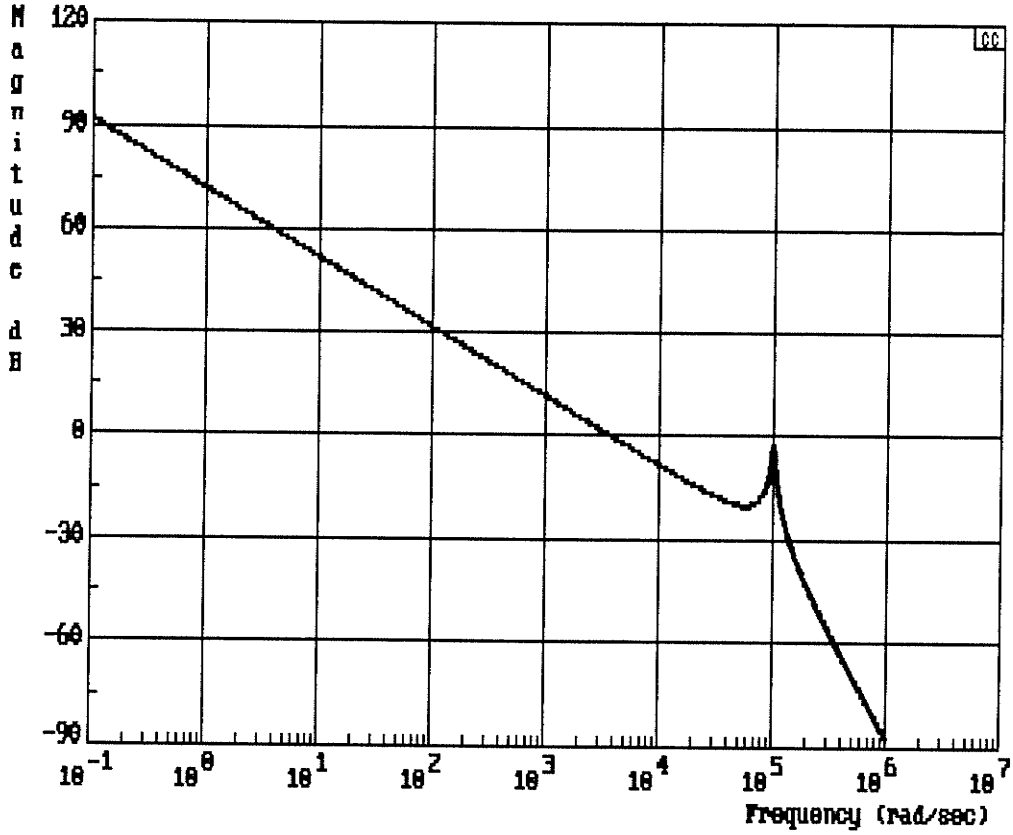

Figure 5.2 Frequency response of the open-loop AFM system with an integral controller

c) PID compensator

The PID compensator is a combination of a proportional controller, an integrator and a derivative controller and these three controllers are placed in parallel. The error signal is fed into these three controllers simultaneously and their outputs are added together and the sum is sent to the plant. The proportional controller and the integrator have been discussed in the Chapter 3 and the previous paragraphs and will not be repeated here. The derivative controller calculates the derivative of the input signal and multiplies the result by a constant called the derivative gain. The transfer function of the derivative in z domain is shown as follow

$$D(z) = \frac{K_D(z - 1)}{z} \tag{5.4}$$

Equation (5.4) calculates the difference between the current and previous input signal and multiplies the result by the derivative gain, K_D. The overall transfer function of the PID compensator is

$$D(z) = K_P + K_I \frac{z}{z - 1} + K_D \frac{(z - 1)}{z} \tag{5.5}$$

Using the same procedure mentioned in Chapter 4, the genetic algorithm is used to find the optimal gains for the PID compensator. Figure 5.3 shows the results of the searching in forty generations. The initial value of all the three gains are 0.1 and the final value of the proportional gain, integral gain, and derivative gain after forty generations are about 0.3, 4, and 0.5 respectively.

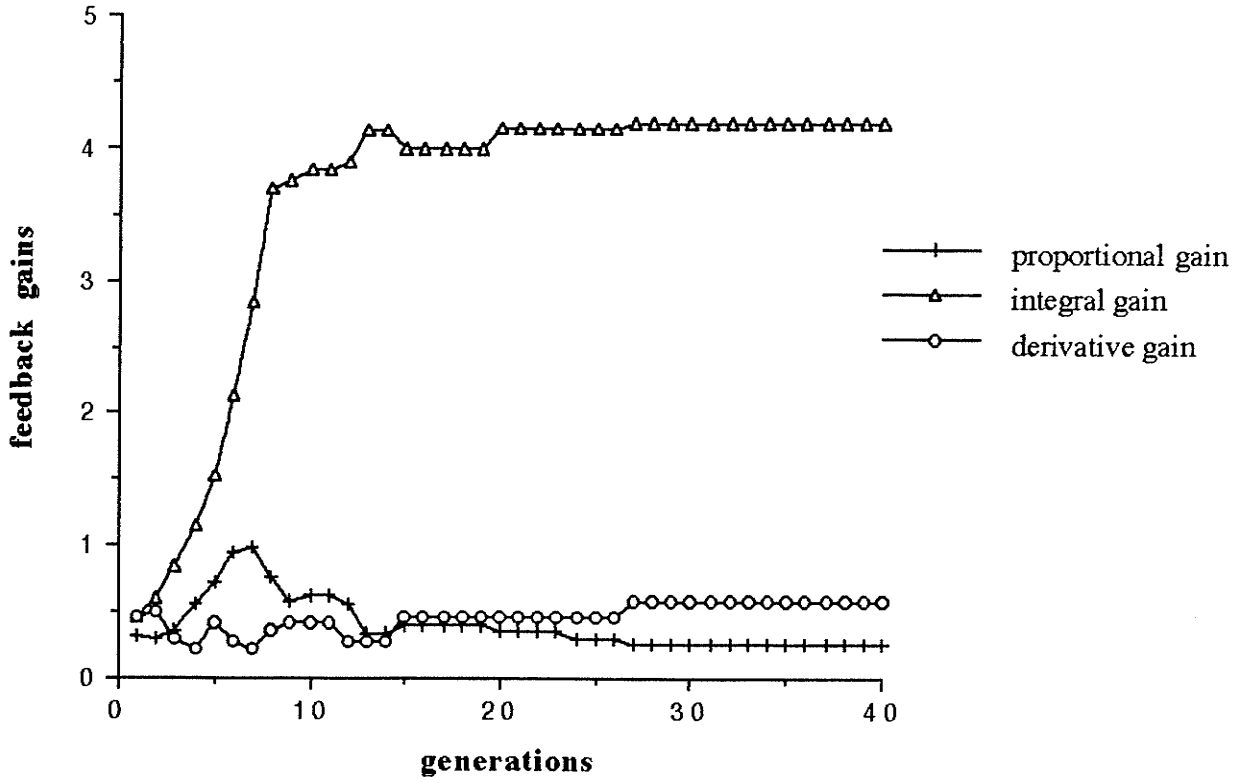

Figure 5.3 Using genetic algorithm to tune the PID compensator, this graph shows the variations of the proportional, integral, and derivative gains within the first 40 generations.

5.1.2 Root-Locus Technique

Designing on the root-locus diagram is another classical control system design method besides the frequency-response technique. The design is carried out on the root-locus diagram in which the roots of the characteristic equation of the plant can be shifted by adding poles and zeros via the compensator in order to improve the system performance. 'Pole-zero cancellation' is a common design method of the root-locus technique, in which new zeros are placed on or near the undesirable poles in order to reduce their effect on the system. The advantage of this technique is that the design can be carried out directly on the root-locus diagram of the z-plane. The transfer function of the compensator can be derived from the root-locus diagram without going through the z-transformation from continuous

system as in the frequency-response technique. In addition, the behavior of the compensated system can be estimated from the root-locus diagram.

Figure 5.4 (a) shows the root locus diagram of the characteristic equation of the AFM system in z-domain. The complex poles (-0.92 ± j0.132) from the piezo introduce undesirable effects to the system since they are close to the circumference of the unit circle. The effect of these two poles can be canceled by placing zeros on or near to them. A new pole is placed at z equal to one so that the system can have zero steady state error for a step input. However, the number of zeros must be less than or equal to the number of poles inside the transfer function of the compensator so that the compensator can be realized [3]. Therefore, a new pole must be added so that the compensator has the same number of zeros and poles. The new pole is placed at the origin of the unit circle in order to reduce its effect on the rate of system response. The transfer function of the compensator from the pole-zero cancellation, $D_{PZ}(z)$, is shown in equation (5.3) and figure 5.4 (b) shows the root-locus diagram of the new design.

$$D_{PZ}(z) = \frac{z^2 + 1.84\,z + 0.86}{z\,(z - 1)} \qquad (5.6)$$

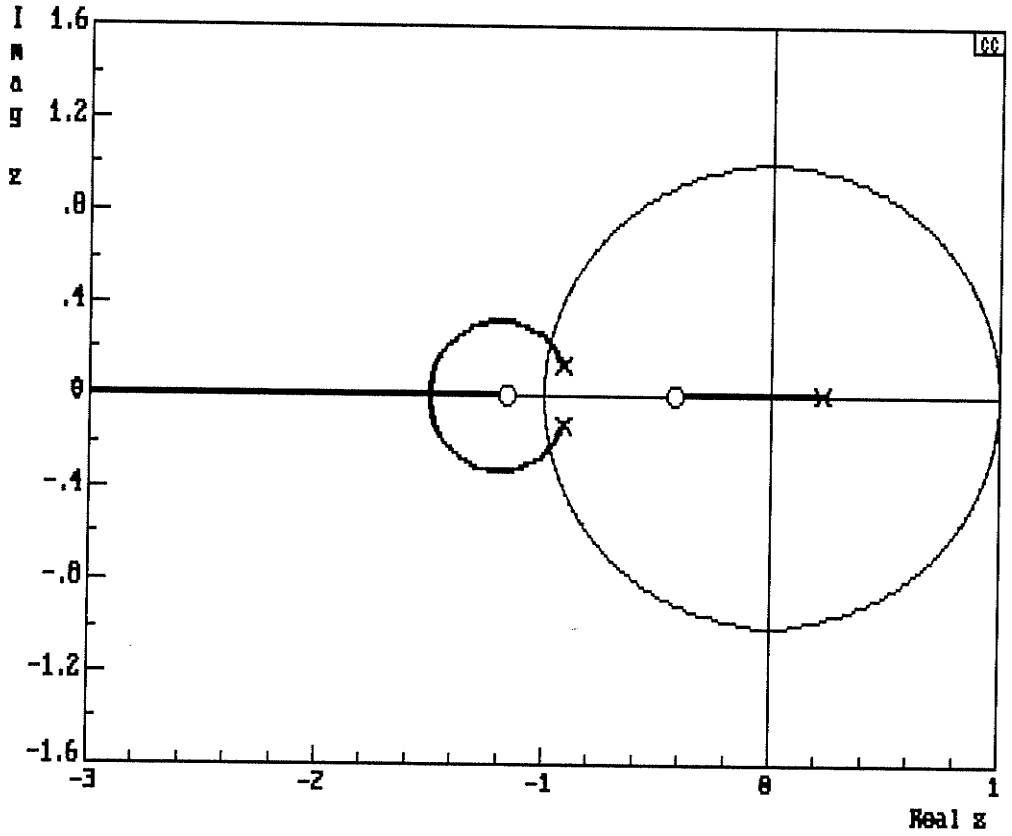

Figure 5.4 (a) Root locus diagram of the characteristic equation of the AFM system.

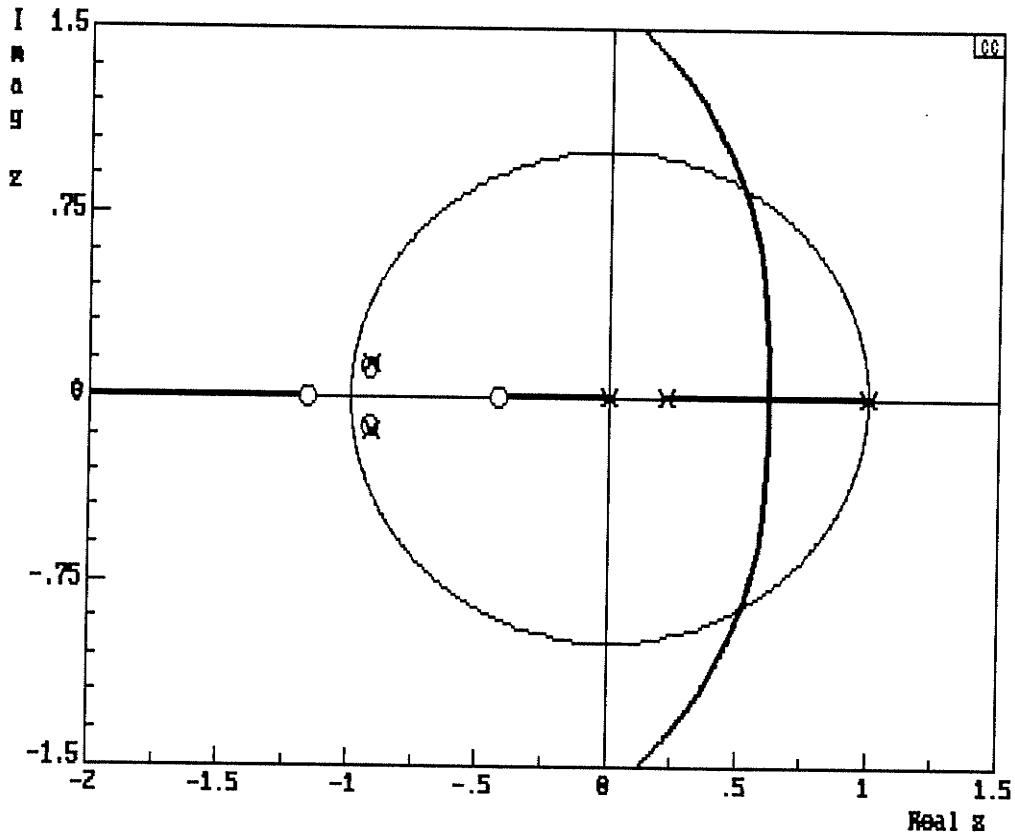

Figure 5.4 (b) Root locus diagram of the characteristic equation of the AFM system that cascades with the compensator $D_{PZ}(z)$ where

$$D_{PZ}(z) = \frac{z^2 + 1.84\,z + 0.86}{z\,(z - 1)}.$$

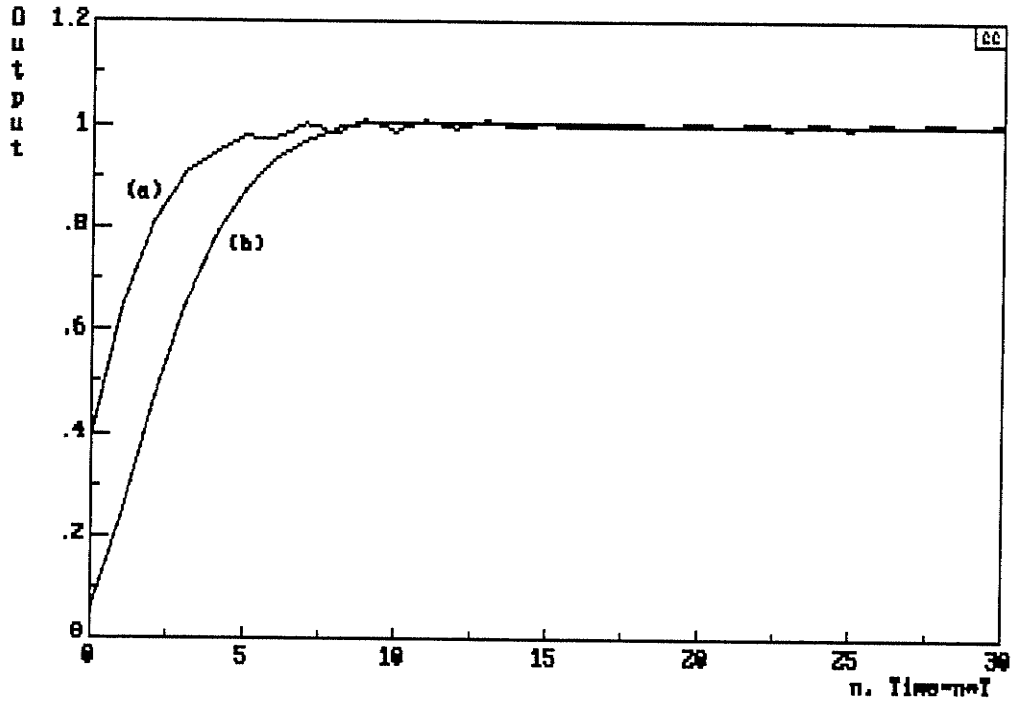

Figure 5.5 (a) Comparison of the simulated step response of (a) the PID compensated system and (b) the pole-zero canceled system. The rise time of curve (a) is two sample cycles faster than that of curve (b). The rise time of curve (a) is 5 sample cycles. One sample cycle equals 30 μsec in our system.

Figure 5.5 (a) compares the simulated step responses of the system using PID compensator and $D_{PZ}(z)$. From the figure, the step response curve from the $D_{PZ}(z)$ compensated system is more steady than that of the PID compensated system. However, the rise time of the PID compensated system is about two sample cycles faster than that of the $D_{PZ}(z)$ compensated system where the rise time of the PID compensated system is about 5 sample cycles. The sampling period in our system is 30 μsec.

Figure 5.5 (b) shows the actual step response curve from the physical hardware system using these two compensators. The experimental results shown in figure 5.5 (b) are very close to the simulation results where the PID compensated system has faster rise time than the $D_{PZ}(z)$ compensated system. The rise time of the PID compensated system and $D_{PZ}(z)$ compensated system are 9 and 10 samples respectively. The rise time of the step responses from the experiments is slower than that from the simulation and this deviation may due to the low-pass filter at the input of the ADC (the anti-aliasing filter).

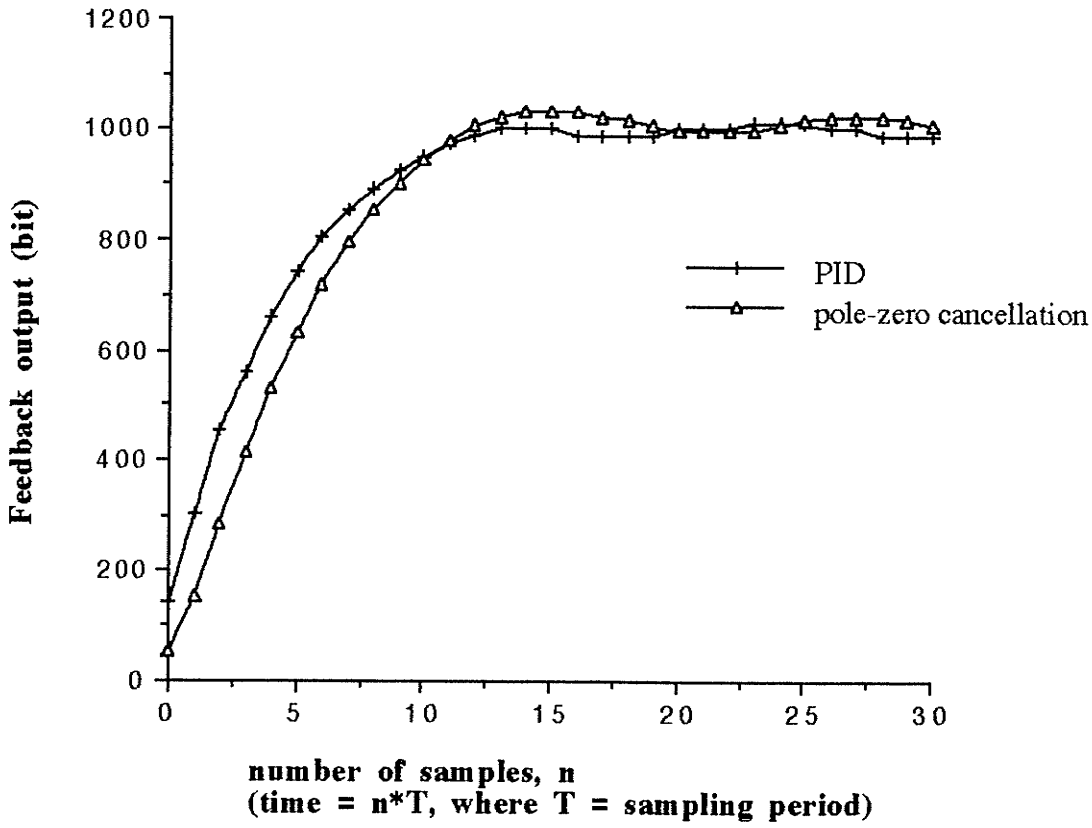

Figure 5.5 (b) Comparison of the actual step response from the physical hardware system of the PID compensated and the pole-zero canceled systems. The sampling period, T, is 30 μsec.

The transfer function of the compensator, $D_{PZ}(z)$, is derived based on the assumption that the resonant frequency of the piezo is 16 kHz and the quality factor is 20 at resonance. However, the resonant frequency and the quality factor of the piezo will vary if the mass of the sample put on top of the piezo is different. Also, these two variables are different for different piezo's. Simulations are done by assuming that the piezo has resonant frequency at 10 kHz and the quality factor is 10. The transfer functions of the new system (cascading the new piezo with the same lowpass filter in its front end) in s-domain and z-domain are shown as follow

$$H(s) = \left(\frac{4*10^9}{s^2 + 6283\,s + 4*10^9}\right)\left(\frac{5000}{s+5000}\right) \qquad (5.7\ a)$$

$$H(z) = \frac{0.5\left(z^2 + 2.28\,z + 0.45\right)}{\left(z^2 + 0.58\,z + 0.83\right)(z - 0.22)} \qquad (5.7\ b)$$

The complex poles of the new system are at ($-0.29 \pm j0.86$) and the root locus diagram of cascading the new system with the compensator $D_{PZ}(z)$ is shown in figure 5.6. Although the zeros of the compensator are not exactly at the same positions of the complex poles of the piezo, the zeros of the compensator are still able to reduce the effect of the complex poles of the piezo. The step response of the new system is exactly the same as the one from the $D_{PZ}(z)$ using the same open loop gain.

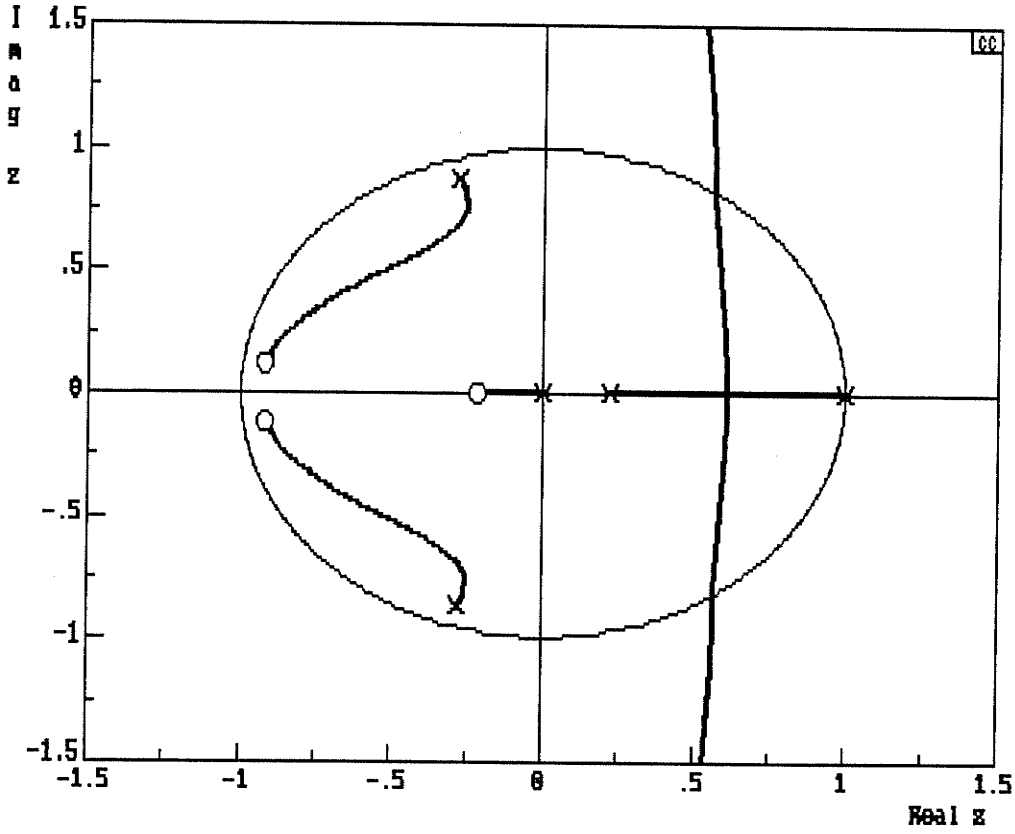

Figure 5.6 Root locus diagram of the compensated system using compensator D_{PZ} (z). where $D_{PZ}(z) = \frac{z^2 + 1.84\,z + 0.86}{z\,(z - 1)}$. The resonant frequency and the quality factor of the piezo are 10 kHz and 10 respectively instead of 16 kHz and 20 in the original design.

The performance of the pole zero cancellation compensator is very close to that of the PID compensator and is able to tolerate the variation of the resonant frequency and the quality factor of the piezo. However, the pole-zero cancellation compensator requires more code in the DSP-program than the PID compensator, which will reduce the sampling frequency of the system. In addition, the compensator derived from the pole-zero cancellation method is system dependent where the design requires knowledge of the system. On the other hand, the PID compensator is a generic compensator to the AFM application and requires little information about the system.

5.2 Alternative AFM Scanners

Inside the digital AFM controller, the movement of the tip along x and y direction is software controlled. The flexibility of software implementation allows the modification of the scan algorithm to go beyond the constant speed scanning as in the analog counterpart. Besides controlling the movement of the tip from one image point to the other along the x-direction, the scan algorithm is also able to help the feedback control system to compensate the sample height. The adaptive scanner inside the automated scanner discussed in Chapter 4 is an example that shows the effect of the scan algorithm on the system performance in reducing the scan time and increasing the accuracy of the image acquired. The idea of the adaptive scanner has been extensively applied to implement other scanners that will be discussed in the following sections.

5.2.1 Predictor Corrector

During scanning, the variation of the error signal reflects the deflection of the cantilever that causes by the height of feature. If the ratio between the error signal and the

vertical displacement of the piezo is known, then the program can adjust the sample height close to the actual feature height in a single subroutine cycle using the product of the error signal and the ratio. The predictor corrector uses this idea to predict and adjust the sample height accordingly whenever there is variation in the error signal. An ideal predictor corrector will calculate the desired sample height and make the height adjustment in single subroutine cycle. The time required to adjust the sample height will depend on the rise time of the Z-piezo with the corresponding lowpass filter if the rise time is longer than the sampling period.

The key factor of this scan algorithm is the value of the ratio used for calculating the desired sample height, and this ratio can be obtained as follows. Before scanning, the program generates a step to the Z-piezo by varying the value send to the DAC, and then records the changes in the error signal. For instance, the program varies the value sent to the DAC by M bits and the corresponding changes in the error signal is N bits as acquired from the ADC. The ratio between the Z-piezo displacement and the error signal is then

$$p = \frac{M}{N} \tag{5.3}$$

The ratio, p, can be named the predictor factor and it relates the Z-piezo displacement to the single bit of variation in the error signal. The advantage of this algorithm is that , for instance, if the predictor factor has 80% correctness, two successive predictions will have 96% correctness and the program takes only two subroutine cycles to adjust the sample height to less than 5% error.

The rise time of the piezo with the lowpass filter inside our digital controller is about 60 μsec and the sampling period is 30 μsec. When the predictor corrector is implemented inside the DSP-program, a repeat loop is needed inside the feedback subroutine in order to wait for the piezo to react to the input voltage. Figure 5.7 (a) and (b) show the step response using the PID compensator and the predictor corrector respectively.

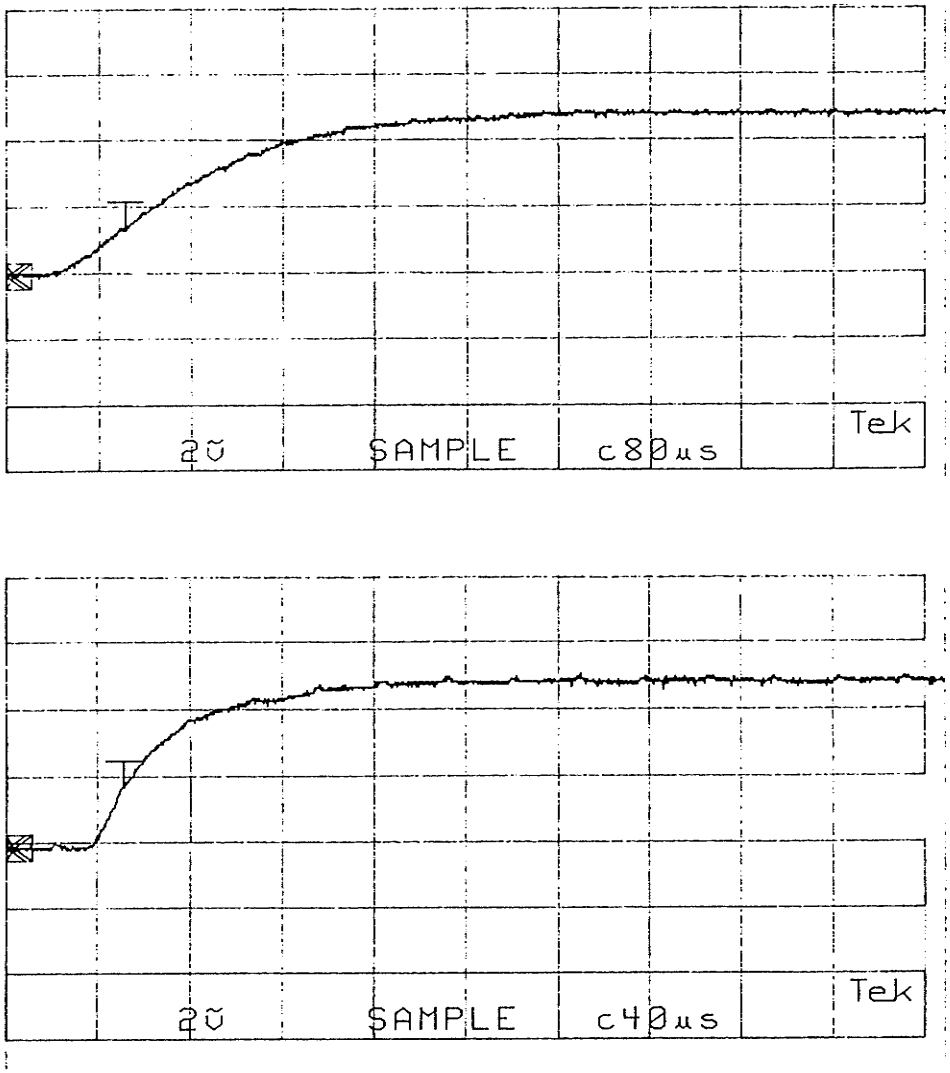

Figure 5.7 The step response curve on the top is from the actual PID compensated AFM system while the one on the bottom is from the AFM system with predictor corrector. The rise times of the step response curves on the top and bottom are about 200 μsec and 100 μsec respectively.

Using the optimal feedback gains, the PID compensator requires about 200 μsec to reach 90% of the steady state output while the predictor corrector needs about 100 μsec to achieve the same level of output. These two graphs show that the predictor corrector is twice as fast as the PID compensator in step response.

However, the predictor corrector suffers high sensitivity to noise. Since the algorithm responds to every bit of variation in the error signal and makes a move at the Z-piezo, even single bit variation in the error signal can have a significant change in the Z-piezo position. There are two solutions to this drawback. One of the solutions is to use predictor corrector to make a first guess when the tip moves to a new image point and then use PID controller to fine tune the sample height. The adjustment of the PID compensator can be terminated by using error-threshold as in the adaptive scanner. Another solution is to restrict the use of the predictor corrector to large error signals only. The predictor corrector is activated when the error signal is above the noise level, otherwise, the PID compensator is used as usual.

5.2.2 Ramp Scanner

Inside the adaptive scanner, the DSP-program will adjust the sample height at both image and feedback points if the error is larger than the error threshold. The main function of the feedback points is to avoid the piezo ringing when the piezo moves from one image point to the other in response to a step voltage to the piezo. Adjusting the sample height at the feedback points helps keeping track of the surface and reduces the height difference between an image point and its previous feedback point. This reduction of height difference can reduces the number of feedback cycles needed at the image points to achieve the error signal that is less than the error threshold. However, for every height adjustment, the program has to acquire the error data from the ADC, and then carry out the feedback calculation and send the feedback output to the piezo if the error is larger than the error threshold. The sample height adjustment at the feedback points may slow down the

scanning. In order to speed up the scanning, an alternative scan algorithm, the ramp scanner is developed which is a modified version of the adaptive scanner. The ramp scanner uses the difference between the sample height of the previous and current image points and makes a guess of the sample height of the next one. Since two image points are used for prediction, this ramp scanner can be named first order ramp scanner.

At every image points, the ramp scanner adjusts the sample height until the error signal is less than the error threshold. Then, the program calculates the difference of the sample height between the current and previous image points. Assuming the next image point has the same height difference from the current one, the program gradually changes the sample height to the predicted value while the piezo is moving toward to the next image point. Figure 5.8 (a) and (b) illustrate the scan algorithm of the ramp scanner in spatial and time domain.

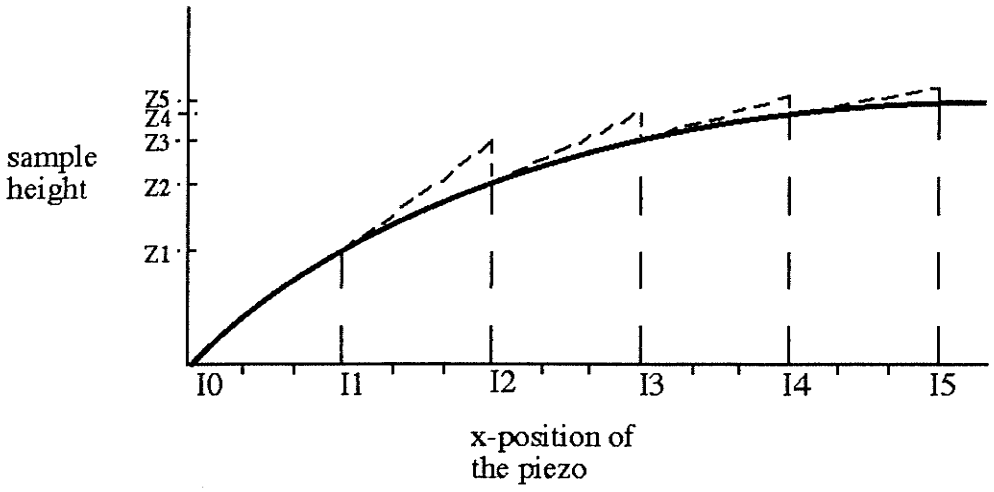

Figure 5.8 (a) This picture shows the movement of the piezo on the sample surface from image point I0 to I5 using the first order ramp scanner. The solid line is assumed to be the sample surface and the dot line is the movement of the sample height.

In figure 5.8 (a), the solid curve and the dotted line represent the sample surface and the movement of the piezo respectively. Image points 'I0' and 'I1' are two initial points. Since the height difference between 'I0' and 'I1' is 'Z1' and the height of the current image point 'I1' is 'Z1' (assuming the height of image point 'I0' is zero), the predicted height of 'I2' is 'Z1 + Z1'. When the piezo moves from 'I1' to 'I2', the program gradually raises the sample height from vertical position 'Z1' to 'Z1 + Z1'.

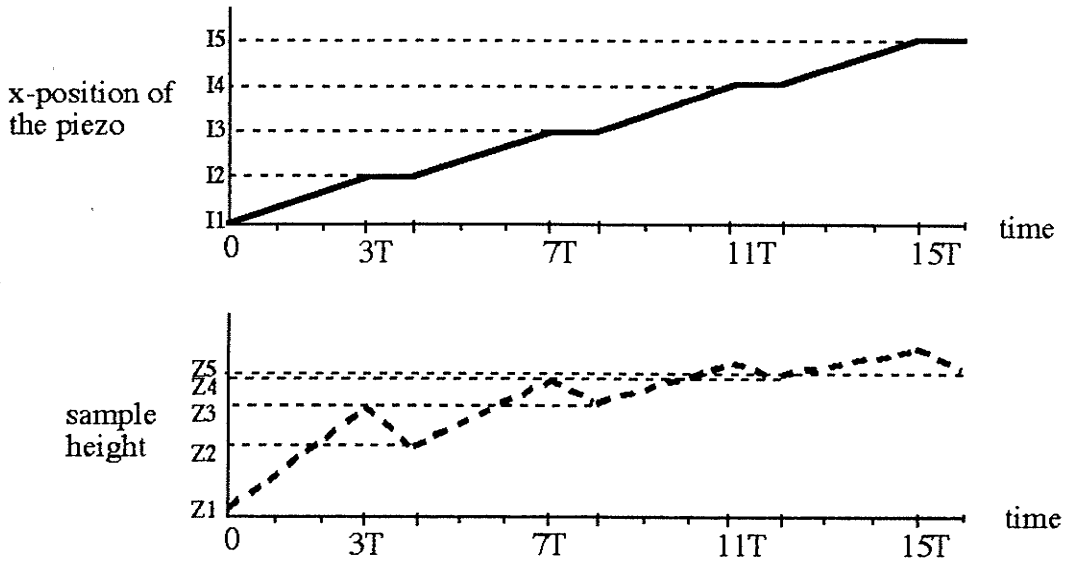

Figure 5.8 (b) These two graphs show the movement of the piezo in x and z direction. When the piezo moves from the current image point to the next one along the x-direction, the system raises the sample height to the predicted value simultaneously. When the system adjusts the sample height at the image point, the piezo stops the x-direction movement.

In this example, there are two feedback points between image points. The system raises the sample height by one third of the height difference, 'Z1', at each feedback point between 'I1' and 'I2'. After the piezo reaches 'I2', the program adjusts the sample height

to position 'Z2', which is the sample surface, within one sample cycle in this case. When the program is updating the sample height at 'I2', the x-position of the piezo stays at the same value until the sample height equals 'Z2' as shown in figure 5.8 (b). Figure 5.8(b) shows that the x-position of the piezo remains constant when the feedback system adjusts the sample height at the image points. Afterward adjusting the sample height at 'I2', the piezo moves to the next image point 'I3' and the predicted height of 'I3' is 'Z2 + (Z2 - Z1)' and the program raises the sample height by one third of the predicted height at each feedback point again. The same process keeps going during scanning.

As mentioned in previous paragraphs, the adaptive scanner will adjust the sample height at both feedback and image points if the error signal is larger than the error threshold. In order words, the DSP-program has to acquire the error signal from the ADC for every feedback cycle. The conversion time of the ADC used in our digital controller is 6 μsec. This extra waiting period slows down the scan time of the adaptive scanner. Figure 5.9 compares the performance of the ramp scanner and the adaptive scanner. The sample used in this experiment is the chemical vapor deposition (CVD) tungsten film and its topography is shown in figure 5.10. The graph in figure 5.9 shows that the ramp scanner requires less scan time than the adaptive scanner using the same error-threshold. In addition, the difference in the scan time for the ramp scanner among different error-thresholds is less than that of the adaptive. Besides saving the conversion time of the ADC, the ramp scanner carries out sample height adjustment only at the image points. Therefore, the variation of the total number of Z-feedbacks for the whole image frame become small for different error-thresholds.

The performance of the ramp scanner very much depends on the roughness of the sample surface. For instance, both the ramp scanner and the adaptive scanner take about the same time to acquire images for the grating sample that is used in the experiments in Chapter 4. Since the surface of the grating sample is generally very smooth, the height

difference between image points is relatively small. The ramp scanner does not have advantage over the adaptive scanner since the prediction of the next image point by the ramp scanner is very small . However, for the rougher CVD tungsten film, the prediction of the ramp scanner helps a lot in adjusting the sample height and saves the total number of Z-feedbacks required.

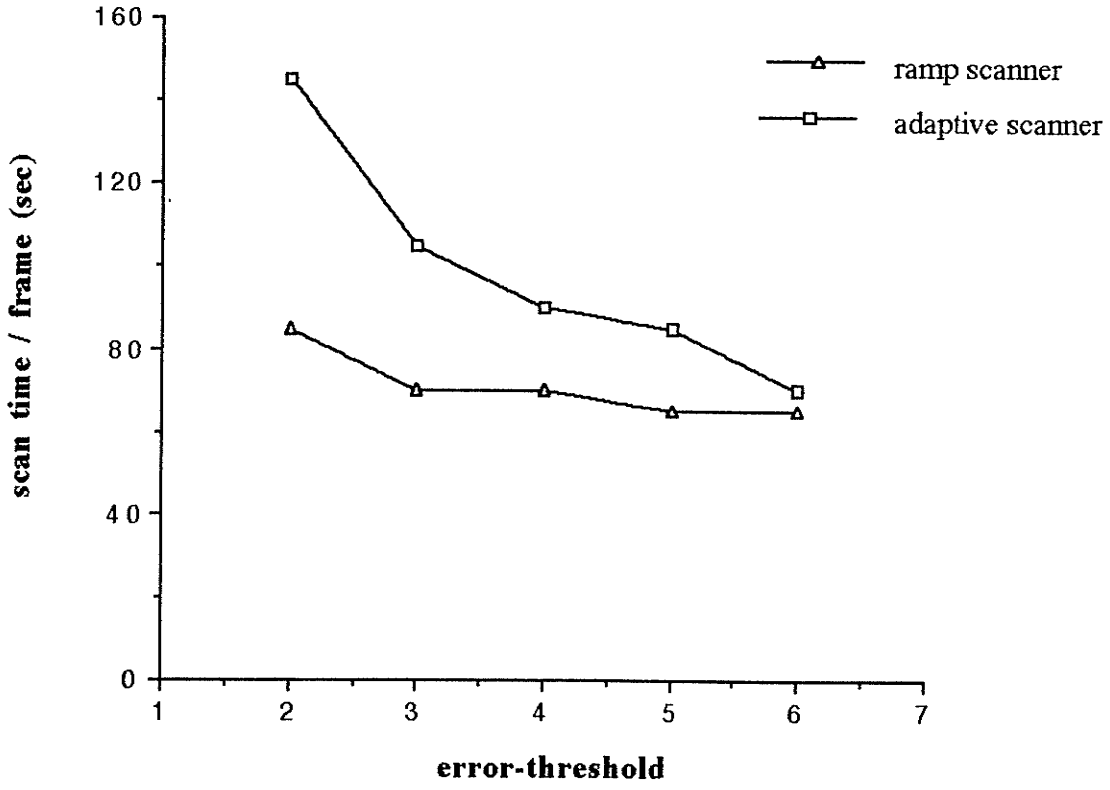

Figure 5.9 Comparsion of the performance between the predictor corrector and the adaptive scanner.

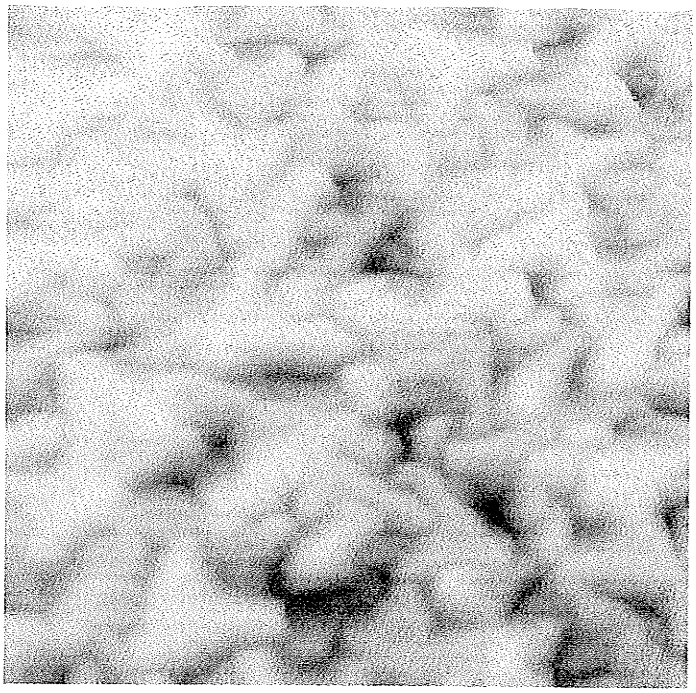

Figure 5.10 Topographic image of the CVD tungsten film with 2 μm by 2μm scan size. This image is taken by a PID compensated AFM system with 2 Hz scan speed.

The first order ramp scanner will have larger deviation in the prediction when the slope of the features between two image points changes rapidly, and more feedback cycles are required to compensate the sample height at the image points. This drawback is illustrated in figure 5.11. There are two samples in figure 5.11, one on the top and the other on the bottom of the page. The surface roughness of the sample on the top is higher than that on the bottom. The first order ramp scanner has a poor prediction on the next image point and the scanner needs more feedback cycles to compensate the sample height. Therefore, when the number of features across a scan row increases, higher order ramp scanner should be used in which more previous images points are taken into account to predict the new one. A similar scanning algorithm has been proposed by T. Wong and et. al. [3] in which each previous image point is assigned a coefficient when adding them together.

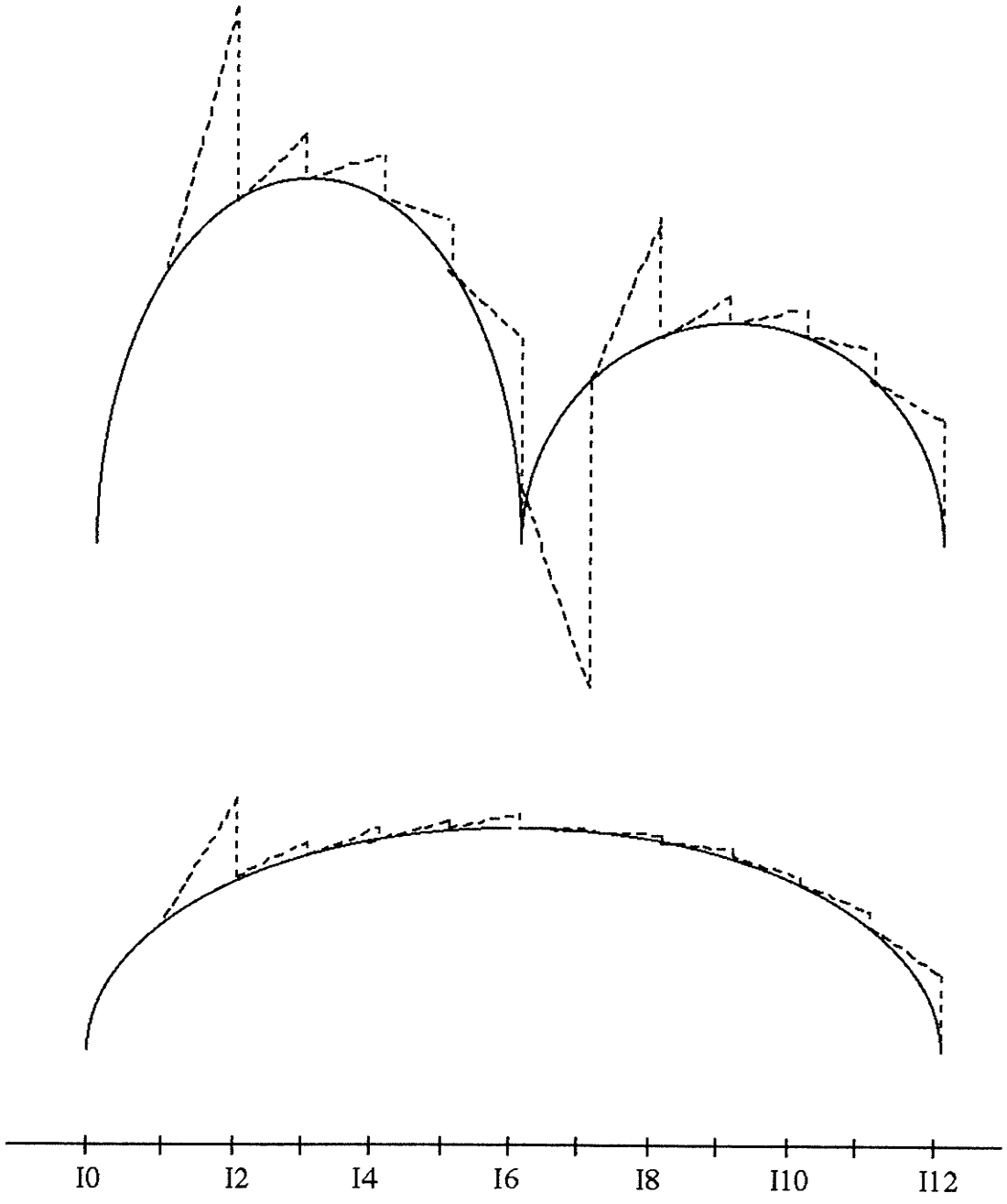

Figure 5.11 The sample on the top has a higher surface roughness than that on the bottom. Using the first order ramp scanner, the scanner has a poor prediction of the next image and takes more feedback cycles to compensate the sample height at the image points.

Reference

[1] G. J. Thaler *Design of Feedback Systems,* Dowden, Hutchingon & Ross, Inc., Penn., pp. 69 - 161, 1973.

[2] J. Schwarzenbach and K. F. Gill *System Modelling and Control,* Edward Arnold, Maryland, pp. 223, 1984.

[3] T. M. H. Wong and M. E. Welland "A digital control system for scanning tunnelling microscopy and atomic force microscopy," *Meas. Sci. Technol.* **4**, pp 270-280, 1993.

CHAPTER 6

RECOMMENDATIONS

The primary objective of this thesis is to search for alternate control system(s) for the atomic force microscope (AFM) besides the common proportional-integral-derivative (PID) controller. Because of the nonlinearity of the bicell detector, the AFM is possible only if feedback control system is used. The function of the feedback control system is to maintain a constant force between the tip and the sample surface by adjusting the vertical position of the sample. The feedback output must have zero steady state error to a step input function in order for the system to track the surface accurately. In addition, a fast and stable system is desired so that the system can scan at a higher rate without losing the tracking ability.

The control system design has been carried out using the frequency response technique and root-locus technique. Since the AFM system with the piezo alone is a type zero system, the system always has a finite steady state error at the output for step input function. Using the frequency response design technique, cascading the system with a phase-lag compensator can increase the gain at lower frequency which reduces the steady state error of the system. By setting the pole of the phase-lag compensator to zero, the compensated AFM system can achieve zero steady state error and such a phase-lag compensator is essentially a proportional-integral (PI) compensator.

Since the integrator inside the PI compensator reduces the system bandwidth, the PI compensated system has very slow system response. Cascading the PI compensated AFM system with a phase-lead compensator, the system will have faster system response since the phase-lead compensator is able to increase the system bandwidth. However, the phase-

lead compensator increases the gain at higher frequency which raises the gain of the resonant peak of the piezo. The resonant peak larger than 0 dB will drive the system unstable. Choosing a proper pole and zero for the phase-lead compensator in order to maintain system stability is largely trial and error. Instead of using a simple phase-lead compensator, proportional-derivative (PD) compensator is used which is a phase-lead compensator with the zero at infinity. Cascading a PI and PD compensator becomes the PID compensator. Using the genetic algorithm to tune the PID compensator, the experimental result shows that the effect of the integrator dominates the effect of the PID to the AFM. The optimal proportional, integral, and derivative gains of our AFM system are 0.3, 4, and 0.5 respectively.

The control system design using frequency response technique results in the PID compensator. An alternate control algorithm has been developed using the root-locus technique. Assuming the resonant frequency and the quality factor of the piezo are 16 kHz and 20 respectively, a compensator which has zeros placed near the complex poles of the piezo and a pole at one on the z-plane is developed. The zeros of the compensator reduce the undesirable effect of the complex poles of the piezo while the pole at one on the z-plane results in zero steady state error at the system output. In order to realize the compensator, a pole is added at the origin of the z-plane so that the compensator has equal number of poles and zeros. The new compensator has slower rise time than that of the PID with more stable steady state output. The rise times of the system with the new compensator and the PID compensator are 7 and 6 sample cycles respectively from the experimental results (the sample period in our system is 30 μsec). Although the performance of the compensator developed from root-locus technique is very close to that from the PID compensator, the compensator from root-locus technique requires more lines of code when implementing it in the DSP-program.

The analysis and design of the control system that have been done in this thesis for

the AFM application focus on the transfer function of the system. Since the behavior of the piezo has a linear relationship with the output of the bicell detector, the dynamic behavior of the system can be described by state space model. Optimal control system can be developed using state space analysis [1]. However, the compensation algorithms developed from the state space analysis are usually very complicated and more lines of code may be required when implementing the algorithm into the DSP-program. Because of the time constraint, design using state space analysis has not been carried out in this thesis.

In Chapter 4 of this thesis, a genetic algorithm is used to tune the PID controller to find the optimal feedback gain. The advantage of the genetic algorithm is that it requires little knowledge of the problem and still can proceed in searching for the solution. Besides the genetic algorithm, there are different tuning methods that have been published in various paper for different application [2,3]. Further investigation may be carried out using different tuning methods to find the optimal feedback gain.

Besides the feedback control system, the scan algorithm also plays a major role in the system performance. Using a digital controller in controlling the operation of an AFM, the scan algorithm can be modified in order to increase the speed and accuracy in acquiring surface topography image. Such a modification in scan algorithm is not possible using the analog counterpart. Three different scan algorithms have been discussed in this thesis; they are the adaptive scanner, the predictor corrector and the ramp scanner. Two very useful concepts have been introduced: the 'error-threshold' and the 'predictor factor'. The 'error-threshold' indicates the difference between the desired and the actual output, which can be used as a termination condition to stop adjusting the sample height at the image points. On the other hand, the 'predictor factor' relates the vertical displacement of the piezo and the output of the bicell detector. The usefulness of the 'predictor factor' is well illustrated in the scanning algorithm 'predictor corrector'. Further investigation of developing more advanced and sophisticated scanning algorithm using these two parameters or other novel

concepts may be carried out to improve the performance of the AFM system.

The system could be further enhanced in several ways. The sampling frequency of our digital controller system is 30 kHz, which is only just fast enough for the AFM application. Faster sampling frequency can speed up the system response and reduce the acquisition time of an image. Optimization has been done on the DSP-program. However, the major hurdle of the program optimization is the C cross compiler that is used to compile the DSP-program from C language into the machine language. Since the compiler requires the Y memory of the DSP to carry out the compilation, the parallel move feature of the DSP56001 can be not fully utilized. Although the DSP-program can be rewritten in assembler language, the flexibility and ease of maintenance of the DSP-program will be reduced.

Reference

[1] C. L. Philips and H. T. Nagle, Jr. *Digital Control System Analysis and Design,* Prentice-Hall, N.J., pp. 306 - 325, (1984).

[2] J. Lee, W. Cho and T. F. Edgar "An imporved technique for PID controller tuning from closed-loop tests," *AIChE Journal,* **36** (12), pp 1891 (1990).

[3] J. Litt "An expert system to perform on-line controller tuning," *IEEE Control System Magazine,* **11** (3), pp. 18 (1991).

Appendix A

A better model of cascading a resistor with a piezo

According to the analysis of Pohl [1], only the first harmonic resonance of the piezoelectric transducer (or simply called piezo) has a significant effect to the AFM system. If only the first harmonic resonance of the piezo is considered, then the piezo can be modeled as a simple series RLC circuit [2] shown as follows:

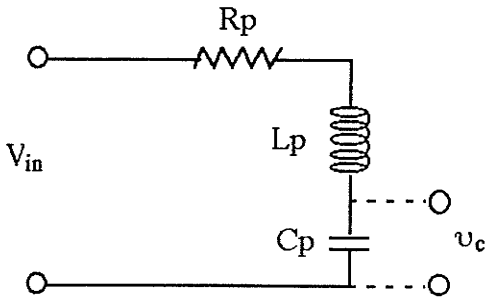

Figure A.1 Equivalent circuit of the piezo

The Rp, Lp, and Cp are the corresponding resistance, inductance, and capacitance of the piezo respectively. The voltage, υ_c, which across the capacitance Cp, determines the motion of the piezo. The impedance of the circuit is

$$Z_p = R_p + j\omega L_p + \frac{1}{j\omega C_p}$$

$$= R_p + j\left(\omega L_p - \frac{1}{\omega C_p}\right) \qquad (A.1)$$

At resonance, the reactance of the piezo equal to zero, and then the resonant frequency, ω_n, of the piezo is

$$\omega_n^2 = \frac{1}{L_p C_p} \qquad (A.2)$$

Since the motion of the piezo is determined by the voltage, υ_c, the transfer function of the

piezo motion is

$$G(j\omega) = \frac{\frac{1}{j\omega C_p}}{R_p + j\omega L_p + \frac{1}{j\omega C_p}}$$

$$= \frac{1}{1 - \omega^2 L_p C_p + j\omega R_p C_p} \qquad (A.3)$$

The Laplace transform of the equation (A.3) can be achieved by replacing jω by s, and then the equation (A.3) becomes

$$G(s) = \frac{1}{1 + s^2 L_p C_p + sR_p C_p}$$

$$= \frac{\frac{1}{L_p C_p}}{s^2 + \frac{R_p}{L_p}s + \frac{1}{L_p C_p}} \qquad (A.4)$$

Equation (A.4) can be rewritten into the form shown in equation (3.1) as

$$G(s) = \frac{\omega_n^2}{s^2 + \frac{\omega_n}{Q}s + \omega_n^2} \qquad (A.5)$$

where $\omega_n^2 = \frac{1}{L_p C_p}$, $Q = \frac{1}{R_p}\sqrt{\frac{L_p}{C_p}}$

where ω_n and Q in equation (A.5) are the resonant and quality factor of the piezo respectively. The quality factor, Q, determines the magnitude of the resonant peak of the piezo and is inversely proportional to the resistance of the piezo. If an external resistor is

cascaded to the piezo, the system becomes

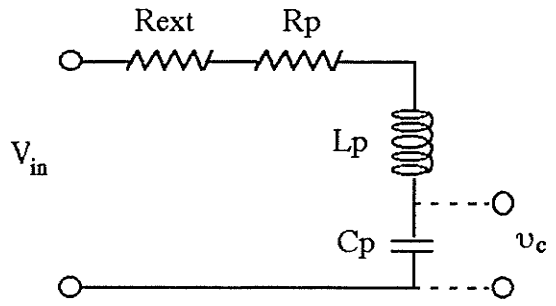

Figure A.2 Cascading the piezo with an external resistor, Rext

The new quality factor, Q', of the circuit diagram shown in figure A.2 becomes

$$Q' = \frac{1}{(R_p + R_{ext})}\sqrt{\frac{L_p}{C_p}} \qquad (A.6)$$

and the value of this new quality factor, Q', is smaller than the original one, Q. Therefore, cascading the piezo with an external resistor reduces the value of the quality factor of the system, and in turn reduces the resonant peak of the piezo. Since the resonant peak of the system can be reduced by adding the external resistor to the piezo, the system can have higher low frequency gain and larger bandwidth.

In Chapter 3 of this thesis, the effect of the external resistor is modeled as a lowpass filter. This lowpass filter model is able to approximate the system behavior since the lowpass filter reduces the resonant peak of the system. However, an extra pole from the lowpass filter is introduced to the system which actually does not exist. Therefore, the analysis of the effect of the external resistor described in this appendix is closer to the actual behavior of the system and should be used for describing the behavior of the AFM system.

Reference

[1] D. Pohl "Some design criteria in scanning tunneling microscopy," *IBM J. Res. Develop*, **30** (4), 417, (1986).

[2] R.T. Beyer, and S.V. *Physical Ultrasonics*, Academic Press, N.Y. pp. 50 - 56, (1969).

Appendix B

Z-transformation of the PI compensator

The z transform of the integrator can be derived from the time domain of integration function shown as follows

$$y(t) = K_I \int_0^t x(t')\, dt' \qquad (B.1)$$

where x(t) and y(t) are the input and output of the integrator respectively. Several numerical techniques [1] are available to approximate the integrator such as left-side rectangular rule, right-side rectangular rule, and Simpson's rule. In our system, the right-side rule is used to approximate the integrator. Figure B.1 shows the graphical representation of the right-side rectangular rule approximation of an integrator.

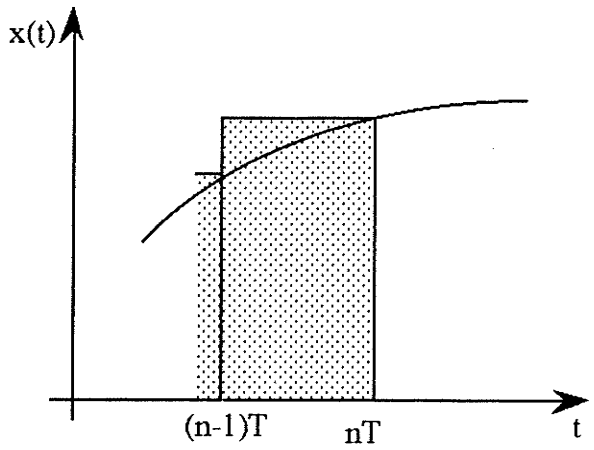

Figure B.1 Approximation of an integrator using right-side rectangular rule

The approximated integrator is

$$y(nT) = K_I T \sum_{i=1}^{n} x(iT) \qquad (B.2)$$

where T is the sampling period and n is the current sample. Equation (3.13) can be

rewritten into a difference equation as shown

$$y(nT) = y[(n-1)T] + K_I T\, x(nT) \qquad (B.3)$$

where y [(n-1)T] is the previous output of the integrator and x(nT) is the present input to the integrator. Using z transform notation, equation (3.14) becomes

$$Y = Y z^{-1} + K_I T X$$

and the transfer function of the integrator in 'z' domain is

$$\frac{Y}{X} = \frac{K_I T}{1 - z^{-1}} \qquad (B.4)$$

On the other hand, the proportional controller is simply a multiplier that multiplies the input signal by a constant called proportional gain, K_P. Thus, the transfer function of a digital PI compensator is

$$D(z) = K_P + \frac{K_I T}{1 - z^{-1}} \qquad (B.5)$$

Reference

1. C.L. Philips and H.T. Nagle. Jr. *Digital Control System Analysis and Design,* Prentice-Hall, N.J., pp. 328-354, 1984

Appendix C

Coding of the Genetic Algorithm

```
{$B-} {Boolean complete evaluation off}
{$I-} {I/O checking off}
{$O+} {Overlayable code}
{$F+} {Force far calls}
{$N+} {Numeric coprocessor}


Unit Optimizationunit; {part of the AFM program}
{ This unit carries out genetic algorithm to search for minimum
    'performance index'.  20 step curves are acquired and sorted for each
    individual, the largest and lowest 5 'performance index' are omitted
    from the 20 step curves to reduce the effect of uncertainty}

Interface
Uses
    Button2Unit,
    ColourInitial, ConvertUnit, Crt,
    Dos,
    ErrorBoxUnit,
    Graph, GreyUnit,
    Handler,
    INITIAL, InfoBoxUnit,
    MenusUnit, MeterUnit;

procedure Optimization (var Terminate : Boolean;
                                        var IMG : ImagePtr;
                                        var Acquire : Boolean);

{ Carrying out optimization process.}
{=====================================================================}
Implementation
(*********************************************************************)

Function R_num(v:real) : real;
var
    num, num1, num2 : real;
begin
    num1 := 1/(1+v);
    num2 := 1+v;
    repeat
    {randomize;}
     randseed := randseed + 1;
        num := random;
        num := num*2;   {make sure num can be greater than 1}
    until ((num>num1)and(num<num2)and(num<>1.0));
    R_num := num;
end;

(*********************************************************************)

Procedure ShellA(var items:One_Generation; count:integer);
var
    gap,i,j,k:integer;
{=======================================================}
Procedure Switch(var item1, item2 : Individual_Rec);
var item3 : Individual_Rec;
begin
    item3 := item1;
    item1 := item2;
    item2 := item3;
end;
{=======================================================}
begin
gap := count div 2;
```

```
while (gap>0) do
    begin
        for i := (gap + 1) to count do
        begin
            j := i - gap;
            while (j>0) do
            begin
                k := j + gap;
                if (items[j].opt_index <= items[k].opt_index) then
                        j := 0
                else
                        begin
                                Switch(items[j],items[k]);
                                j := j - gap;
                        end;
                end;
            end;
        gap := gap div 2;
    end;
end;

(*************************************************************************)

Procedure ShellB(var datas:temp_array_type; count:integer);
var
    gap,i,j,k:integer;
{=======================================================}
Procedure Switch(var data1, data2 : real);
var data3 : real;
begin
    data3 := data1;
    data1 := data2;
    data2 := data3;
end;
{=======================================================}
begin
gap := count div 2;
while (gap>0) do
    begin
        for i := (gap + 1) to count do
        begin
            j := i - gap;
            while (j>0) do
            begin
                k := j + gap;
                if (datas[j] <= datas[k]) then
                        j := 0
                else
                        begin
                                Switch(datas[j],datas[k]);
                                j := j - gap;
                        end;
                end;
            end;
        gap := gap div 2;
    end;
end;

(*************************************************************************)

Function Index(Row : RowArray) : Real;
{Calculate the optimization index of each row}
const ceiling = 1024;
        num_of_pts = 256;
```

```
var i : integer;
    total_error : longint;
begin
total_error := 0;
for i := 0 to (num_of_pts-1) do
       total_error := total_error + abs(abs(Row[i] - Row[0]) - ceiling);
   Index := total_error/num_of_pts;
end; {for i := 0 to (num_of_pts-1) do begin}

(*************************************************************************)

procedure Optimization (var Terminate : Boolean;
                                    var IMG : ImagePtr;
                                    var Acquire : Boolean);
  { Carrying out optimization process.}
const
    random_factor = 1.01;     {better close to one}
    trial = 10;
    num = 5;
    population = num*num-num;      {population must be equal (num^2-num),}
    {where num is an integer using in counting the for loop in crossover}
    generation = 50;
var
    ch : char;
    Done : boolean;
    prompt : string[32];
    i,ii,parent_counter,offspring_counter : integer;
    winner : Individual_Rec;
   mutation_rate : real;
    parent,offspring,next_generation : One_Generation;
    Generation_File : File of Individual_Rec;
    answer : char;
    x,y :integer;
    File_to_Save,Dat_Name :string[16];
    Indexs : temp_array_type;

{=========================================================}

Procedure First_Generation (var first_parent : One_Generation);
{ ***  Generating the first generation  *** }
var temp_opt_index : real;
    counter1, counter2 : integer;
    first_R_num, second_R_num : real;
begin
   EraseInputBox;
   InputBox;
   SetColor(cadmium_lemon);
   SetTextStyle(2,0,6); SetTextJustify(LeftText,TopText);
   OutTextXY(470,550,'Generating first generation');

   for counter1 := 1 to population do begin
      if counter1 = 1 then begin
         first_parent[counter1].p_const := Proportional_Feedback;
         first_parent[counter1].i_const := Integral_Feedback;
      end
      else begin
      repeat
         first_R_num := R_num(mutation_rate);
         first_parent[counter1].i_const    :=    first_parent[1].i_const    *
first_R_num;
      until ((first_R_num<1.5)and(first_parent[counter1].i_const<999));
      repeat
         second_R_num := R_num(mutation_rate)*random_factor;
         first_parent[counter1].p_const    :=    first_parent[1].p_const    *
```

```
second_R_num;
        until
((second_R_num<>first_R_num)and(first_parent[counter1].p_const<999));
        end;

        Convert_Prop_Feedback(first_parent[counter1].p_const,Acquire);
        Convert_Int_Feedback(first_parent[counter1].i_const,Acquire);

        counter2 := 1;
        temp_opt_index := 0;
        Row_To_Draw := FALSE;
        repeat
        {loop and look for action}
        if (Row_To_Draw) then
        begin
            Indexs[counter2] := Index(IMG^[New_Row_Number]^);
            counter2 := counter2 + 1;
            Row_To_Draw := FALSE;
        end; {if (Row_To_Draw) then}
        until counter2 > 20{(trial*2)};
        {end code to collect trial*2 step curves}

        ShellB(Indexs,20);
        for counter2 := 6 to 15 do begin
            temp_opt_index := temp_opt_index + Indexs[counter2];
            end;

        counter2 := 1;    {reset to zero}

        first_parent[counter1].opt_index := temp_opt_index/10;
        first_parent[counter1].parent := True;
     first_parent[counter1].vg := mutation_rate;
    end; {for counter1 := 1 to population do begin}

    ShellA(first_parent,population);
end;
{=========================================================}

procedure CrossOver (parent : One_Generation;
                                    var offspring : One_Generation);
var i,j,k : integer;
begin
{    EraseInputBox;
    InputBox;
    SetColor(cadmium_lemon);
    SetTextStyle(2,0,6); SetTextJustify(LeftText,TopText);
    OutTextXY(470,550,'CrossOver');}

    k := 1;
    for i := 1 to num do begin
        for j := 1 to num do begin
            if (i<>j) then begin
                offspring[k].i_const := parent[i].i_const;
                offspring[k].p_const := parent[j].p_const;
          offspring[k].parent := False;
                k := k + 1;
            end;
        end;
    end;
end;

{=====================================================}

procedure Mutation (var offspring : One_Generation);
```

```
var i : integer;
    temp : double;
    first_R_num, second_R_num : real;
begin
{    EraseInputBox;
   InputBox;
   SetColor(cadmium_lemon);
   SetTextStyle(2,0,6); SetTextJustify(LeftText,TopText);
   OutTextXY(470,550,'Mutation');}

   for i := 1 to population do begin
      repeat
         first_R_num := R_num(mutation_rate);
         temp := offspring[i].i_const * first_R_num;
      until ((first_R_num<1.5)and(offspring[i].i_const<999));
      offspring[i].i_const := temp;
      repeat
         second_R_num := R_num(mutation_rate)*random_factor;
         temp := offspring[i].p_const * second_R_num;
      until ((second_R_num<>first_R_num)and(offspring[i].p_const<999));
      offspring[i].p_const := temp;
    offspring[i].vg := mutation_rate;
   end;
end;

{=================================================}

procedure Evaluation (var parent : One_Generation;
                                          var offspring : One_Generation);
var
   temp_opt_index : real;
   mix_generation : One_Generation;
   counter1,counter2 : integer;
   temp_p_const,temp_i_const : double;
begin
{    EraseInputBox;
   InputBox;
   SetColor(cadmium_lemon);
   SetTextStyle(2,0,6); SetTextJustify(LeftText,TopText);
   OutTextXY(470,550,'Evaluation');}

   counter2 := 1;
   for counter1 := 1 to (population*2) do begin
      if counter1 <= population then begin
         temp_p_const := parent[counter1].p_const;
         temp_i_const := parent[counter1].i_const;
      end
      else begin
         temp_p_const := offspring[counter1-population].p_const;
         temp_i_const := offspring[counter1-population].i_const;
      end;

      if counter1 > population then begin
         Convert_Prop_Feedback(temp_p_const,Acquire);
         Convert_Int_Feedback(temp_i_const,Acquire);

         temp_opt_index := 0;
         Row_To_Draw := FALSE;
         repeat
         {loop and look for action}
         if (Row_To_Draw) then
         begin
            Indexs[counter2] := Index(IMG^[New_Row_Number]^);
            Inc(counter2,1);
```

```
            Row_To_Draw := FALSE;
         end; {if (Row_To_Draw) then}
         until counter2 > 20{trial};
         {end code to collect 20 step curve}

         ShellB(Indexs,20{(trial*2)});
         for counter2 := 6 to 15 do begin
            temp_opt_index := temp_opt_index + Indexs[counter2];
            end;

         counter2 := 1;    {reset to zero}

      end; {if counter1 > population then begin}

      if counter1 <= population then
         parent[counter1].opt_index     :=     parent[counter1].opt_index     +
temp_opt_index/10
      else
         offspring[counter1-population].opt_index := temp_opt_index/10;

   end; {for counter1 := 1 to (population*2) do begin}
end;

{=================================================}

procedure Choose_Next_Generation(parent,offspring : One_Generation;
                                                                    var
next_generation : One_Generation);
const
   min_seperation = 0.5;
var
   j,k:integer;
   counter1 : integer;
   mix_generation : One_Generation;
   dist1,dist2,func1,func2 : real;
begin
{    EraseInputBox;
   InputBox;
   SetColor(cadmium_lemon);
   SetTextStyle(2,0,6); SetTextJustify(LeftText,TopText);
   OutTextXY(470,550,'Choosing Next Generation');
   delay(500);}

   for counter1 := 1 to population do begin
      mix_generation[counter1] := offspring[counter1];{parent[counter1];}
      mix_generation[counter1+population]                                    :=
parent[counter1];{offspring[counter1];}
   end;
   EraseInputBox;
   InputBox;
   SetColor(cadmium_lemon);
   SetTextStyle(2,0,6); SetTextJustify(LeftText,TopText);
   OutTextXY(470,550,'ShellSort');

   ShellA(mix_generation,population*2);
   next_generation := mix_generation;
end;

{=================================================}

begin
   {** initialization **}
   step := TRUE;
   toggle_step(step);
```

```
step := False;
toggle_step(step);

opt := TRUE;
toggle_opt(opt);

InfoBox;
RemoveLegend;
RemoveEverything;
RemoveMeter;

for i := 1 to 100 do InLine($90);

{start section of code to enable interrupts from DSP}
Port[PC_ISR] := $20;
Port[PC_ISR] := $20;

GetIntVec ($f, Old_Int_Vec);
SetIntVec ($f, @Get_Row_Of_Data);

Old_Interrupt_Mask := Port[PC_IMR];
Port[PC_IMR] := (Port[PC_IMR] AND $7F);

Port[DSP_ICR] := 1;
{end section of code to enable interrupts from DSP}
(*********************************************************************)

{initialization}
for i := 1 to population*2 do begin
    parent[i].opt_index := 1e10;
    offspring[i].opt_index := 1e10;
end;
parent_counter := 0;
offspring_counter := 0;
mutation_rate := 0.4;

EraseInputBox;
InputBox;
SetColor(cadmium_lemon); SetTextStyle(2,0,5);
SetTextJustify(LeftText,TopText);
x := Input_Box_x1 + 9;
y := Input_Box_Y1 + 9;
OutTextXY(x,y,'Save data ? (Y/N) : ');
answer := ReadKey;
if answer in ['a'..'z'] then
    answer := Chr(Ord(answer)-32);
if (answer = 'Y') then begin
    File_to_Save := File_to_Load(False);
    if (File_to_Save <> '') then begin
        Dat_Name := Concat(File_to_Save,'.dat');
        Assign(Generation_File,Dat_Name);
        Rewrite(Generation_File);

        First_Generation(parent);
        for ii := 1 to population do
            Write(Generation_File,parent[ii]);

        for i := 1 to generation do begin

            CrossOver(parent,offspring);
            Mutation(offspring);
            Evaluation(parent,offspring);
            Choose_Next_Generation(parent,offspring,next_generation);
            for ii := 1 to population do begin
```

```
                    parent[ii] := next_generation[ii];
                    if parent[ii].parent then Inc(parent_counter,1)
                    else Inc(offspring_counter,1);
                    Write(Generation_File,next_generation[ii]);
                    parent[ii].parent := True;
                end;
                parent_counter := 0;
            offspring_counter := 0;
                winner := next_generation[1];

                EraseInputBox;
                InputBox;
                SetColor(cadmium_lemon);
                str(i:2,prompt);
                OutTextXY(Input_Box_X1+20, Input_Box_Y1+25,prompt);
                str(winner.i_const:10:3,prompt);
                OutTextXY(Input_Box_X1+30, Input_Box_Y1+25,prompt);
                str(winner.p_const:10:3,prompt);
                OutTextXY(Input_Box_X1+130, Input_Box_Y1+25,prompt);
                str(winner.opt_index:10:3,prompt);
                OutTextXY(Input_Box_X1+230, Input_Box_Y1+25,prompt);

            end; {for i := 1 to generation do begin}

            EraseInputBox;
          InputBox;
            SetColor(cadmium_lemon);
            str(winner.i_const:10:3,prompt);
            OutTextXY(Input_Box_X1+30, Input_Box_Y1+25,prompt);
            str(winner.p_const:10:3,prompt);
            OutTextXY(Input_Box_X1+130, Input_Box_Y1+25,prompt);

            repeat
                ch := ReadKey;
            until (ch=#13);
            EraseInputBox;

            Close(Generation_File);
        end; {if (file_to_Save <>'')}
    end; {if answer = 'Y'}

    {*********************************************************************}

    {start section of code to disable interrupts from DSP}
    Port[DSP_ICR] := 0;
    Port[PC_IMR] := Old_Interrupt_Mask;
    Port[PC_ISR] := $67;
    SetIntVec ($f, Old_Int_Vec);
    {end section of code to disable interrupts from DSP}

    InitMeter;
    Convert_Set_Pt(I_Set_Pt,FALSE);
    Convert_Prop_Feedback(Proportional_Feedback,FALSE);
    Convert_Int_Feedback(Integral_Feedback,FALSE);
    Convert_Bias_Voltage(Bias_Voltage,FALSE);
    Convert_Feedback_Num(Feedback_Num,FALSE);
    Convert_X_Offset(X_Offset,X_Size,Acquire);
    Convert_Y_Offset(Y_Offset,Y_Size,Acquire);
    Convert_X_Size(X_Size,Acquire);
    Convert_Y_Size(Y_Size,Acquire);

end; {procedure Optimization}
(*************************************************************************)
End. {unit OptimizationUnit}
```

Appendix D

Predictor Corrector and Integrator

According to the equation (3.18), the transfer function of the integrator is

$$D(z) = \frac{K_I T}{1 - z^{-1}} \qquad (D.1)$$

If X(z) and Y(z) are the z-transform of the input and the output of the integrator, equation (D.1) can be rewritten as

$$\frac{Y(z)}{X(z)} = \frac{K_I T}{1 - z^{-1}} \qquad (D.2)$$

From Appendix B, the closed form of the integrator in the time domain is

$$y(nT) = K_I T \sum_{i=1}^{n} x(iT) \qquad (D.3)$$

$$y(nT) = K_I T * [x(T) + x(2T) + \ldots + x(nT)] \qquad (D.4)$$

If the input to the integrator is the error signal, e(t), from the bicell detector, then the output of the integrator becomes

$$y(nT) = K_I T * [e(T) + e(2T) + \ldots + e(nT)] \qquad (D.5)$$

On the other hand, the predictor corrector multiplies the error signal, e(t), with the predictor factor, p. The result is added to the previous predictor value to the Z-piezo. For example, let h(t) be the predictor value, then the mathematical representation of the predictor corrector can be written as

$$h(nT') = h[(n-1)T'] + p * e(nT') \qquad (D.6)$$

The sampling period of the predictor corrector, T', may be different from that of the integrator since T' depends on the rise time of the piezo and the corresponding lowpass

filter. Equation (D.4) can be rewritten as

$$h(nT') = p * [e(T') + e(2T') + ...+ e(nT)] \qquad (D.7)$$

$$h(nT') = p * \sum_{i=1}^{n} e(iT') \qquad (D.8)$$

Equation (D.3) and (D.8) are virtually the same. Therefore, the predictor corrector is actually an integrator with different integral gain and sampling period.