NOTICE

AVIS

A Computer Vision System for VLSI Wafer Probing

Ramanamurthy V. Dantu

A Thesis

in

The Department

of

Electrical and Computer Engineering

Presented in Partial Fulfillment of the Requirements
for the Degree of Doctor of Philosophy at
Concordia University
Montréal, Québec, Canada

March 1990

ISBN 0-315-59176-5

ABSTRACT

A Computer Vision System for VLSI Wafer Probing

Ramanamurthy V. Dantu, Ph.D.
Concordia University, 1990

This thesis is concerned with an important problem encountered in automating VLSI wafer probing. In wafer probing, metallic probes are used to inject test vectors through a set of input/output (test) pads. The accurate and sensitive operation of touching a probe onto a pad has to be performed repeatedly with good resolution. This operation, currently performed by human operators, demands great accuracy which is only available through highly skilled labour. The probe tip has a thickness of less than a micron and is very fragile. Hence it is essential that the touch of the probe onto the target wafer surface be as soft as possible while ensuring a reliable electrical connection. In this thesis we describe a method for guiding the probes using vision as feedback. The use of a multiprocessor is also investigated as a means of speeding up the low level vision algorithms used in the method.

Accurate information on the distance of a probe from the wafer is important for the successful guidance of the probe to its destination. The camera in the vision system is focused on the wafer and several images of the probe as it approaches the wafer are analyzed. A theory has been formulated for calculating the degree of blur using the step height of the probe's edge and the slope of the intensity profile at the location of the edge. The resulting formula provides a measure of the proximity of the probe to the surface of the wafer. This formula is simple and easy to use. It is also robust in the sense that it is valid even in the presence of significant noise in the images. When the probe is very close to (i.e. about to touch) the wafer surface, accurate detection of touch is very important for non-destructive wafer probing. The variance of pixel values of the image of the probe together with insight gained from experimental observations are employed for accurate determination of contact between the probe and the wafer. The methods proposed in this thesis have been validated by experimentation with various images of a probe approaching different VLSI wafers containing metal pads.

Parallel low level vision algorithms are implemented on a simulator specifically written for the Homogeneous Multiprocessor. Substantial speedups have been shown for local algorithms such as smoothing and edge detection, as well as nonlocal ones such as histogram generation.

ACKNOWLEDGEMENTS

I would like to thank my supervisors and friends, Nikitas Dimopoulos, Rajnikant Patel and Asim Al-Khalili, for their guidance, support and encouragement. All of them have spent a lot of time and effort in preparing me for meeting the objectives of the thesis. Nikitas Dimopoulos has been very helpful in discussions regarding various aspects of the depth perception and vision algorithms. I can not forget his patience during the discussions. Rajni Patel has introduced me to this research topic and his constant help at various stages of the experimental work greatly contributed to the success of this research. Certainly I will miss his helping hand. He has improved my writing skills and I owe him for this. Asim Al-Khalili has been very helpful in several aspects of my university life and I greatly appreciate his help. In brief, I am fortunate to have three great human beings as my supervisors.

Many thanks go to the office mates with whom I have spent time during my years at concordia university. I am especially thankful to Costas, Hamid, Nagamuttu, Piyu, and Vlad. Costas and Hamid participated in the technical discussions concerning my thesis. Discussions with Murali Subbarao of SUNY, Stony Brook have been very useful for the experiments in depth perception. I thank the technical staff of the Computing Lab for their support throughout the course of this work. Henry Kovalcik has been particularly helpful at various stages of the experimental work. I also thank David Stewart for his help in building the image processing package.

I would like to thank my parents Sivaramakrishna Sarma and Venkata Lakshmi for their encouragement and moral support. Their motivation has been a constant source of guidance and support for me. Last but not least I would like to thank my wife Srilakshmi for her help, patience and support during the last four years of this research. The timely completion and submission of this thesis would not have been possible without her.

DEDICATION

To my parents *Sivaramakrishna Sarma* and *Venkata Lakshmi*, and my wife *Srilakshmi*.

Table of Contents

List of Figures

List of Tables

Chapter 1
Computer Vision in IC Manufacturing

1.1 Introduction

Vision is one of the most powerful and complex of human senses. It enables us to interact intelligently in the world without direct physical contact. It also enables us to understand positions and identities of objects and the relationships between them. It is natural that we attempt to provide computers with a sense of vision. Computer vision is used in various industrial applications, some of the more recent ones being in the semiconductor industry. To date, computer vision is used in several stages of IC manufacturing, the more prominent ones being wafer inspection, testing, die bonding and wire bonding.

Inspection and testing of VLSI wafers are becoming increasingly difficult to perform. This difficulty is due to the continuous improvement of the fabrication process which has resulted in reduced feature size and increased die size [1], [2]. Consequently, more devices are put in a unit area which in turn increases the fault density and reduces the yield*. Goldstein describes a simple model [3] for yield Y as

$$Y = e^{-\sqrt{AD}} * 100\% \qquad (1.1.1)$$

where A is the area of the die in cm^2 and D is the defect density in defects/cm^2. Thus the model indicates a lower yield as the die size increases. Future demands will be even more stringent with yield dropping further, not only due to the increases in packing density, but also because of increases in the relative size of particle-to-device geometry. In spite

* Yield is calculated as the ratio of the number of good parts to the total number of parts manufactured.

of the dramatic changes in feature size, die size, and defect density, several functions in inspection and testing continue to be done manually [2]. Automating various aspects of IC manufacturing provides repeatability and better control of the quality of the ongoing manufacturing process. This in turn tends to decrease the defect density and thus improve the yield. The growing need to automate critical functions in IC manufacturing was realized early on. Industries were surveyed to determine the detailed requirements and the relative importance of each function. Based on this survey, in [1] and [2] Harris reported a summary of current and future needs for wafer inspection, automation, and yield benefits.

Automation in IC packaging using machine vision has been explored by various researchers [1], [2] and [4]—[12]. Most of the papers deal with wafer inspection, wire bonding, and die bonding. To our knowledge, no work has been reported so far in automated testing of in-process wafers. We think that IC inspection for localization of physical faults and wafer testing for functional and parameter estimation will benefit the most from automating the different stages of IC packaging. The objective of this thesis is to study the problem of automating wafer probing, which is used in the testing of various semiconductor parameters.

In this chapter, we describe various steps in IC packaging which benefit from computer vision, and describe the need for automated wafer probing. Sections 1.1.1, 1.1.2 and 1.1.3, briefly overview different tasks of wafer inspection, wafer testing and wire bonding which can be automated using computer vision. In Section 1.2, we describe the need for wafer probing and the techniques used presently. The need for a computer vision system for wafer probing is explained in Section 1.3. Next, the various components in a typical computer vision system and their operations are explained in Section 1.4. Finally, the objectives and organization of this thesis are described in Section 1.5 and 1.6.

1.1.1 Wafer Inspection

An application of machine perception in the semiconductor industry is in the growing field of IC screening [1], [2], [4], [8], [11], and [12]. Here, inspection of IC chips is carried out to detect defects. This is done in-process before pre-cap testing either by feature inspection or by generic property verification. In either case, inspection can be automated. A well defined image of the die on the wafer is an essential ingredient for this type of inspection. The end result is defect detection, identification and classification. The inspection is usually done in two major steps: macro-inspection and micro-inspection. During macro-inspection, the full wafer is inspected and a quick check is made for large defects such as contamination (down to 10 microns), poor spin, and non-uniformities of exposure or development. Macro-inspection is a fast way of avoiding catastrophic yield loss due to repeating defects. Micro-inspection includes registration checking, pattern checking for systematic defects and point defect checking. This inspection allows automatic tracking of the manufacturing process and is usually used as an advance warning that the process is drifting. The majority of detectable faults fall into one (or more) of the following classes [4]:

- open circuit on any layer
- short circuit between points in the same layer
- minimum width violation of any layer, particularly conductors
- minimum spacing violation between layers, particularly conductors
- minimum overlaps and poor step coverage
- crack faults
- conductor corrosion, etc.

Automated wafer inspection is a popular research area and several articles have been published related to it. Harris [1], [2] reported an automated inspection system which requires 5 minutes for macro-inspection and 13 minutes for micro-inspection of a wafer of size 1cm x 1cm. These timings are found to be 6 to 8 times faster than the human operator. Recently Yoda [13] reported a system which derives a minimum of 0.6 micrometer defective patterns from the multilayered wafer patterns at a speed 30 times faster than that of a human inspector. The false alarm rate is less than 0.5 occurrences/chip. All image processing of the system is performed in a one-pass manner by a high speed pipeline-structured image processor that can analyze an input image signal at 7 Mbytes/sec video rate.

1.1.2 Wafer Testing

Once macro and micro-inspections have been performed and defective dies have been identified, the remaining dies are tested electrically and are verified functio:ally. Lesign considerations for integrated circuits consist of two parts: the functional specifications and the specifications of various semiconductor parameters. The first set of specifications is verified using boolean testing of the ICs. The second set is verified using parametric analysis.

Generation of various test vectors for boolean testing has been a topic of interest in "design for testability" [14]. In boolean testing the test points on the die are normally the bonding pads. However, most VLSI chips produced today have convenient and well determined areas for contact probing called "test pads" [8], [14]. These pads are inserted in order to enhance controllability and observability of the VLSI circuits [3]. So, VLSI circuits now a days are designed with testability in mind [14]. Test pads are included in the circuit to

i. overcome fault blocking (a situation where a test vector is prevented from propagating through the network to the outputs).

ii. reduce test vector generation or storage area.

iii. speed up testing time.

The AC and DC parametric analysis is performed by testing all the semiconductor parameters. The representative AC timing parameters are: set-up time, hold time, and propagation delays. During set-up and hold times, the data has to be stable before and after an active clock transition occurs. The propagation delay is the time between a stimulation and its reaction at the output of the device. Examples of DC parameters are: i) threshold voltage (the gate voltage required to cause a predetermined value of drain current to flow); ii) transconductance (device ability to drain current in response to gate voltage variations, with drain-source voltage constant); iii) channel conductance (ratio of drain current I_{DS} to drain-source voltage V_{DS} when V_{DS} is near zero).

For measuring the above mentioned parameters, test vectors are stimulated and signatures are picked up at the primary input/output bonding pads using micro-probes. Whichever testing method is used, to be able to test a circuit completely, all the bonding pads and test pads must be accessible for probing. A popular method is to use mechanical probe needles for inserting test vectors. Hence, wafer probing requires the following:

i. Identification of the bonding and test pads.

ii. Placement of selected probes on the appropriate pads for non-destructive contact probing (i.e. achieving a good electrical contact without scratching the pad).

iii. Active testing through signature analysis.

Future developments will also create problems in this task because of smaller pads and more pads on a single die with less clearance between them. Therefore, there will be more probes operating at the same time, sometimes more than a hundred probes!.

The above mentioned tasks are presently performed manually. The research described in this thesis is aimed at automating such tasks by using vision as feedback. More details of the wafer probing operations are explained in Section 1.2 and 1.3.

1.1.3 Bonding

Another area that benefits from the use of machine perception in IC packaging is die and wire bonding [5], [6]. During the final stages of packaging, the ICs to be assembled are placed on chip carriers. Two sub-tasks are performed here:

i. Die bonding: This involves identification of various ICs (as they arrive) by pattern matching using predefined IC features, and bonding each die on a chip carrier with a fixed accuracy.

ii. Wire bonding: This is the process of connecting gold or aluminum wires between the pads on an IC and the leads of the lead frame.

The objective of the automated system is to identify the bonding pads which are placed along the outer edges of the die. The shape and size of the bonding pads are usually standard. Together with the test pads they are normally of square shape with a top layer of metal which reflects light more than the other layers. Generally bonding pads have a larger diffused component than their surroundings, making them easily identifiable. Future reductions of pad sizes, smaller clearances between pads, and the increased number of pads per die will require more accurate recognition as well as manipulation techniques.

1.2 Wafer Probing

An important factor in improving yield is to study the relationship between fault coverage and the quality of integrated circuits [15]. Three classes of wafer tests are performed on integrated circuits: i) Gross tests, to check for open/short circuits (see Section 1.1.1), presence of input protection circuits, etc., ii) Functional or boolean tests, which ensure that the device logic behavior is correct, and iii) Parametric tests which measure voltage, current and timing parameters (see Section 1.1.2) [15]. The yield loss identified through these three tests has been studied in [15]. In some experiments on the Motorola 68020, 93 wafers from 5 lots of varying yield were tested. Gross failures eliminated 4,006 of the total of 22,506 dies, 6,444 dies failed the boolean test and 5,000 failed the parametric test. Significant improvements in yield can be achieved by employing these tests during fabrication. Hence, development of "hands off" automated probing equipment will definitely be useful for designers and manufacturers of semiconductor circuits [9].

There are three classes of probing equipment available on the market: mechanical probing, electronic beam probing and mercury contact probing. Probing using metal needles (mechanical probing) is the most popular.

In electron beam probing, the electrical state inside a wafer is analyzed by irradiating the wafer with an electron beam (having a small diameter). This method is used while the circuit is operating. In this method, the node (e.g. a metal pad) to be investigated is exposed to primary electrons, thereby causing the emission of secondary electrons. The number of secondary electrons emitted at the node is dependent on the surface potential. When the node is at a high potential, it attracts back a large proportion of the secondary electrons. The secondary electron collector then sees fewer electrons. When the node is at a low potential, the collector sees a larger number of electrons. Hence, by projecting all the points on the wafer to a two dimensional grid, an image representing wafer potentials can be created. This image gives the complete electrical state of the wafer under test. Because, there is no physical contact, this method of probing will not cause any mechanical destruction of the IC surface. Another advantage over its mechanical counterpart is that it can be used to scan the surface of an IC and thereby test large circuit areas in a single stroke [16]. In recent years a great deal of work has been published in the field of electron beam testing and a summary can be found in [17].

There are also some limitations of electron beam testing. The probing must be done in vacuum and there is a problem of instability due to the charge-up phenomenon. Irradiation damage of the IC is possible because of the high voltage levels of the primary electron source. Also, due to lower line widths in the die, fewer secondary electrons are available for measurement, hence long measurement times must be accepted as a trade-off [17].

The demand for "safe" wafer probing has resulted in yet another technique. Material Development Corporation [18] has developed a mercury probe. In this probing technique, the measurement is done by making contact on the wafer with a mercury dot in a well-defined area. Such a probe can be connected to various measuring instruments. On bare silicon, mercury forms a Schottky barrier that enables the analysis of doping profiles. This kind of probing is non-destructive, but to our knowledge, this method has not been used by other manufacturers.

1.2.1 Probing using Metal Needles

As mentioned earlier, mechanical probing using metal needles is widely used and is the most popular. There are two types of mechanical wafer probing: one for high volume IC production and other for low volume. Probe cards are used for high volume production where the ICs are tested for certain known electrical parameters. In these probers, the wafer is loaded and aligned automatically, probes are lowered a certain distance (without any feedback) and electrical measurements are made. At present this method is used at the final testing stages of the finished wafers before packaging. Because there is no feedback provided, good electrical contact cannot be made consistently. Moreover, probing causes a loss of more than 10 percent in the yield [9], mostly due to the damage caused by the probes themselves.

Analytical probing is generally manual and is suitable for low volume IC production or when a few dies per wafer need to be tested. In this type of probing, X and Y positioning is done manually and the probes are lowered manually to the target pad. Important uses for analytical probing occur in the design of a new device and in the failure analysis laboratory. These applications employ microanalytical probing which requires a higher degree of accuracy since one has to access geometries of less than one micron. Moreover, not all manual probing takes place during the final testing of the wafer. Wafer probing can also take place during wafer fabrication for checking the electrical characteristics of the device and for monitoring the processing equipment's adherence to wafer specifications. The research described in this thesis is aimed at automating this type of probing using computer vision. This will also help to reduce the lead time between the conceptual stage and the actual production of the circuits.

1.2.1.1 Materials used for Probe Needles

Generally, three materials are used for probe needles [19]—[21]. They are Tungsten, Beryllium Copper, and Paliney–7 Alloy. Tungsten is commonly used because of its hardness and high yield point. Also, it can be electropolished to an extremely fine tip. Beryllium Copper has a higher bulk conductivity than tungsten and offers better electrical characteristics. However, it is a softer material and so is not useful for most analytical probing and trace cutting applications [22]. Paliney has a lower contact resistance and is an expensive material. It has higher mechanical strength than beryllium copper but less than tungsten, with better electrical characteristics than tungsten. Laboratories stock all three materials so that they can best match a material with an applications.

1.3 The Need for Soft Probing

Metallic probes are used to inject test vectors and check for continuity [9]. Input, output or test pads are used for such probing (as explained in Section 1.1.2). The test pads have dimensions of the order of 10 microns or less. The accurate and sensitive operation of touching a pad with the probe has to be performed repeatedly for testing each parameter. An important task in probing is to achieve good metal-to-metal contact between the probe and the device. Resistance is often caused hy oxidation and dirt on the device. Hence these two layers must be broken to make good metal-to-metal contact. One way to break the oxide layer is to increase the probe pressure by increasing the Z drive on the probe. As the probe pressure increases, contact resistance decreases. However, this must be balanced against excessive pressure, which could cause damage to the device or to the tip. Silicon devices are prone to these microfractures and recent Gallium Arsenide devices are extremely susceptible to such damage.

Another concern while probing the wafer is the fragility of the probe tip. The probe tip, with a thickness of less than a micron, is very fragile. Hence for non-destructive touching of a wafer by the tip of the probe, the probe should have zero velocity at the moment of touch. Our observations have shown that when the probe is being lowered and the velocity of movement is small, the probe does not initially pierce the pad on contact, but slides on the pad without scratching the surface. Hence, it is very important to detect the slide before any damage takes place. The above described tasks could use vision feedback for better and more accurate control.

1.4 A Computer Vision System

Computer vision is a key component of automated manufacturing systems. It is required for tasks involving identification, location, verification, inspection, as well as for guidance of the probes. In this sectior, we describe the processing steps involved in a typical computer vision system [23]. Different stages of processing in the vision system presented in this thesis closely resemble the classification given in [23]. This hierarchical classification of the computational tasks performed in a computer vision system is presented in Figure 1.1. Low level processing deals with noise removal, edge detection, image enhancement, two-dimensional feature detection and region analysis. This processing results in the partitioning of an input image into meaningful parts. These parts correspond (in our case) to physical objects such as probes, pads and background. Grey levels of the image are used for this purpose.

The next level consists of intermediate processing, where the image is analyzed for three dimensional information (depth from focus in our application). High level processing deals with relational analysis and coordination of various low level modules. A given task can be divided into several high-level actions. These high-level actions (such as touch, move, lift) are achieved by coordinating low level-features such as mean intensity of the scene, orientation and size of the objects, and intermediate level features (such as focus, proximity and zoom). Sometimes, it is required to refer to a knowledge base containing various characteristics of the scene before making a high-level decision. Thus, it is required to execute layers of algorithms for a given task. On the other hand, there is also a need for real-time operation in some applications (like ours). Hence, speeding up the above mentioned algorithms is another concern in a computer vision system.

Extensive research and experimentation has been done for the last 30 years in all three stages of processing. But due to the complexity of visual perception, no general purpose computer vision system is yet available.

1.5 Motivation and Objectives of the Thesis

Several laboratories [24] have reported difficulties in carrying out manual wafer probing. This is because the operations involved in probing are sensitive, time consuming and require highly skilled operators. On many occasions either the probe tip or the wafer have been damaged while probing. The availability of completely automated probing

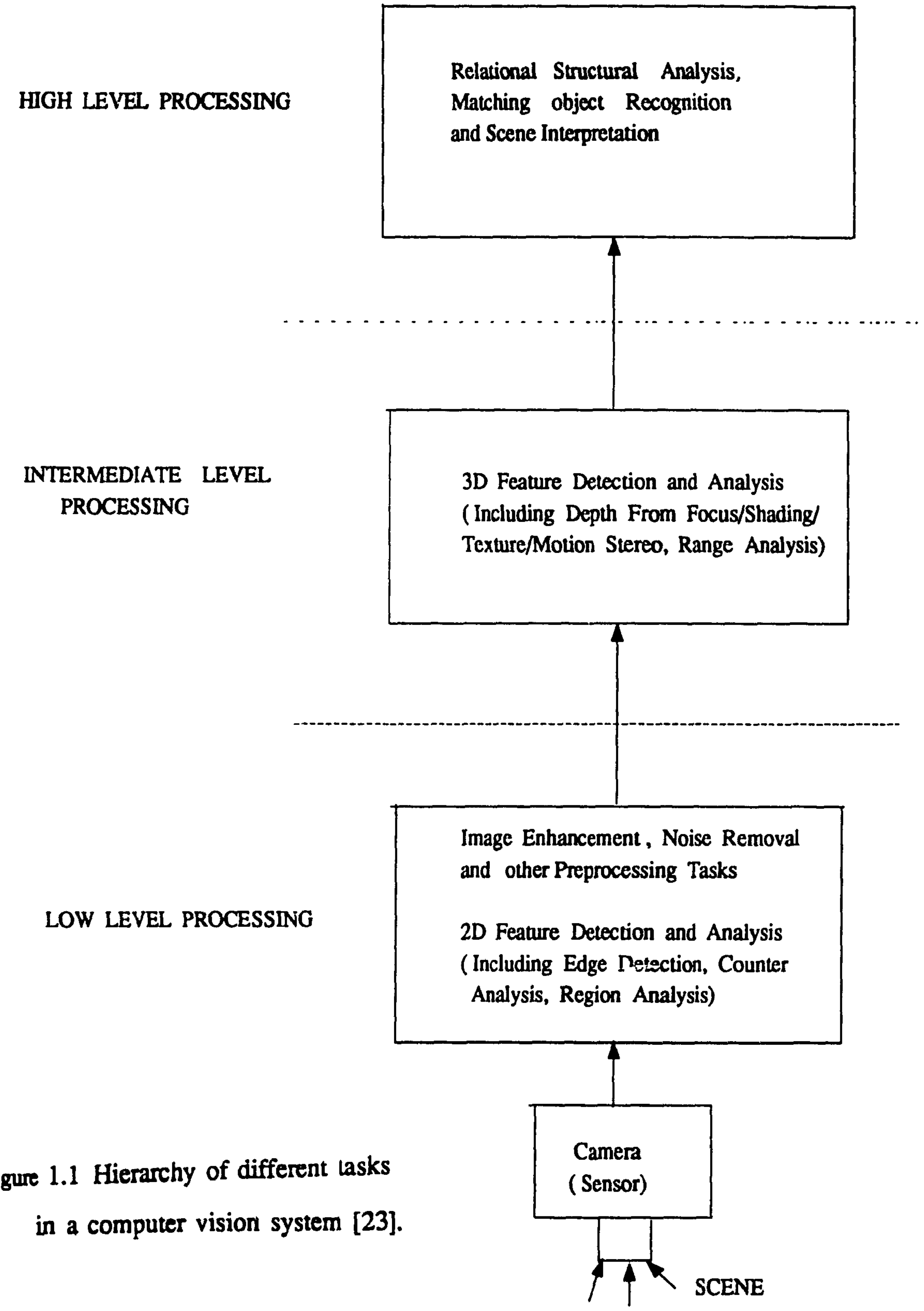

HIGH LEVEL PROCESSING
Relational Structural Analysis, Matching object Recognition and Scene Interpretation
INTERMEDIATE LEVEL PROCESSING
3D Feature Detection and Analysis (Including Depth From Focus/Shading/ Texture/Motion Stereo, Range Analysis)
LOW LEVEL PROCESSING
Image Enhancement, Noise Removal and other Preprocessing Tasks
2D Feature Detection and Analysis (Including Edge Detection, Counter Analysis, Region Analysis)
Camera (Sensor)
SCENE

Figure 1.1 Hierarchy of different tasks in a computer vision system [23].

equipment could alleviate these difficulties. We have surveyed some commercially available equipment (details are given in Chapter 2) but have not been able to find a completely automated one. Automating the operations involved in wafer probing would definitely result in more accurate measurements and ultimately better devices and improved yields. In this context, we have found some challenging research problems, one such problem being that of a probe to a target surface on a wafer.

There are two important tasks in automating wafer probing. The first is to lower the probe to the pad and the second is to make good electrical contact without scratching the surface. We decided to use computer vision to solve these problems. We have several constraints in solving these tasks. First, we can only view the wafer from one direction; second, the wafer is illuminated by bright-through-the-lens illumination; and third, the probe is inclined with respect to the optical axis. In addition to these constraints, all the vision algorithms need to be general and independent of the patterns inscribed on the wafer. Once the probe has been brought very close to the wafer, the next task is to establish a reliable and non-destructive contact of the probe with the wafer. A problem that needs to be addressed in any vision system is that of real-time response. Consequently, it is therefore necessary to consider some means of speeding up the algorithms that are required. Research was carried out towards achieving the above objectives and the results are reported in the following chapters.

1.6 Organization of the Thesis

The thesis contains seven chapters. The experimental set-up and the details of its various parts are described in the Chapter 2. We have developed an interactive image processing package as part of this research project. This is also described in Chapter 2. Next, various low level algorithms used in preparing the image data for further analysis are explained in Chapter 3. Some special purpose algorithms designed for our experimental set-up are also explained in this chapter. For estimating the distance of the probe to the wafer, we have developed a new technique for measuring depth using the sharpness of the edges in the images. The theoretical formulation of this method and its experimental verification are described in Chapter 4. The ultimate objective of the work is to guide a probe for good contact with a wafer. It is found that the probe starts sliding after touching the surface and some sliding is permissible without risk of damage to the wafer. This is measured using variance of the intensity values and enables us to detect touch. Analytical expressions for the variance of intensity values are derived for a general scenario in Chapter 5. The relationships in these expressions are verified using experimental data, and the procedure is explained in Chapter 5. For speeding up the system response, parallel low level vision algorithms are implemented on a multiprocessor and the results are described in Chapter 6.

The automation of a complete wafer probing operation is complex and involves several interdisciplinary areas. We have addressed one aspect of the operation, i.e. probing using computer vision. Further work to completely automate such a system is discussed in Chapter 7.

Chapter 2
Experimental Set-up

2.1 Introduction

A step proven most effective in shortening the lead time between the conceptual stage and actual productiun of VLSI chips is laboratory automation. A system capable of automatically extracting parameters for analysis and providing the design engineer with instant feedback is essential [25]. Wafer probing is therefore important in semiconductor parameter analysis. Several probe systems available on the market are manufactured by companies such as Wentworth Laboratories, Signatone Corporation, Alessi Industries, Micromanipulator Company and Rucker & Kolls. These systems are made for extreme precision and ruggedness for use in small-geometry probing. Several consist of high-power optics such as the Bausch & Lomb "Microzoom" microscope with magnification in the range of 25 to 2000. Wentworth Inc. and Signatone Corporation probe stations are of similar configuration. Signatone Corporation has developed a probing station with a built-in controller for micropositioning. They also have computer software for repeated operations on the die. The probing is manual in all these systems and to our knowledge no automation of wafer probing currently exists. The need for automation and its effect on yield benefits were discussed in Chapter 1. We shall now describe the experimental set-up that we designed to meet this objective.

Our set-up consists of a Wentworth Test Station (probe platform, probes, wafer chuck and a three-lens microscope [19]—[21]) and is shown in Figure 2.1. A camera is mounted on top of the test station, and an image can be captured through a custom-made image grabber. The images are preprocessed by an HP 9000 (series 300) computer where they are displayed and/or transferred to a SUN 3/60 workstation for analysis, processing and storage. The processed images can later be used to provide information to the motion controller for direct placement of the probes on the testing pads. This feedback is achievable through the implementation of several layers of vision algorithms.

So far, we have developed a working system for image capturing, display, and processing, an effective method of determining the probe distance from a contact point, and the detection of contact. In this chapter, we shall describe the modules of the experimental set-up, which we developed, and their operation. The camera is fixed to the microscope using an extender tube and an adapter. The description of this optical arrangement is given in Section 2.3. Section 2.4 describes a digital circuit designed for interfacing the handshaking signals of a GPIO bus through which data is collected by a DMA controller. Section 2.5 describes a program written to drive the hardware interface and collect and format the data. The data is transferred to a SUN 3/60 for further processing. The protocol for the data transfer is described in Appendix B. Section 2.7 describes an image processing package developed in our lab using Sun Windows for display and manipulation of images.

All the probes we use are presently controlled manually. Recently, we acquired a computer controlled probe and we are in the process of developing a motion controller for it. Even so, the present set-up is not suitable for real-time operation. It has been developed on a limited budget. Our prime objective was the proof of concept, i.e. our ability to accurately determine the position of the tip of the probe. We hope to achieve real-time operation by using a multiprocessor, and replacing the DMA controller by a high-speed image grabber with dedicated image processing hardware. Some distributed low level vision algorithms developed for a particular multiprocessor are explained in Chapter 6.

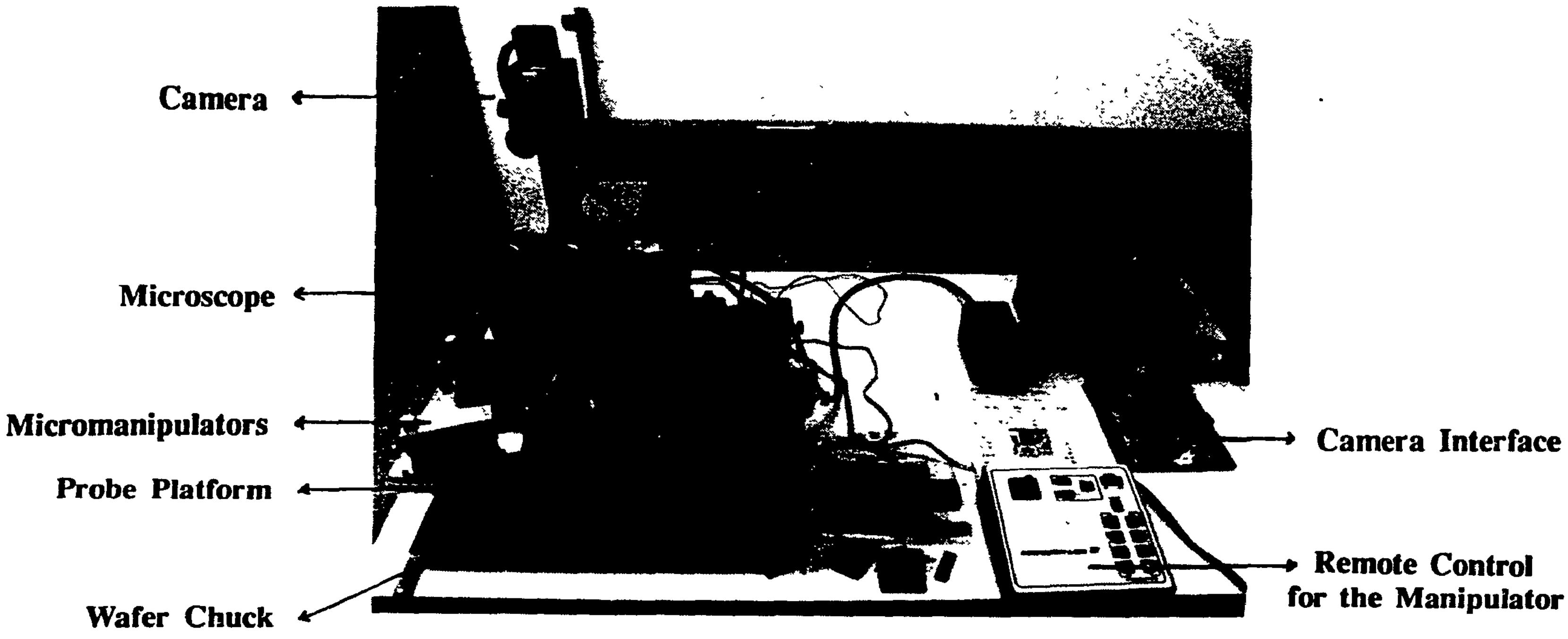


Figure 2.1 Arrangement of various equipment in our experimental set-up.

2.2 Block Diagram of the Set-up

The block diagram in Figure 2.2 shows the micromanipulator system under development. We will now describe the various parts of this system.

2.2.1 Wafer Chuck

The wafer chuck is used to load and unload the wafers under test. This unit has four degrees of freedom, three translational along the three axis of the X-Y-Z system of coordinates and one rotational about the Z-axis. The chuck has a wafer probing capability of 10 cm for the X-Y stage and 15 cm for Z stage. The micrometer resolution for these movements is 30 threads per cm. For mounting the wafer under test, the chuck is provided with two kinds of adapters. The first contains a base suitable for inserting an IC and is useful for testing ICs already packaged. The second contains a mechanical frame for mounting a wafer consisting of several dies.

2.2.2 Probe Platform

The probe platform shown in Figure 2.1 can take upto twelve magnetically-based micromanipulators which are manually controlled. Each micromanipulator has three degrees of freedom in the X, Y and Z axes. Initially the X and Y micrometers on the probe are used for aligning the probe tip with the pad. Next, the Z drive is applied for lowering the probe tip onto the target pad (one revolution of the micrometer corresponds to a vertical motion of 6.5 microns). Touching the probe tip to the wafer surface is a very sensitive operation. Hence each probe tip is spring loaded, causing it to slide horizontally after touching the wafer. This has become a useful feature for us in detecting contact between a probe and a pad as it will be explained in Chapter 5.

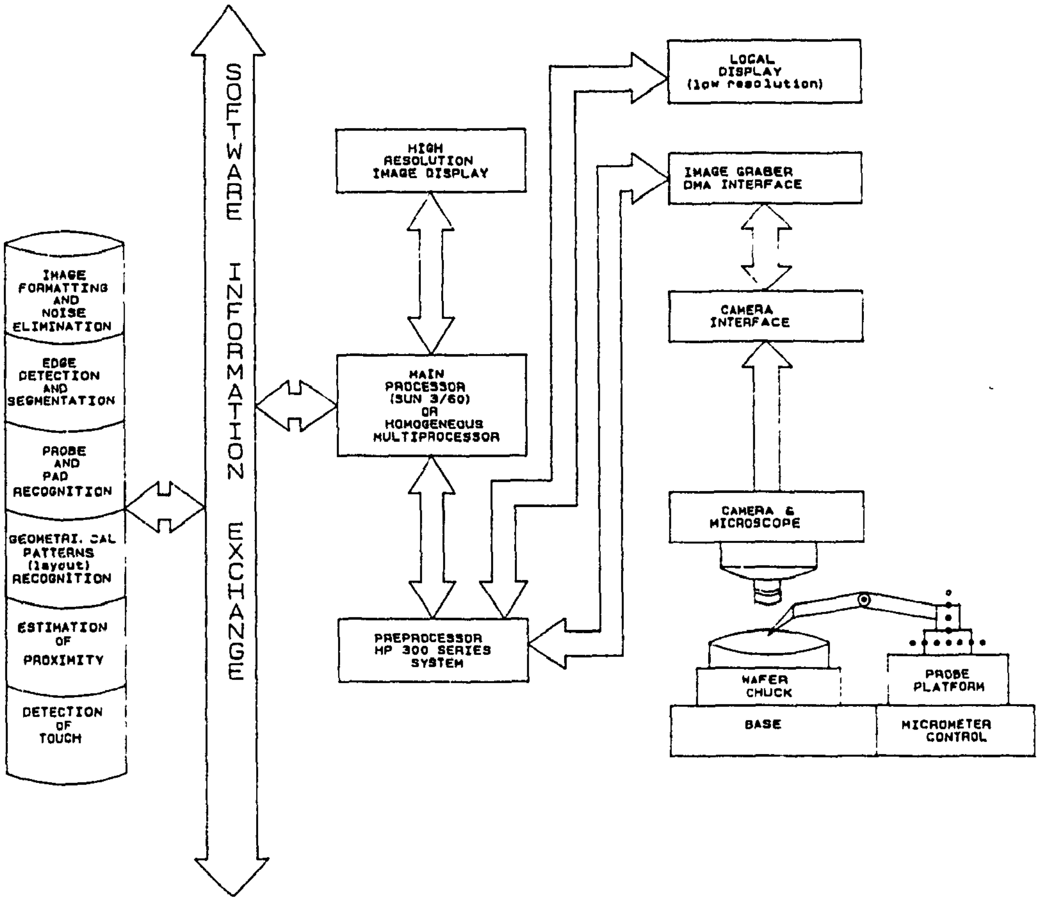


IMAGE FORMATTING AND NOISE ELIMINATION
EDGE DETECTION AND SEGMENTATION
PROBE AND PAD RECOGNITION
GEOMETRI. CAL PATTERNS (layout) RECOGNITION
ESTIMATION OF PROXIMITY
DETECTION OF TOUCH
SOFTWARE INFORMATION EXCHANGE
HIGH RESOLUTION IMAGE DISPLAY
MAIN PROCESSOR (SUN 3/60) OR HOMOGENEOUS MULTIPROCESSOR
PREPROCESSOR HP 300 SERIES SYSTEM
LOCAL DISPLAY (low resolution)
IMAGE GRABER DMA INTERFACE
CAMERA INTERFACE
CAMERA & MICROSCOPE
WAFER CHUCK
PROBE PLATFORM
BASE
MICROMETER CONTROL

Figure 2.2 Block diagram consisting of various modules in our computer vision system.

2.2.3 Microscope and Camera

In this application, we need to manipulate geometries in the order of a few microns. So a microscope of very high magnification is required. We use the MicroZoom® Microscope supplied by Bauch and Lomb Inc. A camera (manufactured by EG&G Reticon Inc. model MC9256) is fitted on top of the microscope. It consists of a 256x256 photodiode array with 64 grey levels. The camera outputs are differential RS422A signals transmitted on a 50 pin cable.

2.2.4 Image Grabber and Preprocessor

An important requirement in a computer vision system is high speed data collection. This is generally done by a hardware module called a "frame grabber" or "image grabber". This module uses a high speed DMA controller to collect data. Since we already have the HP 98620 Direct Memory Access controller in the HP 9000 computer, we developed a custom-made image grabber by adding extra hardware circuitry and a software driver. Information from the video data formatter is collected by the DMA controller and stored in an HP 9000 series 300 computer. This computer is based on an MC68020 microprcessor with 1 Mbyte of RAM memory and has a 20M HP9133 hard disk.

2.2.5 Low Resolution Display

We have a low resolution (16 grey levels) video monitor attached to the HP 9000 computer. During the set-up phase, it is not necessary to store images or transfer them to the main computer. So, we use the video monitor to preset controls on the camera, microscope and light source. We choose various options of the set-up, e.g., the digital threshold of the camera, the magnification required in the microscope, and the power of the light source. By using the local monitor, we can display a grey scale histogram and choose the camera threshold which represents the range of the video to be digitized. This feature is useful for expanding or compressing resolution over a portion of the video signal. This operation is called histogram equalization [26]. Since this is a hardware feature of our camera, we can enhance the image during the set-up phase and thereby reduce the software overhead. Another control preset uses the histogram to adjust the power supplied to the lamp affecting the intensity level of the light source. This is set according to the mean value of the histogram. All the above settings are done before a test-phase of the wafer. During the test-phase all images are transferred to the SUN 3/60 [27].

2.2.6 Vision Algorithms

We have developed a set of low-level vision algorithms which include smoothing, thresholding, edge detection, etc. for processing the images of the VLSI wafers. Further details are given in Chapter 3. We have devised a method for guiding the probe so as to satisfy our primary objectives of safe and reliable probing. We have used a two stage procedure for this purpose. The first estimates the proximity of the probe, and the second is concerned with the detection of contact between the probe and the wafer. Details of these two stages are described in Chapters 4 and 5.

2.3 Optical Set-up

The optical arrangement of the existing Wentworth test station is tailored to suit our imaging requirements (see Figure 2.3). A Halogen cycle lamp with white light is used for illumination. The light source is controlled by a diffuser filter and an aperture control. There is also a three-stage control for the power supplied to the lamp, so three different intensity levels can be chosen.

We have replaced the existing microscope head in the Wentworth test station with a tri-ocular head having a secondary viewing port. This tri-ocular head permits viewing through the eyepieces and also imaging through the secondary viewing port. The camera

Microscope Lenses	Numerical Aperture	Magnification	Field of view (mm)	Working Distance (mm)
Objective 1	0.042	22.5x to 45x	8.76	29.5
Objective 2	0.15	80x to 160x	2.54	19.9
Objective 3	0.31	250x to 500x	0.8	12.9

Table 2.1 Characteristics of the objectives of the microscope

is attached to the secondary port through an extender tube of length of 70 mm. The tube increases the length of the optical path and allows a focused image to be obtained at the camera. The distance from the image to the objective of the microscope depends on the magnification selected and the focal length of the optical set-up in the microscope which is the distance of the object from the lens. The optical characteristics of all three lenses are given in Table 2.1.

2.4 Hardware Interface to the Camera

The camera in our system consists of two units connected through a cable. One is the camera head (containing the image sensor) and the other is the video data formatter. Since the camera head is detachable from the power supply and data formatter, it is mounted on top of the microscope and connected to the data formatter through a 25–conductor cable. The camera head is operated with an internal master clock of 1MHz. The analog video signal from the camera head (containing line and frame synchronization) is fed to the A/D converter located in the video data formatter. The digital output (D1 — D7), Line Enable (LEN), Frame Enable (FEN), and clock signals (DCLK) are available on the 50 pin parallel connector with RS422A balanced lines. These signals are connected

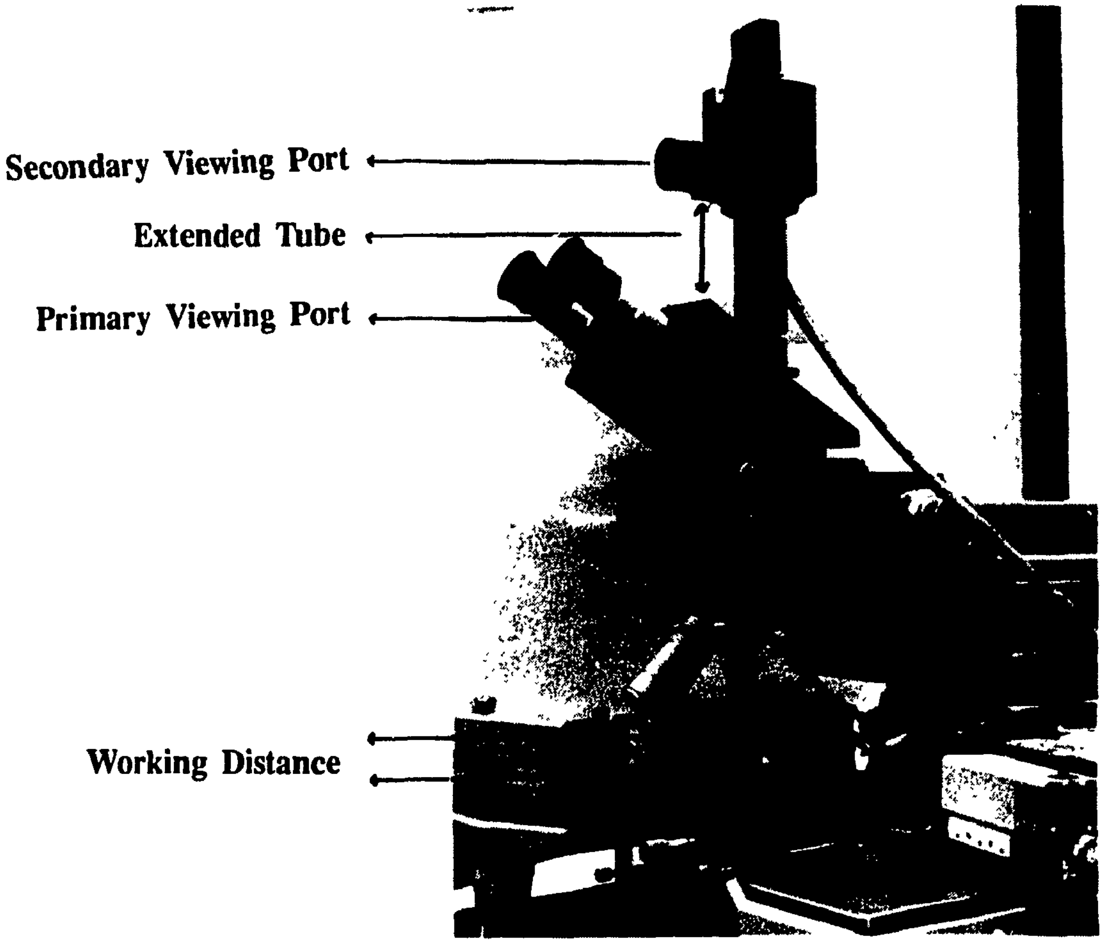


Figure 2.3 Optical arrangement showing various distances in our optical set-up.

to the GPIO (General Purpose Input and Output) bus of the HP 9000 computer. Since the handshaking signals of the GPIO bus are not compatible with the timing signals of the video data formatter, we designed an interface circuit. The circuit diagram is shown in Appendix C. The signals: PFLG, CTL0, and CTL1 on GPIO are used for the DMA transfer. The control signals (CTL0 and CTL1) are used to initiate the data transfer. The Peripheral Flag Line (PFLG) is used to synchronize the data transfer. A software driver was developed to start the data transfer, collect the data, and format the data. This software module is explained in the next section.

2.5 Software Driver for the Camera

A software driver, running on the HP9000 controls the hardware interface circuit shown in Appendix C. Initially the control line (CTL0) is used to set the second flip-flop to zero to inhibit the pulses going into the GPIO bus. Next, the control line (CTL1) is used to start the transfer. For every pulse on the PFLG line, the data (one pixel) on the input line is transferred to the buffer of the DMA Controller. This transfer ceases after one frame, i.e. 65536 pixels. This is made possible by gating the clock pulses by the frame enable line (FEN). The data is collected at the rate of 1Mbyte/sec. Then, the data in the buffer is formatted, displayed and transferred to the SUN 3/60 workstation. A complete listing of the software driver program is given in Appendix A.

2.6 Data Transfer between the HP 9000 Computer System and the SUN 3/60 Workstation

As mentioned earlier, the data in our system is processed in two stages. The first stage is the collection and formatting of the data and is carried out by the HP 9000. We used the HP 9000 to capture our data, since it provided us with a compatible and accessible interface to our camera. A SUN 3/60 on the other hand is used for its computational power and its good graphics environment for development of our application. The communication protocols of these two computers were studied [27] and a procedure was developed for reliable transfer of data between these computers at 19200 baud rate. This procedure is given in Appendix B.

2.7 The Image Processing Package

In Chapter 1, we explained the need for a soft, reliable and safe method of wafer probing. To this end, we have used visual feedback for manipulating images and have formulated a method for lowering the probe onto the wafer. All these methods are verified using data collected from the experimental set-up. During the course of this work, we developed an image processing package to facilitate the development of the various algorithms. This package was developed on the SUN 3/60 workstation and was written using the SUN Windows environment. The Image Processing Package (IPP) comprises the user interface and a set of functions that can be applied to selected images. The user interface offers a seta set of buttons which can be used to activate particular vision algorithms such as smoothing, thresholding, edge detection using the Roberts, Sobel and Laplacian operators, the Hough transform, and histogram generation. Examples of the menus are shown in Figures 2.4 and 2.5. The IPP also has the ability to select and display upto 8 raster files. Each of the displayed images can be individually zoomed in/out. Also additional buttons can be attached to the unallocated buttons to increase the functionality of the package (Figure 2.4). The procedure for including the user defined functions is given in [28].

In addition to the processing of images, we have added a feature, to convert ASCII files to raster format and vice-versa. The reason for this is that the uploaded data from the camera is available in ASCII format, but, SUN workstations support only the raster file structure. Also, the raster files are an efficient method for storage and retrieval. This general purpose package can be easily exported to other Sun Workstations such as the

SUN4 (Sparc) Workstation and has been found to be very useful in the development and applications of image processing algorithms.

2.8 Conclusions

Several manufacturers offer computer-aided parameter testing, inspection and wafer handling equipment. But not much automation has been done so far in the context of wafer probing. However, safe and reliable probing is an important operation in wafer testing. Mechanical probing is the most popular one among the different probing techniques available. It is reliable and useful for testing several parameters of the wafer. We have designed an experimental set-up. This incorporates a Wentworth Test Station with a modified optical arrangement. Our set-up uses vision feedback and complex image processing algorithms. With our experimental set-up, we are able to capture images and at the same time observe the wafer through the viewing port in the microscope. We have also developed a user-friendly image processing package. However, at present our image capturing, storage, and processing is slow and not suitable for real-time applications. Due to budget constraints we developed a custom-made frame grabber for image acquisition. The use of high-speed frame grabbers with built-in hardware functions for image processing environments would alleviate this difficulty. In the near future, it is intended to use a Homogeneous Multiprocessor for real-time operation. Speeding up processing time using this multiprocessor is explained in Chapter 6.

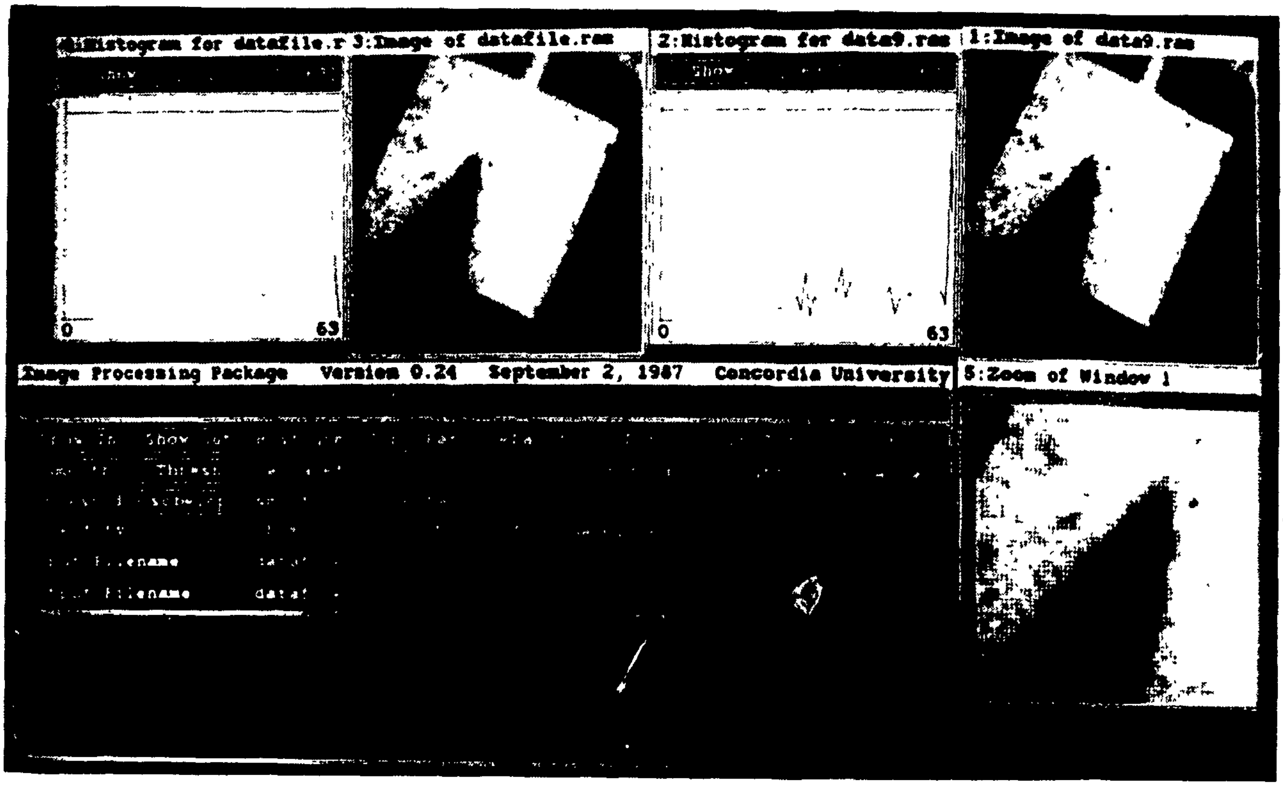
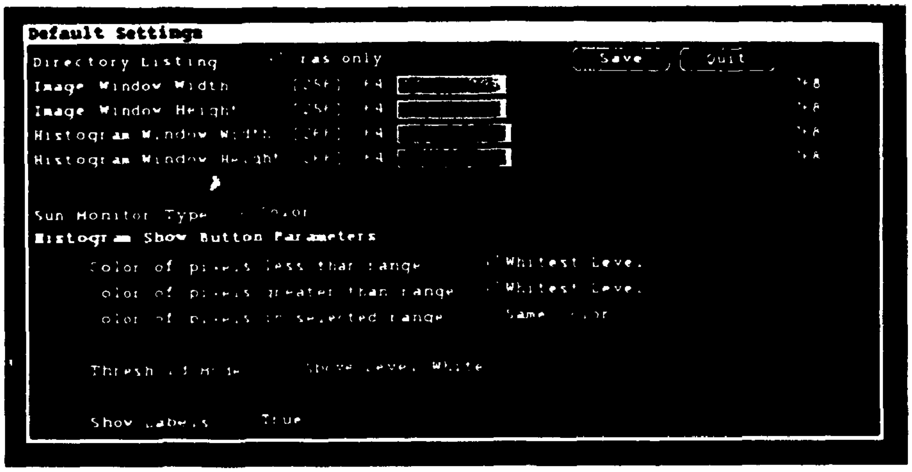

4:Histogram for datafile.r
3:Image of datafile.ras
2:Histogram for data9.ras
1:Image of data9.ras
Show
0
63
0
63
Image Processing Package Version 0.24 September 2, 1987 Concordia University
5:Zoom of Window 1
Show
Thresh
Filename
Filename dataf

Figure 2.4 Various functions available in the Image Processing Package (IPP).



```
Default Settings
Directory Listing          .ras only                Save   Quit
Image Window Width         [256]  64  [     ]  768
Image Window Height        [256]  64  [     ]  768
Histogram Window Width     [256]  64  [     ]  768
Histogram Window Height    [256]  64  [     ]  768

Sun Monitor Type     Color
Histogram Show Button Parameters
    Color of pixels less than range        Whitest Level
    Color of pixels greater than range     Whitest Level
    Color of pixels in selected range      Same Color

    Threshold Mode        Above Level, White

    Show Labels      True
```

Figure 2.5 Selection of various display formats using the Image Processing Package (IPP).

Chapter 3
Low Level Vision Algorithms for the Experimental Set-up

3.1 Introduction

The imaging process compresses much useful physical information into the gray level array. This array represents intensity values in the image. The initial task in understanding this data is to format it in a form suitable for display and manipulation. The way this is done depends on the various equipment used for processing the data. We introduced the details of formatting and displaying the image data in Chapter 2. There are three levels of processing in a computer vision system [23]; low-level processing, intermediate-level processing and high-level processing. Low-level processing is concerned with neighborhood operations and extraction of features using the intensity values of the image. This kind of processing is generally done at the front end of the computer vision system. In intermediate-level processing, an image is analyzed for three dimensional information (depth from focus, in our application). High-level processing is concerned with the symbolic interpretation of an image and the coordination of various low level modules.

We have used two levels of processing, low and intermediate levels in our application. In Chapters 4 and 5, we describe a method for detecting the depth of objects in an image. This method is used in measuring the distance of a probe from the wafer. In this chapter, we describe the low-level vision algorithms used in our system. These algorithms include image formatting, elimination of noise, segmentation of the image into regions, edge detection, and alignment of the probe and wafer. All these algorithms are described in Section 3.2.

During our experimentation we also noticed specific discrepancies in some experimental data. We found these to be mainly due to the arrangement of our set-up. To explain and resolve these discrepancies, we have developed some solutions using our practical experience. These are explained in Section 3.3. However, the main objective of this work is to guide the probe to its target pad. In this operation, the first task is to bring the image of the pad to the center of the scene. Wafer orientation is required for this purpose. We use the edge-map of the metal layer of the die and detect the orientation of the die using the Hough transform [29], which is described in the Section 3.2.4.

3.1.1 Existing Methods for Low-Level Vision

In this section we review the existing methods for low level image processing. We concentrate on operations such as smoothing and edge detection. These are well established techniques and are provided in many commercial vision systems. Since the performance of a computer vision system depends directly or indirectly on these operations, there has been a lot of research interest in improving the performance of the existing techniques.

The Smoothing operation is generally used either for filtering noise or as a preprocessing operation for edge detection. Some of the popular image smoothing techniques are [30]: i) Neighborhood averaging, ii) Gaussian smoothing, and iii) Lowpass filtering. All these techniques produce a blurring effect in the image. Neighborhood averaging is a spatial-domain technique for image smoothing. In this technique, each pixel is replaced by the average of the intensity values of the neighborhood. The degree of blurring produced by this method is proportional to the size of the neighborhood. Another technique is Gaussian smoothing, where the image is convolved with a Gaussian function with a space constant σ. This Gaussian function blurs the image effectively, wiping out all structures (image features) at scales much smaller than the space constant σ of the

Gaussian. This kind of smoothing is used as a first step for edge detection by the Marr and Hildreth edge operator [31]. The Gaussian distribution has the distinct characteristic of being smooth and localized in both the spatial and frequency domains and of being the unique distribution that is optimally localized in both domains. If the blurring is smooth in both the spatial and the frequency domains, then it is least likely to introduce any features that were not present in the original image [31].

Another common approach for smoothing is lowpass filtering. This is an established technique used for filtering noise in the image data. Sharp transitions (such as noise) in the gray levels of an image contribute heavily to the high-frequency content of its Fourier transform. It follows, therefore, that blurring can be achieved via the frequency domain by attenuating a specified range of high-frequency components in the Fourier transform of a given image [30]. Several lowpass digital filters are described in the signal processing literature where there are examples of Butterworth, exponential and trapezoidal filters [32]. Since we need to compute the Fourier and inverse Fourier transforms, these filters are computationally very expensive. Of all the above described methods, neighborhood averaging is simple and computationally inexpensive. Gaussian smoothing is used for detecting edges on different scales (for different sizes of image features). Lowpass filtering is used for removing various types of noise, where other methods are ineffective.

Another low level operation is edge detection. An edge in an image may be defined as a discontinuity or abrupt change in the gray levels. In general, images may contain a variety of edge sizes, some short and others long. Also, these edge segments may occur at any orientation. An efficient edge-detection procedure would necessarily be able to distinguish contrast at different angles [26]. Mask operators are the most popular ones for edge detection. The first significant operator was due to Roberts [26] who employed a simple 2x2 mask, a first-order difference operator, to enhance the edges of solids. The

Roberts operator detects both horizontal and vertical edges but is sensitive to noise and object surface irregularities. The Sobel operator, a second-order difference operator, [26] was designed to approximate the discrete gradient function in a selected orientation. This means, different masks are used on each pixel of the image for different orientations. Generally, the operator producing the maximum response represents the orientation of the edge. The Sobel operator is used in industrial inspection and is explained in [26]. A parallel implementation of this operator is explained in Chapter 6.

Instead of taking a set of directional masks as done by Roberts and Sobel, it is also possible to employ a scalar, isotropic (omni-directional) edge detector. A second-derivative operator, the Laplacian [26], is one of these. We have used this operator as described in Section 3.2.3. The Laplacian is a rotationally-symmetric operator which is useful because it treats image features in the same way, irrespective of their orientation [33]. A new theory of edge detection, based on finding the zero-crossings of the output of an even-derivative operator applied to the image, is described by Marr and Hildreth [31]. They applied the Laplacian operator after Gaussian smoothing to produce a rotationally symmetric edge operator. This edge-finding method is very popular and well experimented. Several researchers have used this operator for various applications and this is still an active area of research.

All the above described operators are either first-derivative or second-derivative and it is well known that derivatives emphasize high frequency noise. In fact the higher the order of the derivative, the more pronounced is the presence of high frequency noise [34]. Also, the noise-characteristic of an operator depends on its size. The larger the operator, the more it averages out random noise. However, it is also more likely to overlap several edges or corners simultaneously and thus degrade its resolution capability. So one has to be careful in selecting the proper edge detector since the edge-information is going to influence further analysis (e.g. object recognition, depth determination) of the image.

3.1.2 The Hough Transform

The Hough transform [29], as suggested originally, is a method for detecting straight-line segments in an input image. This concept is extended to include circles and ellipses [35]. Recently, the Hough transform has been used for the estimation of scale, rotation, and translation of an input image with respect to a reference image [36].

In this transform, we apply a coordinate transformation such that all points belonging to a given line in the edge-map are mapped to a single location in the transformed space [26]. To illustrate the approach, let us consider an image plane *f(x,y)* as shown in Figure 3.1. Given a point *(x',y')* on a line AB, an infinite number of lines can be drawn to pass through it. Let us suppose that a line KK' is passing through that point. It can be parametrized by two parameters ρ and θ, where ρ is the normal distance of the straight line from the origin and θ is the angle of the normal to line KK' passes through the origin. Thus any point *(x',y')* on line AB in *f(x,y)* is mapped to a sinusoidal curve in Hough space $H(\theta,\rho)$ given by

$$\rho = x'cos\theta + y'sin\theta \tag{3.1.1}$$

This sinusoidal curve gives the ρ and θ parameters of all the straight lines passing through the point *(x',y')*. If we perform the identical transformation for all the points on the line AB in the image plane *f(x,y)*, then each point maps to a different sinusoidal curve given by the above equation. This is shown in Figure 3.2. All these curves intersect at a point (θ',ρ') in the Hough space and this point defines the ρ and θ parameters for the straight line AB as shown in Figure 3.2.

We now consider a discretized space (θ,ρ), forming an array of discrete (θ,ρ) pairs. As the transform for a given value *(x,y)* of each image point is plotted in (θ,ρ) space, each

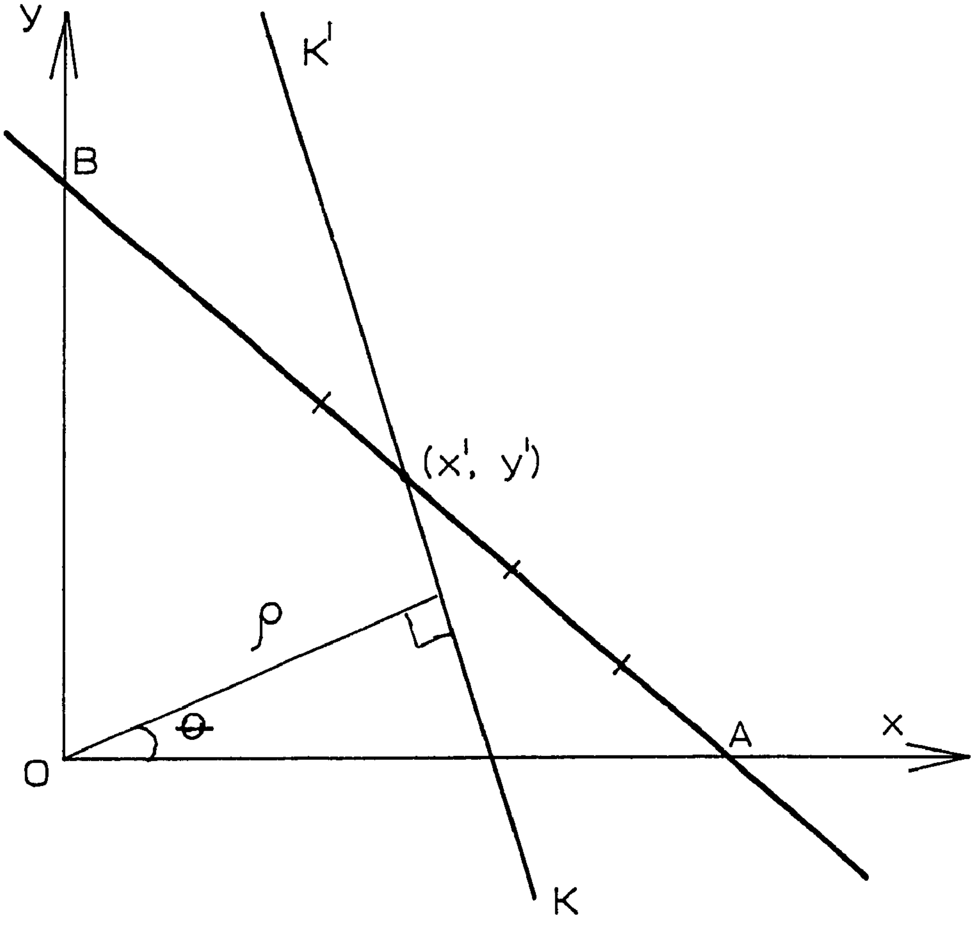


Figure 3.1 Parametrization of the line KK′ in the image plane *f(x,y)*. Points on the line AB are mapped to sinusoidal curves in the Hough transform plane (Figure 3.2).

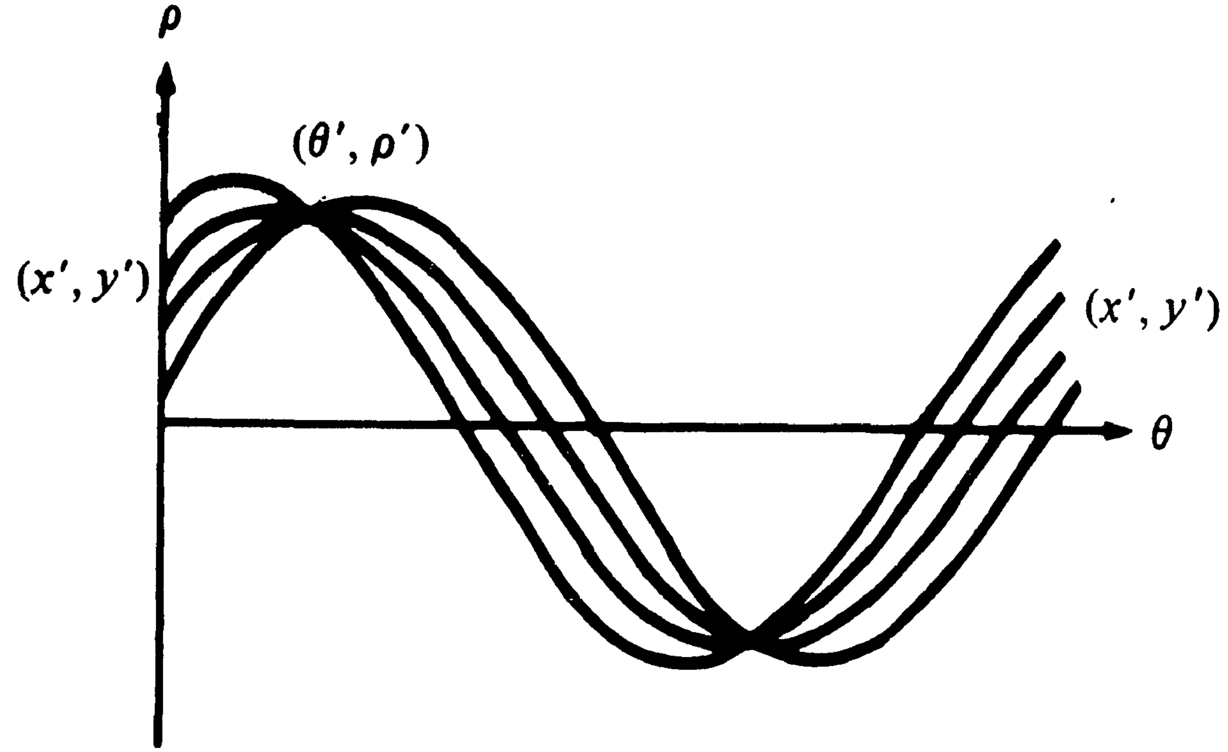


Figure 3.2 The Hough transform space representation of four points on the line AB shown in Figure 3.1. (θ',ρ') gives the parameters of the line through the points [26].

array element is incremented whenever a sinusoidal curve passes through it. Thus, several (θ,ρ) points are "voted" for each point (x,y) forming an accumulated array $H(\theta,\rho)$. After all the points in the $f(x,y)$ plane have been transformed, the array elements ($H(\theta,\rho)$) with high counts are determined by thresholding. These elements represent the straight lines in the image plane. In our application, the array elements having high counts are used for calculating the orientation of the wafer. The results are described in Section 3.2.4.

3.2 Generic Low Level Vision Algorithms used in our System

In this section we describe low level vision algorithms which are frequently used in our system. These algorithms are: smoothing the raw image from the camera, identifying various regions in a scene, generating an edge-map and determining the wafer orientation using such an edge map. These algorithms form a stepping stone for estimation of the proximity of the probe to the wafer (Chapter 4), and detection of contact of the probe with the wafer (Chapter 5). Furthermore, parallel implementations of these algorithms are discussed in Chapter 6.

3.2.1 Smoothing

Smoothing is the first preprocessing step applied to the raw image of the camera. In our experimental set-up, the camera is mounted on top of the microscope head and along the optical axis of the light source. Therefore, the viewing angle is the same as the incident angle of the illumination. Due to the bright through-the-lens illumination of the VLSI metal patterns, the image consists of artifacts and spurious effects. In order to smooth these effects and also to reduce the overall noise level, it is required to smooth the gray level variation of the image. Our primary purpose of smoothing the image is to obtain three distinct regions (probe, metal and diffusion layer) from the histogram of the gray levels and then accomplish separation of the regions of the image. We accomplish smoothing by simple neighborhood averaging. This can be described as follows. Given an *NxN* image, *f(x,y)*, the smoothed image, *g(x,y)*, is obtained by averaging the gray level values of the pixels of the original image contained in a predetermined neighborhood *S(x,y)* of a pixel *(x,y)*

$$g(x,y) = \frac{1}{M} \sum_{n,m \in S(x,y)} f(n,m) \qquad (3.2.1)$$

where *M* is the total number of points in the set *S(x,y)*. We have used a 3x3 neighborhood in this work.

The image from the camera is shown in Figure 3.3. The histograms of the raw and smoothed images are shown in Figures 3.4 and 3.5. The histogram for the smoothed image consistently shows three distinct regions corresponding to metal, probe and background. The task of separating different regions is accomplished by thresholding the image using the gray values which identify the three regions in the histogram. The procedure for thresholding is explained next.

3.2.2 Identification of Probes, Pads, and VLSI patterns

In our application, the images generally consist of regions containing highly regular geometrical patterns which correspond to the VLSI component geometry, as well as triangular and much darker regions which correspond primarily to the probe. The metal and diffusion layers appear gray, the darkest being in the region corresponding to the probe.

The averaging process described in the previous section tends to smooth out the image histogram which now exhibits three easily discernible regions corresponding to the probe, background and metal. Thresholding is then used to segment the patterns of interest corresponding to these regions. The procedure for thresholding is explained as follows. Let us consider Figure 3.5 with different peaks in intensity at I_p, I_b, I_m,

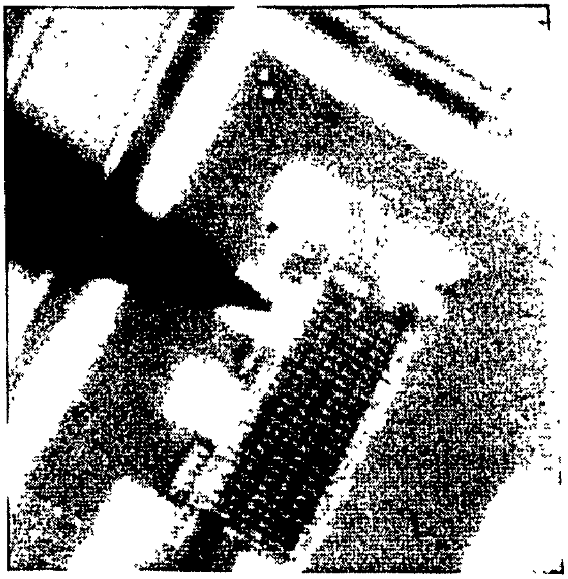


Figure 3.3 Image of the probe touching the pad.

corresponding to the probe, background, and metal layer. These values correspond to the lamp setting and can be determined from the histogram of the smoothed image*. Therefore, if we know I_θ, where θ denotes the region of interest, we can convert any image *I(x,y)* to an equivalent binary image, such that,

$$B(x,y) = 1 \quad for\ I(x,y) \leq I_\theta$$
$$B(x,y) = 0 \quad for\ I(x,y) > I_\theta \tag{3.2.2}$$

where *B(x,y)* represents the thresholded image of the region with intensity value I_θ.

In our research, we are primarily interested in the metal layer, since this is the most visible one and also since both the test pads and I/O (input/output) pads are implemented in this layer. Also, the processing steps outlined above rely heavily on the fact that the images obtained correspond to essentially two-dimensional-objects of high reflexivity.

* For example, sample parameters for Figure 3.5 are: I_p=15, I_b=30 and I_m=45.

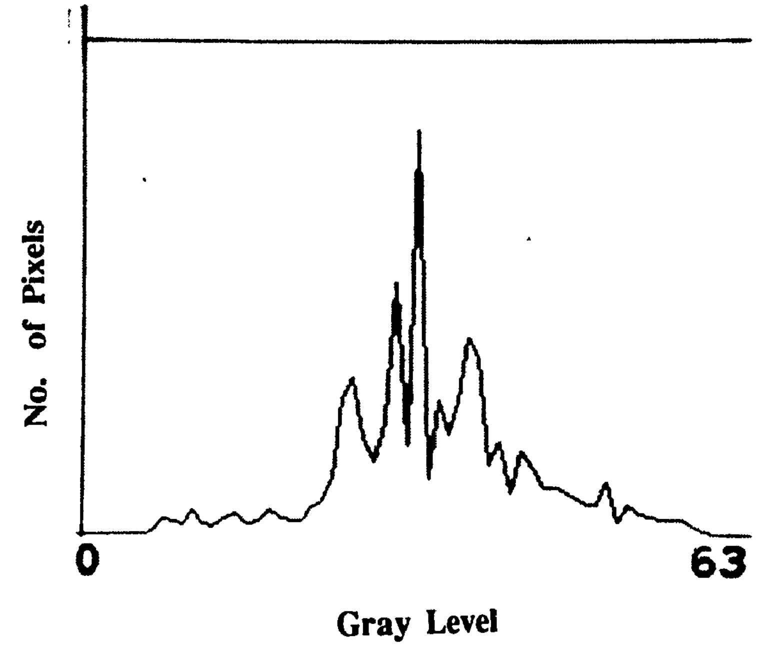


Figure 3.4 Histogram of Figure 3.3.

The only "dark" region corresponds to the probe (which is out of focus and of lower refractive index). Simple thresholding therefore seems to be adequate in distinguishing patterns of interest.

3.2.3 Edge Detection using Zero-crossings

After separating the different regions using thresholding, we can find the boundaries of each region by edge detection. The Laplacian operator is used as an edge detector. The Laplacian operator we have used is as follows:

$$\begin{matrix} -1 & -1 & -1 \\ -1 & 8 & -1 \\ -1 & -1 & -1 \end{matrix} \qquad (3.2.3)$$

By convolving the thresholded image with the above operator, we obtain second derivatives of large magnitudes around the edges. These values are positive and negative corresponding to the intensity variations. A neighborhood of 3x3 is considered in the convolved image and a pixel is considered a zero-crossing, whenever there is a change of sign in the convolved image. The set of such zero-crossings represents the boundaries (edge-map) of each region. Figures 3.6 and 3.7 show the zero-crossings for the probe and the metal layer in Figure 3.3.

3.2.4 Calculation of Wafer Orientation using the Hough Transform

Wafer orientation information is useful in aligning the probe with the pad and for further recognition of the elements of the die. The edge-map of the metal layer is obtained using the procedure described in Section 3.2.3. The Hough transform of such an edge-map is calculated. As shown in Figure 3.8, the patterning on the ICs is accomplished

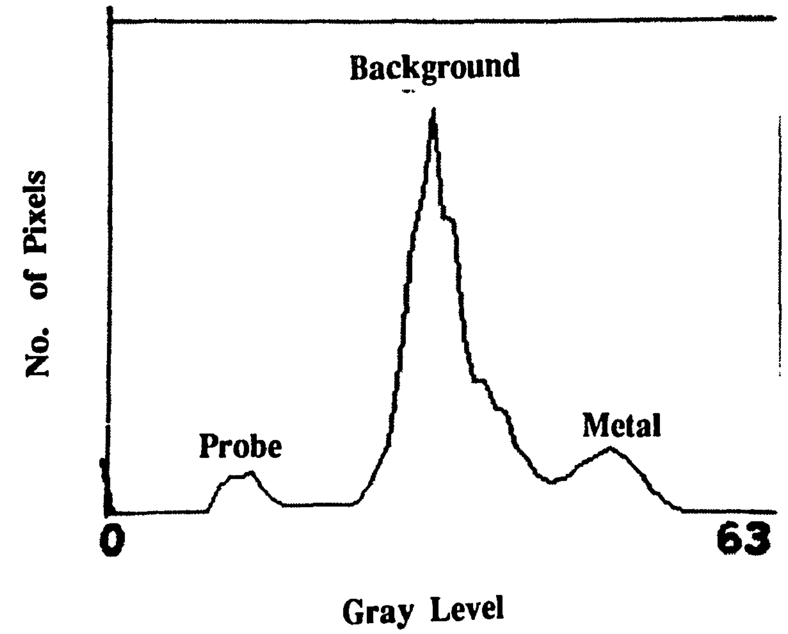


Figure 3.5 Histogram of Figure 3.3 after smoothing.

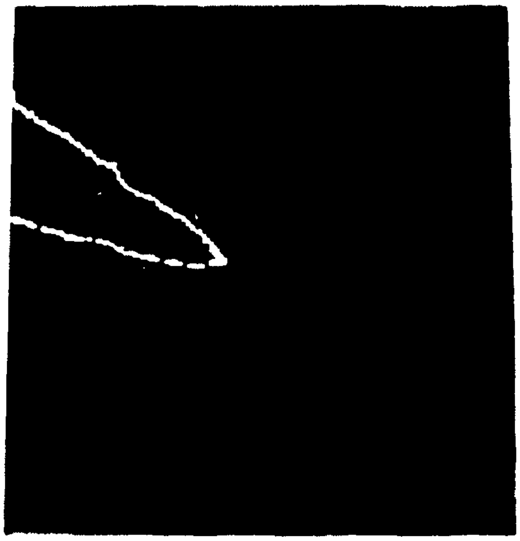


Figure 3.6 Segmentation of Probe in Figure 3.3.

by using "Manhattan distances". Hence the prevailing patterns in the edge-map are collections of short vertical and horizontal line segments. Therefore, in the Hough space (a plot of θ versus ρ as described in Section 3.1.2) two peaks at a distance of 90° are observed. These peaks can be seen in Figure 3.8 and correspond precisely to the groups of vertical and horizontal line segments in Figure 3.7. The value of θ corresponding to the peaks in this figure represents the orientation of the wafer with reference to cross-wires in the camera (an example of 62^0 in this figure).

3.3 Special Purpose Algorithms used in our System

The requirement to build a computer vision system in an unstructured environment is not uncommon. Our application is one such example. Some components in our experimental set-up cannot be changed easily, e.g. the optical arrangement of the microscope. We encountered some difficulties due to reflections from the wafer and viewing collinearly with the illumination. We observed periodic noise while using a small aperture light source with high lens magnification. Sometimes, we also observed misalignment of images as the probe approached the wafer, due to the vibration of the wafer chuck. Since these effects occur only on certain occasions, and we can not easily change the set-up, we have developed procedures to counteract them. These are explained in the following sections.

3.3.1 Elimination of Periodic Noise

We found that some raw images taken from the camera contained periodic noise. This can be seen in Figure 3.9. We observed this using some combinations of the parameters of the apparatus, e.g. high numerical aperture of the lens, small aperture of the light source, and also certain optical filters at the light source. Since we need to use

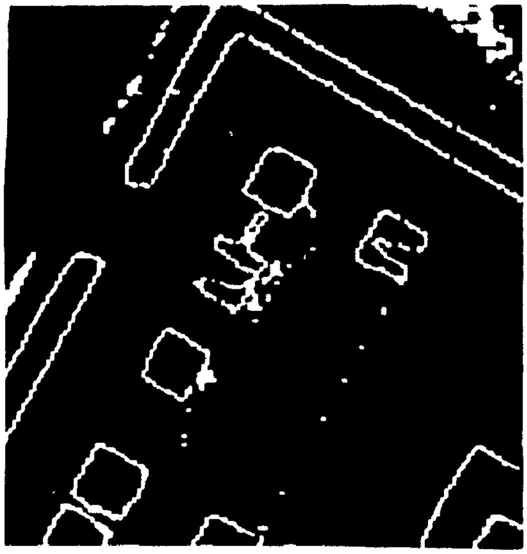


Figure 3.7 Edge map of metal region in Figure 3.3.

these components in a probing operation, we have developed a technique for eliminating this periodic noise. This noise is in the form of parallel lines in the image. This is shown in Figure 3.9. Since the noise is periodic, the frequency components of the noise will be harmonic. Hence, in order to isolate this noise, it is convenient to study the frequency components of the image data. Therefore, an FFT of the image is taken. This shows that frequency components of the parallel lines lie on the vertical axis [37]. An inverse FFT is taken after masking these frequency components. Figures 3.9 and 3.10 show the original and filtered images. This method of eliminating noise is computationally very expensive and is not useful in real-time operation. However, this method was used to create a library of "clean images" for use in studying the algorithms described in Chapters 4 and 5.

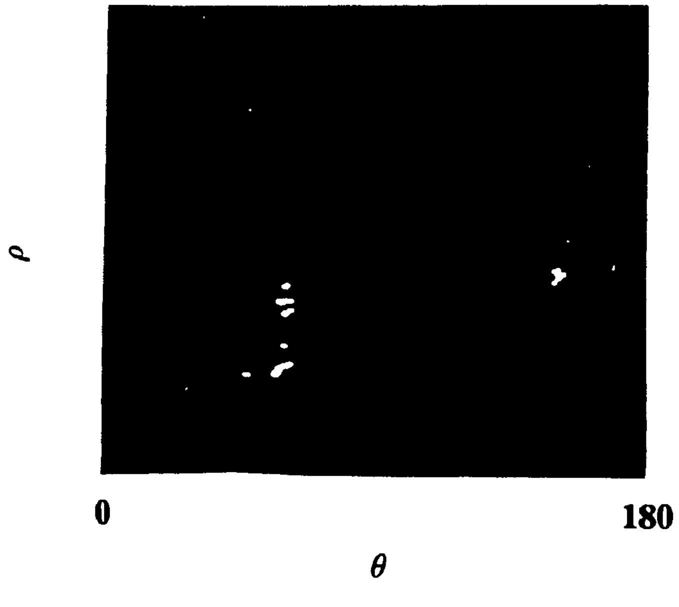


Figure 3.8 Hough picture of Figure 3.7.

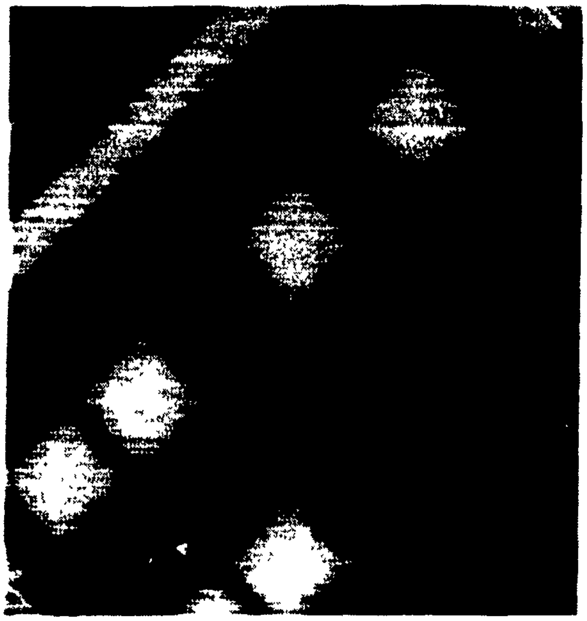


Figure 3.9 Raw image from the camera.

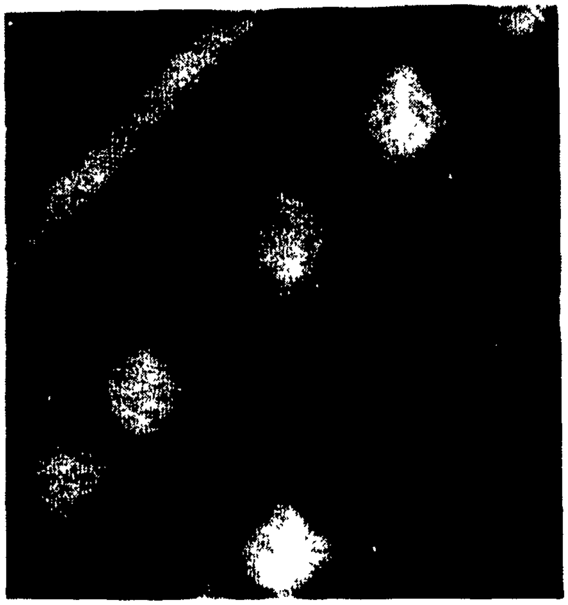


Figure 3.10 Image after filtering periodic noise.

3.3.2 Alignment of Images

In our experimental set-up, the wafer remains stationary, while the probe is approaching it. We capture the images of the focused wafer at different instants until the probe touches the wafer. In this operation, only the z-drive of the probe is used for lowering the probe. However, a small displacement of the wafer (equal to a fraction of a micron) with respect to its initial position *(x,y)*, is observed. This results in misalignment of consecutive images of a given section of the wafer. This is due to vibration of the wafer chuck, probe platform, etc. In this section, we describe an algorithm for the alignment of the images of the wafer captured at different instants. For instance, the images are captured at different instants of the probe movement, starting at time t_0, when the image I_0 represents the pad, and ending at time t_n, when the image I_n represents the contact of the probe with the pad. A sequence of these images is shown in Figure 3.11. We notice misalignment in these images by close observation. Since we use the intermediate images for estimating the proximity of the probe to the pad, the preliminary task is to align these images, i.e. to calculate the shift of the image I_i with respect to I_0.

The shift described in the above paragraph is calculated using the algorithm given in Appendix D. A brief description of this algorithm is given below. Each pixel in the image I_i at time t_i is considered to be shifted by a few rows and columns away from the original position of the image I_0 at time t_0. Certain parts of these images, namely windows W_i and W_0 of images I_i and I_0 (e.g. top left quarter of the image) are selected for calculation of this shift. Subsequently a search is made to determine the number of rows and columns shifted by sliding the window W_i on I_0. In other words, an error value

$$E(x,y) = \sum_{i,j \in W_i} \| I_i(i+x, j+y) - I_0(i,j) \| \qquad (3.3.1)$$

is calculated for all the values of x and y (maximum of ±16 rows and columns), where x and y represent the row and column index for the slide. When $E(x,y)$ is minimum,

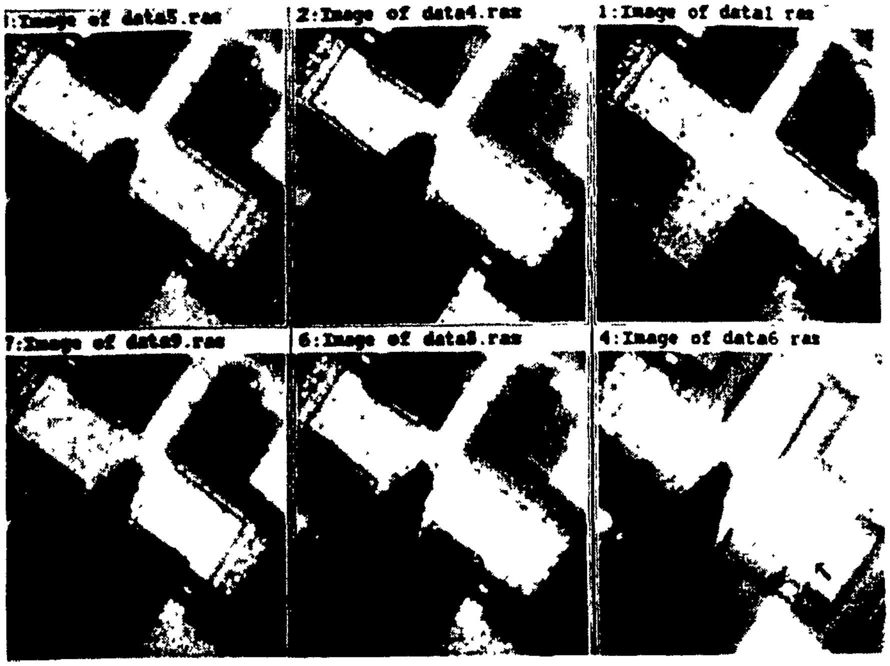


Figure 3.11 A sequence of displaced images while the probe is approaching the pad.

the corresponding index values of this slide, the *row_shift* and *column_shift* are noted. These values represent the row and column shift of the image I_i with respect to image I_0. Finally, all the rows and columns of the image I_i are shifted by an amount equal to *row_shift* and *column_shift.*

The parameters for the length and breadth of this search are set as *search_length,* and *search_breadth.* The location of the window W_i is set by the parameter *search_origin.*

3.3.3 Elimination of Traces of the Background

All the images described so far contain the metal layer of the wafer and the image of the probe. We separate the pixels corresponding to the probe and use these pixel values in determining the proximity of the probe to the wafer. The method used for estimating proximity using these pixels is described in Chapter 5. For separating the probe pixels from the background, we use the thresholding technique described in Section 3.2.2. However, if there is a misalignment between the background and the probe, we cannot completely separate the probe pixels using this thresholding technique. Although we use the alignment procedure described in the previous section, the image of the probe sometimes contains traces of the background patterns. This is shown in Figure 3.12. We have developed an algorithm (given in Appendix E) to eliminate such traces. In this algorithm, the probe is considered as a gray continuum, so all the isolated pixels are considered as traces of the background. This process helps in providing a clean probe image for further processing. More specifically, number of pixels in the *neighborhood* (16 x 16) of each pixel are counted. If more than 40% (empirical) of the *neighborhood* (i.e. 0.4 x 256 =102 pixels) is present , then this pixel is considered as a part of the probe. Otherwise, it is considered as the background trace and deleted from the image. The size of the *neighborhood* is set by experimental observations. By choosing a large value for the size of the *neighborhood*, even dense background traces can be eliminated.

The above procedure was applied to the image in Figure 3.12. and the result is shown in Figure 3.13. The last two algorithms described in this section were used in sequence and found to be very useful in rectifying misaligned images.

3.4 Conclusions

There are broadly two kinds of processing in our computer vision system. The first involves cleaning of the image and preparing the data for further processing. The second uses the data from the first and interprets 3–dimensional information in the image under consideration. This includes feature detection, object recognition and image analysis. In our application, this stage contains depth perception and detection of touch. For successful completion of a given task, each stage of processing is equally important. In this chapter, we have considered the first stage of processing, i.e., low level processing. We first described the generic low level algorithms used in our system. For obtaining edge-maps from high contrast images the following sequence of operations is widely used: smoothing, thresholding, and the Laplacian mask operation. The histogram of the smoothed image (Figure 3.5) contains three peaks which correspond to the three regions in the image. Hence, the thresholding operation works very well on smoothed images. This is because the images of VLSI wafers including the probe have very high contrast. The Laplacian works well after thresholding because we get a step edge by thresholding. It was found that a window size of 3x3 produced satisfactory results for edge detection. Such an edge-map of Figure 3.3 is shown in Figure 3.7. A small size window is acceptable because the image being thresholded has sharp edges.

Although the data in our application can be easily separated into different regions, we face several other problems due to physical constraints in our experimental set-up. We have unconstrained illumination, high intensity of light, specularity due to the metal layer, noise due to multiple layers of the VLSI patterns, shadow cast by the probe, vibration

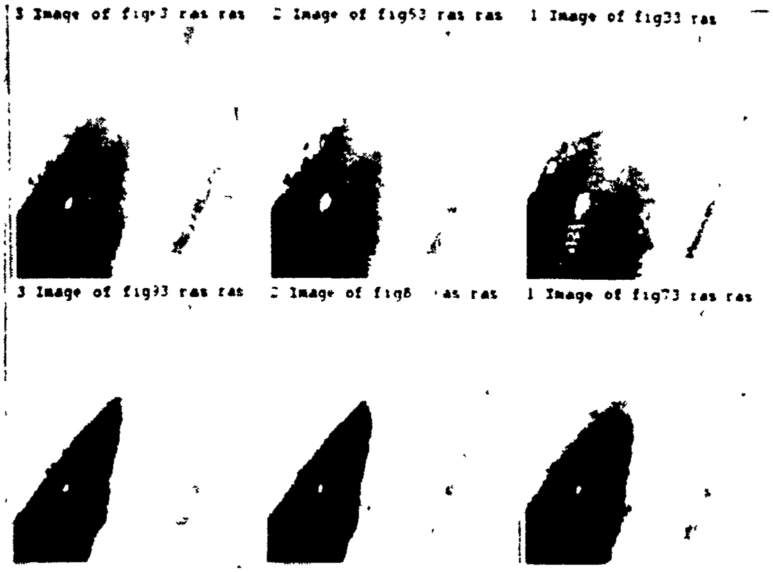


Figure 3.12 Images of the probe containing traces of the background patterns.

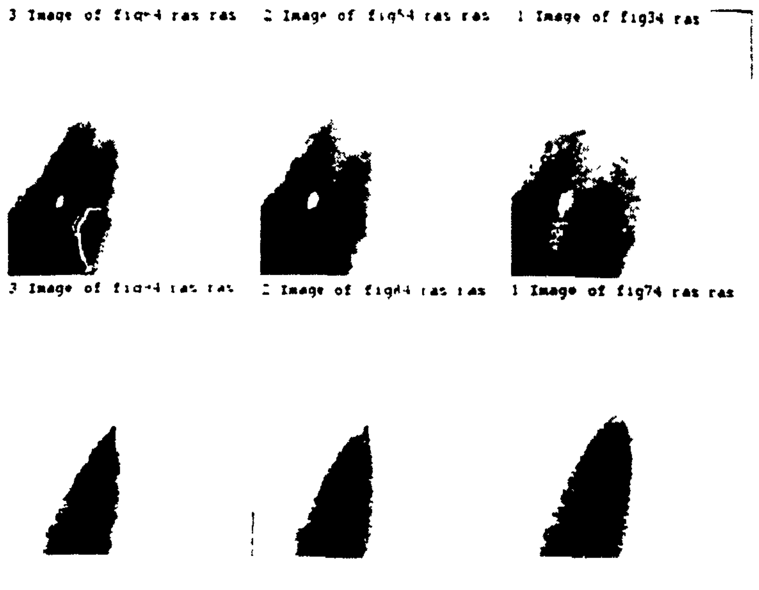


Figure 3.13 Images of the probe after eliminating traces of the background in Figure 3.12.

of the wafer chuck, etc. In addition, we have image resolution of only 6 bits per pixel. We used some special purpose algorithms to overcome the above mentioned difficulties. Elimination of periodic noise was discussed in Section 3.3.1. This noise is intermittent and is due to the different components of our optical set-up. We used an FFT technique to obtain up a "clean image". For a production environment, a special study of the image acquisition and optical equipment must be performed to ensure that the equipment used does not give rise to periodic noise. We have also encountered some problems due to vibration of the equipment. This results in misregistration of the images. Presently we have no means to provide a vibration free environment. Hence, we have used algorithms based on heuristics to resolve this difficulties.

Chapter 4
Estimation of Proximity using Blurring around Edges

4.1 Introduction

Computer vision provides important sensory information in many robotic tasks. Measurement of depth of an object in a scene is very useful information for the success of these tasks. This measurement is required for obstacle avoidance and navigation, pose determination, inspection, manipulation, and assembly of objects [38]. The methods for distance measurement can be categorized into two subclasses. The first is monocular, relying on a single-viewpoint. The second is nonmonocular requiring two or more images from cameras displaced laterally along a defined baseline. The depth of an object is calculated by measuring the displacement of image features in these images. In our application, automatic wafer probing, we have used the monocular method for obtaining the depth information. The description of this method and the experimental results are described in this chapter.

Guidance of the probe to its target pad is an important operation in the automation of wafer probing. In order to achieve closed-loop control of the probe, the position of the probe relative to its target must be established. This includes the proximity of the probe to the wafer which is determined from the depth of the object. Hence, the measurement of the depth of the probe is a vital task in this automation. As described in Chapter 2, in our experimental set-up, we only have a single view of the probe along the optical axis of the light source. Hence, we have to measure distance using only the available data, i.e. the pixel value of the image of the probe. We have used two modes of measurement. The first mode is applied when the probe is far from the wafer. In this case, the pixel values are corrupted by the bright light of the through-the-lens microscope (Figure 4.1). Hence, we cannot use the absolute values of the probe pixels for measurement. We have devised a method for measuring the distance using the degree of sharpness of the edges in the image. In our application, the scene comprises of the probe on a background of pads in the VLSI wafer. We have chosen the edges of the probe for the measurement of sharpness. A parameter related to sharpness is the blur around an edge. The amount of blur in a defocused object is inversely proportional to the sharpness of the object. We have developed a method of measuring the blur in the image and which, we then relate to the distance from the probe to the wafer.

The second mode of measurement is used when the probe is close, but not touching the wafer. In this case, we use the variance of intensity values to detect the touch of the probe to the wafer. This technique is described in Chapter 5.

As the probe approaches the wafer, images are captured at various instants and the two methods are applied for every probing operation. In this chapter, we discuss the first method for estimating the distance. In the following sections we describe the measurement of the degree of blur of the edges and relate this to the distance of the object. In Section 4.2, the previous work done in perception of depth using a single view is reviewed. A theoretical relation for estimation of blur near the edges is derived in Section 4.3. In Section 4.4, an algorithm for implementation of this relation is described. Finally, the algorithm is applied to a synthetic image and to experimental data, and the results are described in Sections 4.5 and 4.6 respectively.

4.2 Measurement of Depth using Blurring

Measurement of depth from focusing/defocusing has been studied by several researchers [39]—[46]. Initial work concentrated on automatic focusing algorithms.

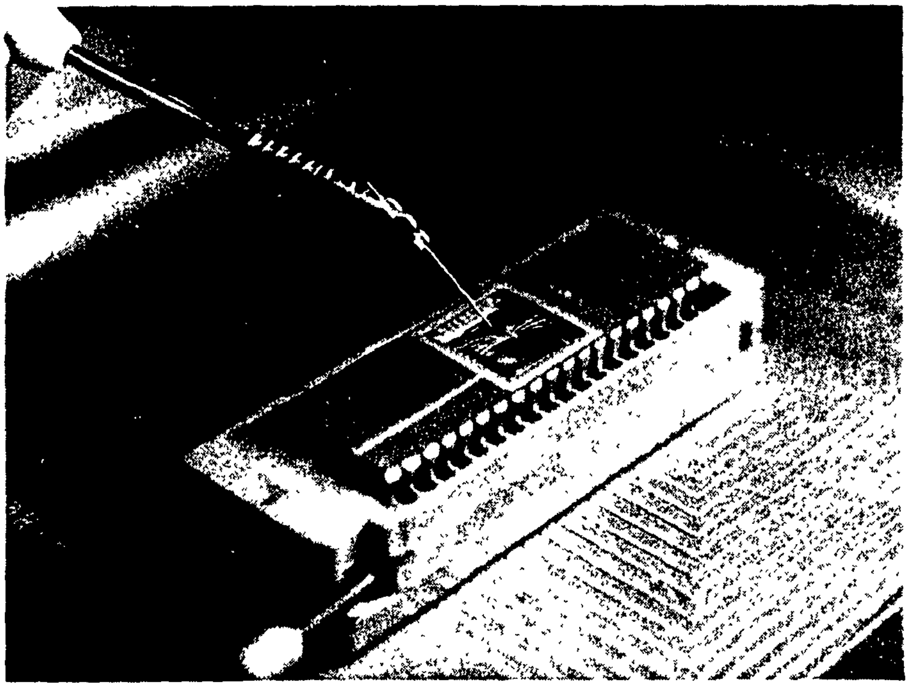


Figure 4.1 Figure showing the probe approaching the pad in the experimental set-up.

Hausler and Korner [46] used the following focus function for finding the extrema of intensity.

$$R(z) = \sum_{n=0}^{N} \left| \frac{\partial I_n(x,z)}{\partial z} \right| \tag{4.2.1}$$

where *n* is the number of pixels considered, *I(x,z)* is the intensity function with position *x* and depth *z*. They expected *R(z)* to have only one lowest extremum at the best focus. At this extremum the derivative of the image intensity is zero. This method may be useful for automatic focusing, but in our application since we analyze the grey values of the pixels on one side of the focus plane (i.e. when *z* is positive or negative), we cannot use this technique. Moreover it is also our intention to estimate the proximity of the probe to the wafer.

A recent approach to depth perception is to measure the degree of blur in the image. To our knowledge, three researchers, Grossmann [47], Pentland [40], and Subbarao [44] have used edge information in measurement of depth of objects. A method of obtaining depth profiles from sharpness of the edges was first used by Grossmann. The author described this method using experimental data. Initially, the first derivative of the intensity profile perpendicular to the edge direction is computed. The derivative function is bell shaped with a peak at the location of the edge. The width (*W*) of this distribution peak is evaluated. Grossmann also suggested a linear relationship between '*W*' and the distance '*d*' from the focused plane. The constant of proportionality was determined through calibration of experimental data. However, Grossmann concluded that this method is ill-defined and sensitive to noise. No further improvements on this method have been reported so far.

Pentland [40] used the second derivative of the intensity profile for calculating the blur in an image. The blurred image of a point object is considered as the point spread function of the optical system. This function is usually modeled by a two-dimensional Gaussian *G(r,σ)* with a spatial constant σ and radial distance r [40]. The value of σ in this model is the radius of the imaged point's "blur circle" or the blur parameter. A direct relation exists between the value of σ and the distance of the imaged point. This is given by the following expression [40].

$$D = \frac{Fv_0}{v_0 - F - \sigma f} \tag{4.2.2}$$

where v_0 is the distance between the lens and the image plane (i.e. the sensor location in the camera), f is the f-number of the lens system, F is the focal length of the lens system, and σ is the spatial constant of the point spread function. So, the spread parameter σ of the Gaussian distribution is inversely proportional to the depth D of the object in the scene. It is then possible to determine the distance of an object from the camera by estimating the value of σ.

Pentland [40] used the Laplacian operator of the intensity profiles near the edges for the calculation of σ. He derived the following expression for the calculation of the spread parameter σ of the Gaussian distribution as a linear regression of x^2.

$$ln\frac{\delta}{\sqrt{2\pi\sigma^3}} - \frac{x^2}{2\sigma^2} = ln\left|\frac{C(x,y)}{x}\right| \tag{4.2.3}$$

where x is the variable perpendicular to the edge and δ is the step height of the edge. By estimating the value of σ Pentland determined the distance of the image point. In this approach the Laplacian *C(x,y)*, around each edge is calculated. The value of δ is taken to be constant.

Subbarao [44] considered the first derivative of the intensity profiles perpendicular to each edge and calculated the distance of the object using the formula

$$\sigma = kDs\left[\frac{1}{f} - \frac{1}{u} - \frac{1}{s}\right] \tag{4.2.4}$$

where k is the camera constant, f is the focal length of the lens, D is the diameter of the lens, s is the distance from the lens to the image detector plane, u is the distance of the object from the lens and σ is the spread parameter (or blur parameter) of the line spread function. Subbarao computed the first derivative g_x along the intensity gradient by taking the difference of gray values of adjacent pixels (perpendicular to the edge direction). A line spread function $\theta(i)$ is computed using the following equation:

$$\theta(i) = \frac{g_x(i)}{\sum_{i=o}^{N} g_x(i)} \tag{4.2.5}$$

where g_x is the first derivate of the intensity function near the edge and N is the number of pixels considered. The spread σ of the line spread function is computed using the expression for the standard deviation of the line spread function.

$$\sigma = \pm\left[\sum_{i=0}^{N}(i - \bar{i})^2\,\theta(i)\right]^{1/2} \tag{4.2.6}$$

where $\bar{i}$ is the edge location. Although the height of the edge is considered in this approach, the first derivative in a noisy environment results in erroneous depth values.

The two methods described above attempt to estimate the spread σ of the point spread function which is modeled as a Gaussian distribution. They assume that the two surfaces separated by the edge are homogeneous and smooth. We found it difficult to use the above methods in our application as we have a mix of blurred and focussed regions. Also the object of interest (i.e. the probe) is at an angle to the optical axis, and therefore the focus varies along its edges. This can be observed in Figures 4.2 and 4.3. In addition, in our environment, we do not have smoothed regions or edges for estimation of proximity. The task is made even more difficult by the fact that we have to work with unconstrained illumination, background and hidden surfaces. We have derived a workable solution to this problem by dividing the region around an edge into three sections. The first and third sections belong to the background and the blurred object respectively while the second one belongs to the edge itself. We analyze these three parts of the edge separately and calculate the spread σ of the point spread function contributing to the blurring of the edge. By doing so, we arrived at a robust procedure for estimating σ and hence the distance from the tip of the probe to the focal plane i.e., the target pad. We have applied our algorithm to images of different backgrounds such as those shown in Figures 4.2 and 4.3. The following sections describe the theoretical background to our approach, the details of the algorithm, and the results obtained in our application.

4.3 Theoretical Development

We consider a step edge *f(x)* with magnitude δ, i.e.

$$f(x) = \begin{cases} k & \text{if } x<0 \\ k+\delta & \text{if } x \geq 0 \end{cases} \tag{4.3.1}$$

The line spread function of the camera with the lens system is modeled as a Gaussian distribution

$$g(x,\sigma) = \frac{1}{\sqrt{2\pi}\sigma} e^{\frac{-x^2}{2\sigma^2}} \tag{4.3.2}$$

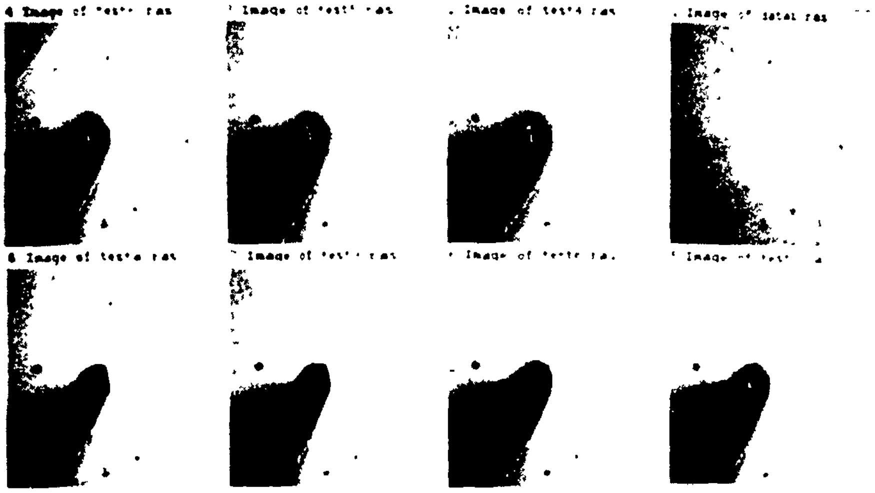


Figure 4.2 Images of the probe while approaching a uniform background.

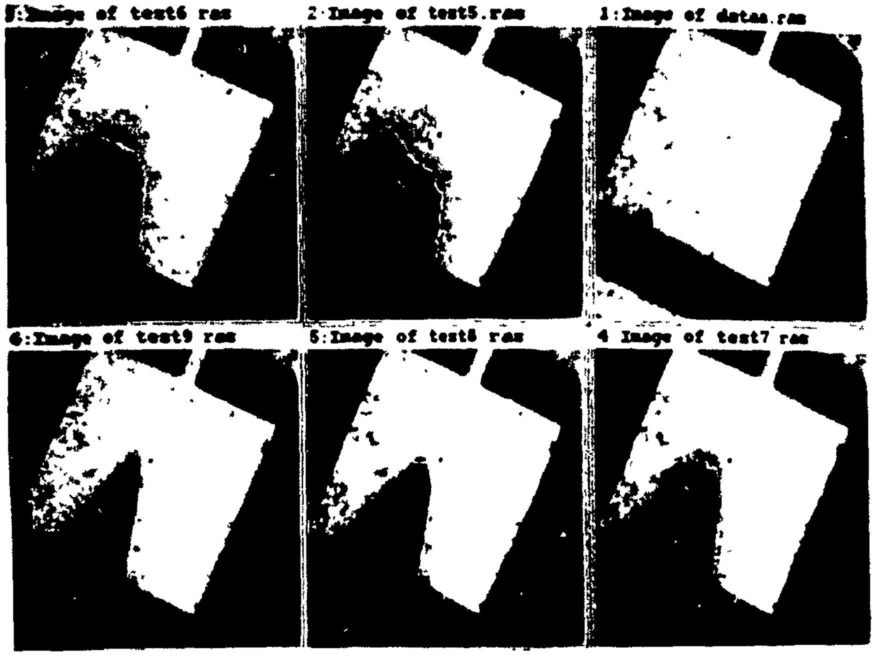


Figure 4.3 Images of the probe while approaching an Input/Output pad.

The step edge is convolved with the line spread function and yields a blurred edge. The blurred edge is represented by the following convolution function *h(x,σ)*.

$$h(x,\sigma) = \int_{-\infty}^{\infty} f(\xi)\, g(x-\xi,\sigma)\, d\xi \tag{4.3.3}$$

$$= k \int_{-\infty}^{0-} g(x-\xi,\sigma)\, d\xi + \int_{0}^{\infty} (k+\delta)\, g(x-\xi,\sigma)\, d\xi \tag{4.3.4}$$

$$= k \int_{-\infty}^{\infty} g(x-\xi,\sigma)\, d\xi + \delta \int_{0}^{\infty} g(x-\xi,\sigma)\, d\xi \tag{4.3.5}$$

Taking the first term on the right hand side

$$\begin{aligned} k \int_{-\infty}^{\infty} g(x-\xi,\sigma)\, d\xi &= -k \int_{-\infty}^{\infty} g(x-\xi,\sigma)\, d(x-\xi) \\ &= -k \int_{\infty}^{-\infty} g(\tau,\sigma)\, d\tau \end{aligned} \tag{4.3.6}$$

where we have changed the variable of integration to *τ=x-ξ*. Then (4.3.5) can be written as

$$h(x,\sigma) = k \int_{-\infty}^{\infty} g(\tau,\sigma)\, d\tau + \delta \int_{-\infty}^{x} g(\tau,\sigma)\, d\tau \tag{4.3.7}$$

Since

$$\int_{-\infty}^{\infty} g(\tau,\sigma)\, d\tau = 1 \tag{4.3.8}$$

and

$$\int_{-\infty}^{0} g(\tau,\sigma)\, d\tau = 1/2 \tag{4.3.9}$$

Equation (4.3.7) can be written as

$$h(x,\sigma) = k + \frac{\delta}{2} + \delta \int_{0}^{x} g(\tau,\sigma)\, d\tau \tag{4.3.10}$$

$$h(x,\sigma) = k + \frac{\delta}{2} + \delta \int_{0}^{x} \frac{1}{\sigma\sqrt{2\pi}} e^{\frac{-x^2}{2\sigma^2}} \tag{4.3.11}$$

The integral on the right hand side can also be expressed as an error function*. However, by differentiating the above equation with respect to *x*, we get

$$\frac{\partial h(x,\sigma)}{\partial x}\Big|_{x=0} = \frac{\delta}{\sigma\sqrt{2\pi}} \tag{4.3.12}$$

The function *h(x,σ)* can be represented using piece-wise linear segments.

$$h(x,\sigma) = \begin{cases} k & for\ \ x \in (-\infty .. p_1) \\ J_1(x,\sigma) & for\ \ x \in [p_1 .. p_2) \\ J_2(x,\sigma) & for\ \ x \in [p_2 \quad ..) \\ .... & \qquad .... \\ a_0 + a_1 x & for\ \ x \in [p_{z-1} .. p_{z+1}) \\ J_{z+1}(x,\sigma) & for\ \ x \in [p_{z+1} .. p_{z+2}\ ) \\ .... & \qquad .... \\ k + \delta & for\ \ x \in [p_3 .. p_\infty) \end{cases} \tag{4.3.13}$$

* An error function is defined as $\frac{1}{\sqrt{2\pi}} \int_{0}^{x} e^{\frac{-t^2}{2}} dt$

where $J_1(x,\sigma)$, $J_2(x,\sigma)$ and $J_{z+1}(x,\sigma)$ are piece-wise linear functions. Here p_z is the point of zero crossing. In order to calculate σ (using (4.3.12)), we must calculate the derivative of the blurred edge at its zero crossing and also the height of the step edge δ. This is shown in Figures 4.4 and 4.5. In our case, we approximate the blurred edge by a least-squares fit of the piece-wise linear segments. Then, we differentiate this function with respect to x and evaluate the result at $x=0$ i.e., at the point p_z in Figure 4.4.

$$\frac{\partial h(x,\sigma)}{\partial x}|_{x=0} = a_1 \tag{4.3.14}$$

From equation (4.3.12), we obtain an expression for σ as

$$\sigma = \frac{\delta}{a_1\sqrt{2\pi}} \tag{4.3.15}$$

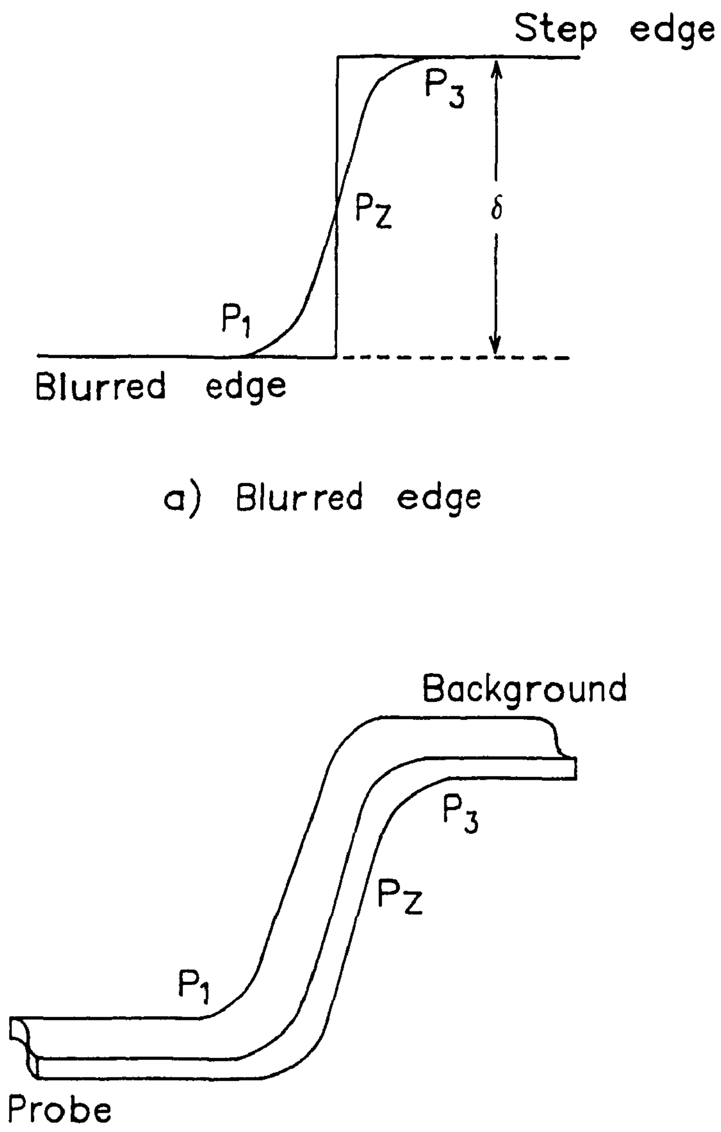
We can also estimate δ from (4.3.13) as

$$\delta = h(+\infty) - h(-\infty) \tag{4.3.16}$$

where

$$h(\pm\infty) = \lim_{x \longrightarrow \pm\infty} h(x,\sigma) \tag{4.3.17}$$

The standard deviation σ (which from now onwards we shall call "the blur parameter") of the Gaussian distribution allows us to determine the distance of the object using equation (4.2.2) or (4.2.4). We have used the following algorithm to obtain different parameters in (4.3.15).



a) Blurred edge

b) Surface around an edge

Figure 4.4 Ideal edge profile.

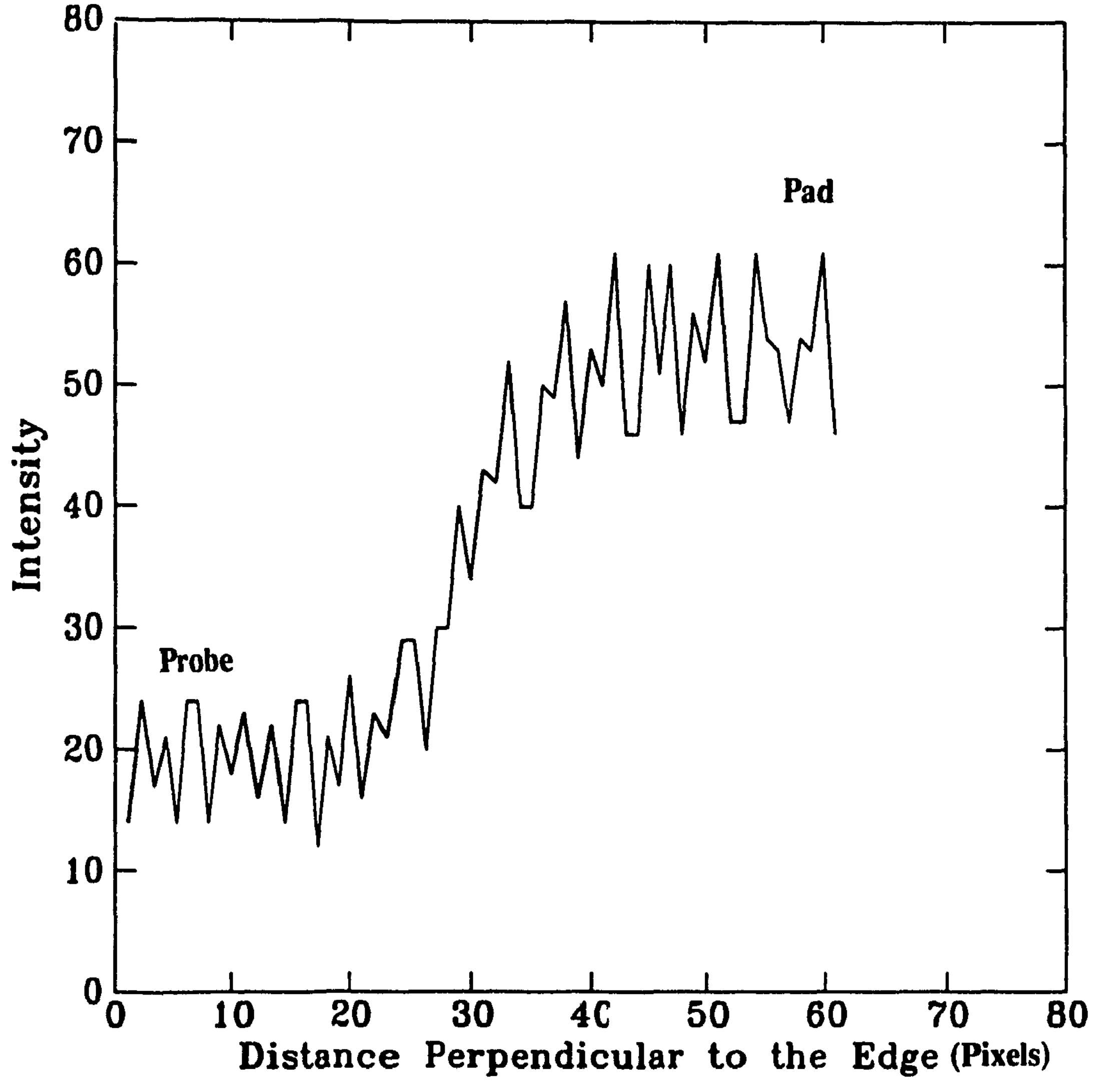


Figure 4.5 Edge profile for Figures 4.2 and 4.3.

4.4 Algorithm

In this section we describe the different steps in evaluating (4.3.15). These processing steps include the following: i) Obtaining an edge-map of the probe, ii) Using this estimate to find the actual zero-crossing and hence the "blur parameter" σ along each edge, and iii) Correlating σ with the measured distance of the probe from the surface of the pad. A detailed description of the algorithm is given in the following sections.

4.4.1 Generation of an Edge Map of the Probe

For analyzing the neighborhood of a blurred edge we need to determine the location of the edge. The procedure for obtaining the edge-map of the probe is described here. The image of the VLSI wafer consists of pads, probes and layout patterns. Due to bright through-the-lens illumination and different reflection characteristics of the materials of the wafer (e.g. substrate, metal and poly-silicon) the image is corrupted by a considerable amount of noise. Hence, for the extraction of edges, we smooth the image using a neighborhood of 3x3 pixels (the procedure for smoothing is described in Section 3.2.1). The probe, pad and background have distinct grey levels; we use these levels as threshold values for creating their images (the procedure for thresholding is described in Section 3.2.2). The edge-map of the probe is generated from the zero-crossings of the Laplacian of the probe-image (the procedure for edge detection is explained in Section 3.2.3). Such an edge-map is shown in Figures 4.2 and 4.3.

4.4.2 Interpolation of the Edge Map

[illegible] the presence of noise, the edge-map obtained in the previous section is not continuous, but contains gaps within the neighboring edges. These gaps are "filled in" with linear segments as follows: If (x_1, y_1), and (x_2, y_2) are the end points of two adjacent regions, then the new point (x_{new}, y_{new}) at a distance d from (x_1, y_1) is calculated using the following equation.

$$x_{new} = Int\left[x_1 \pm \left[\frac{d_i^2}{1+m^2}\right]^{1/2}\right] \tag{4.4.1}$$

$$y_{new} = Int\left[y_1 \pm \left[\frac{d_i^2}{1+\frac{1}{m^2}}\right]^{1/2}\right] \tag{4.4.2}$$

where m is the slope of the line joining (x_1,y_1), (x_2,y_2). The operator '*int*' produces a rounded value of the right hand side of the above formula.

4.4.3 Chaining of Edges

The probe is a conical object and the projection of this on the image plane has a triangular shape. We can observe this in Figures 4.2 and 4.3. The edges in the image plane represent the boundaries of the conical surface of the probe. Also, the probe is at an angle to the optical axis of the microscope. The end point of the tip is considered as the first element of the edge-map. The next element of the edge-map is located by searching the neighborhood and then linking to the first one. This chaining is continued upto the mount of the probe, i.e. until the farthest end is reached. This procedure results in a linked list of the edges starting from the tip to the mount.

4.4.4 Determination of Pixels Perpendicular to Edges

The blurring around each edge in the edge-map is analyzed. More specifically, we have to analyze the pixels perpendicular to each edge. This section describes the procedure for determining these pixels.

Each probe pixel in the image plane is the vertical projection of the three dimensional surface of the probe. Also, the grey levels of the pixels depend on the blurring due to defocusing. So the grey level profile around all the edges is considered for the calculation of the blur parameter σ. As per our assumption in (4.3.1), a profile of the edge is perpendicular to the edge. This profile is a set of image pixels lying normal to the edge under consideration. For each edge at (*x*,*y*) with slope *m*, we calculate a point (x_i,y_i) in the direction perpendicular to the edge and at a distance d_i from (*x*,*y*) using the following formula:

$$\begin{aligned} y_i &= Int\left[c \pm \left[\frac{d_i^2 + x^2 + (y-c)^2}{\left[1+\frac{1}{m}\right]^2}\right]^{1/2}\right] \\ x_i &= Int\left[\frac{y_p - c}{m}\right] \end{aligned} \tag{4.4.3}$$

By varying the distance d_i (*i*=1,2,3....) we find all the points (x_i,y_i) in the perpendicular direction. The operator '*Int*' in the above mentioned formulas produces the rounded value of the right hand side. Sixty pixels (*i*=1,2,3...60) in the perpendicular direction are considered for each edge. This number is determined experimentally to give a sufficiently detailed profile for most of the edges in our images. The edges and the region of pixels in the perpendicular direction are highlighted in Figures 4.2 and 4.3.

4.4.5 Extraction of the Three Regions in an Edge Profile

In this section we describe the procedure for determining the three regions in the edge profile. The vertical cross-section (profile of the edges) obtained with the above procedure (previous section) constitutes the blurred edge. The degree of the blur σ at each edge point can be calculated from the slope of the edge at its zero-crossing, and the height of the edge (see (4.3.15)). In our application, the height of the edge is the grey level difference between the two regions, i.e., the body of the probe and the background. The edge location computed using a Laplacian (described in Section 4.4.1) may not be the true zero-crossing of the edge profile. Hence we re-calculate the zero-crossing of the Laplacian using the following procedure. All the grey levels along the perpendicular direction are fitted with a polynomial. A zero-crossing of the second order derivative of this polynomial is taken as the true edge. The shape of the edge profile is shown in Figure 4.4. The point p_z denotes the zero-crossing and points p_1 and p_3 denote the two extrema of the edge. The points p_1 and p_3 for every edge profile are obtained by detecting the zero-crossings of the third order derivative of the edge profile. All the grey values of the points with coordinates in the range from the first point on the perpendicular direction to p_1 are considered to belong to the body of the probe. The grey levels of points with coordinates in the range (p_1, p_3) are considered to belong to the edge. Finally, the grey levels of points with coordinates from p_3 to the last point on the perpendicular direction point (60 in our case) are considered to belong to the background. These three sets of grey levels are used in the next step to calculate the blur parameter σ .

4.4.6 Piecewise Plane Fitting Around the Surface of the Edge

To determine the accurate slope of the edge profile (even for noisy profiles) we have used the neighboring edge profiles. This operation is valid since all the edge profiles are formed by a single object (probe) and the background. Thus, in order to evaluate the slope of the zero-crossing for an edge, we obtain the best fit plane for the set of points lying near the zero-crossing. This plane is represented as a_0+a_1x in (4.3.13). This is additionally parallel to the edge (refer to Figure 4.4). Similarly, the sets of points in the range $(-\infty, p_2)$ and $(p_{3,} + \infty)$ are also fitted with two planes and represent the probe and the background. Their difference yields an estimate of the height of the edge. In practice, the grey level value of the probe and the background cannot be accurately evaluated using the above mentioned fitting of two planes. This is because the grey levels of the probe in the image are corrupted by the high intensity reflection of the background. Moreover, the true height of the step edge is the grey level difference between the two ends of the edge profile (4.3.16). Since we have a single object (probe) and a unique background (metal pad), the grey level difference between these two ends is a constant for a selected lamp setting. This value was determined experimentally.

4.5 Application of the Algorithm to a Synthetic Image

The theoretical development of a new technique for depth perception was described in Section 4.3. Also, an algorithm was suggested in Section 4.4 for implementation of the formula derived in Section 4.3. Before applying this new technique to real experimental data, we have validated and evaluated our approach using a synthetic image. A step edge (shown in Figure 4.4) is generated and convolved with a Gaussian function of known σ (as given in Section 4.3). The blurred edge (for various values of σ) is shown in Figure 4.6. The blur parameter σ for these images is calculated using the algorithms described in Section 4.4. The results are given in Table 4.1. The two columns in this table represent the theoretical and experimentally determined values of σ.

We have used six bits per pixel in generating the synthetic edge. This resolution is chosen since our camera has a resolution of only six bits per pixel. Nevertheless, we have achieved close agreement between the theoretical and estimated values of σ. The estimated values at lower ranges are affected by quantization. In this method, we are more interested in estimating the proximity between the probe and the wafer when the probe is far from the wafer, i.e., at higher ranges of σ. Hence, this method can be used with confidence for this application (see Table 5.1).

4.6 Application of the Algorithm to Images of VLSI Wafers

The real challenge for any new technique is the ability to solve problems in the real world. In fact, all our efforts are directed towards the successful guidance of the probe to the target surface on the wafer. In this section we describe the results obtained by applying of the formula (4.3.15) to experimentally obtained images of the probe. The algorithm described in Section 4.3 is used to estimate the blur σ in the image of the probe. The algorithm is applied to images containing different backgrounds in VLSI wafers. In particular two kinds of image are considered; i) Diffusion layer as the background

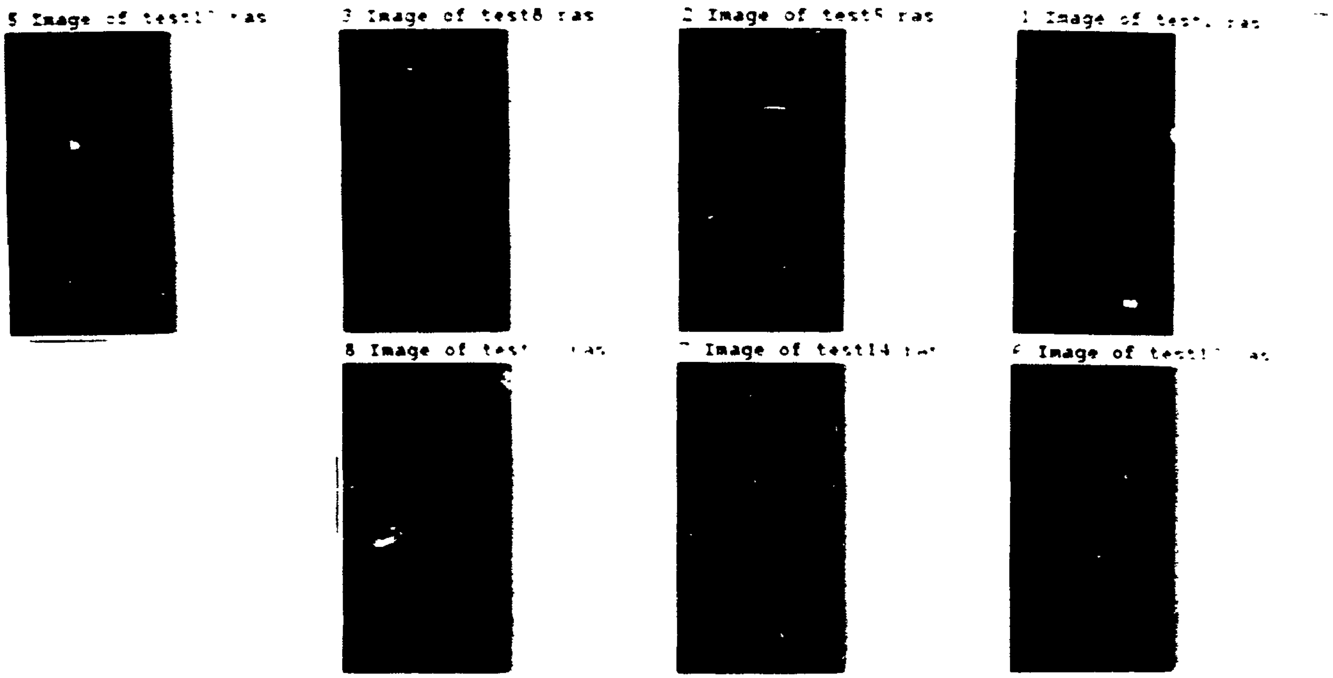


Figure 4.6 Synthetic image of an edge blurred with different values of σ.

Theoretical	Estimated
3	4.27
4	4.98
5	5.09
6	6.15
7	7.2
8	7.98
9	9.034
10	10.09
11	11.05
12	12.074
13	12.94
14	13.66
15	14.67

Table 4.1 Theoretical and estimated values of σ for the synthetic image shown in Figure 4.6.

(described in Section 4.6.1) and ii) I/O metal pad as the background (described in Section 4.6.2). However, in practice, only metallic pads are used for probing the wafer. But, for validating our technique of calculating σ, we have considered various pads in the integrated circuits available in our laboratory. The deviation in the calculated value of σ for ten sets of experimental data is discussed in Section 4.6.2. Finally, with this value of σ, the actual distance can also be evaluated using (4.2.2) or (4.2.4). For this, we require the optical constants obtained from the specifications of the microscope and the camera.

4.6.1 Diffusion Layer as Background

Images of uniform background, i.e. images of non-inscribed wafers are considered in this section and the value of σ is calculated using (4.3.15). A sequence of frames (images of the probe and background) with known distance is captured. These are shown in Figure 4.2. We can observe in the figure that the image of the probe is blurred when the probe is "far" from the wafer. The last frame in Figure 4.2 represents the probe when it is touching the wafer. In our experimental set-up the probe is at an angle to the optical axis. Since the microscope is focused on the wafer, the mount of the probe is more blurred compared to the tip. Although the wafer surface in Figure 4.2 appears smooth, the grey level surface is rough due to the specular and diffused noise (Figure 4.5). The situation is further aggravated by the blurred image of the probe. Even then, the results are satisfactory.

The edge profile of the probe is obtained as described in Section 4.4. Consecutive images are analyzed while the probe is approaching the wafer. For each image, the value of σ at every edge of the probe is calculated. The variation of σ along the surface of the probe (while it is touching the surface) is shown in Figure 4.7. Each point in this figure represents the value of σ for the corresponding edge in the image of the probe. Least-squares fitting of a straight line through all these points is also shown in Figure 4.7. We can observe from this figure that the probe orientation with respect to the optical axis can be estimated from the measurement of σ. This is a useful feature in understanding orientation using blurring.

The procedure described above, i.e., fitting a straight line along the probe, is repeated for all the images while the probe is approaching the wafer. The variation of σ for each image (represented by a straight line) of the probe is shown in Figure 4.8. In this figure, all the lines are staggered and almost parallel to each other. The bottom line represents the image of the probe while touching the wafer. The top line represents the image of the probe farthest away from the wafer surface. This confirms our assumption that the probe is approaching the wafer through movement along the z-axis.

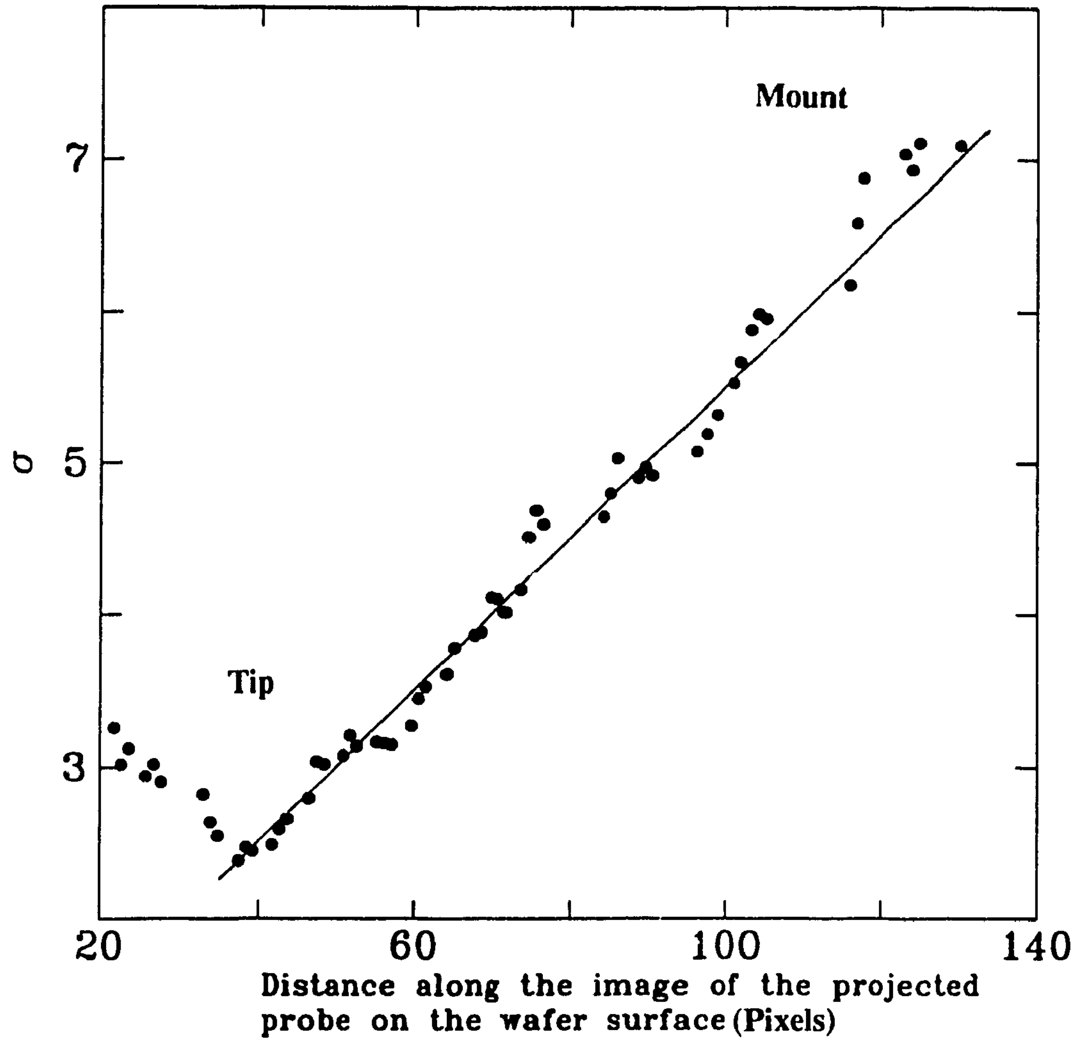


Figure 4.7 Fitting a straight line for σ along the probe.

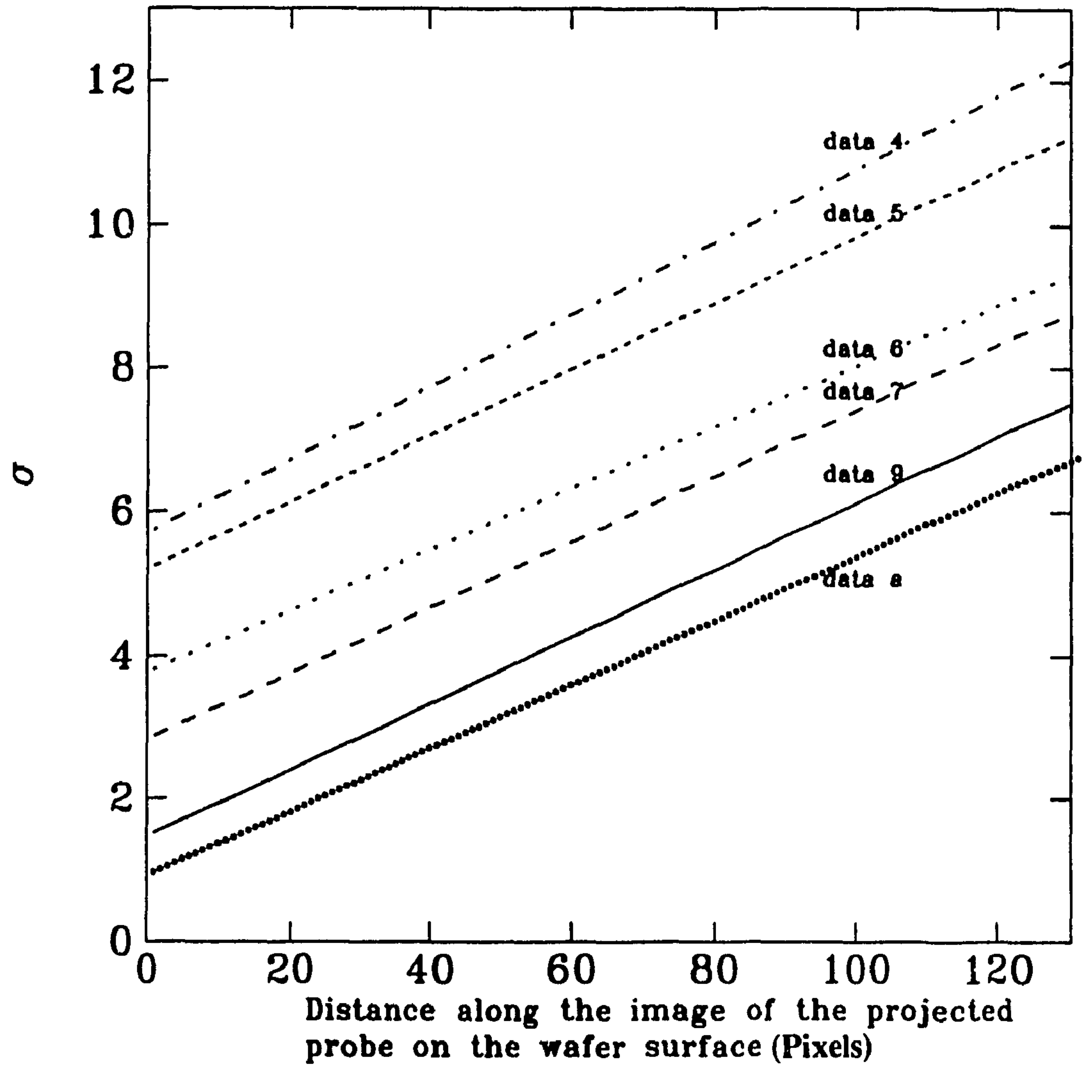


Figure 4.8 Variation of σ along the probe for Figure 4.2.

4.6.2 Metal Pad as Background

The second set of images considered are the I/O pads. A sequence of images was analyzed as the probe approaches the wafer. These are shown in Figure 4.3. There are two visible layers in the image of the wafer; metal and substrate. The metal region is bright and consequently has very high contrast with the probe. The tip is positioned at the center of the I/O pad. As the tip of the probe is very sharp, the loci of the perpendiculars to the edges consist of different regions in the image. Also, the edge profiles at the point of contact are corrupted by the high intensity reflections from the metal surface. Moreover, the surface of the probe near the boundary of the metal pads does not have a homogeneous background (Figure 4.3). This is not desirable because irregular edge profiles result in inaccurate σ values. Because of the above mentioned reasons, we have considered the edge profiles over a small region (30 edge profiles) near the tip and calculated the average value of σ for this region. The values of σ for various frames while the probe is approaching the wafer are shown in Figure 4.9.

The final objective for the measurement of σ is the controlled guidance of the probe to the target pad. The degree of confidence in the measurement of σ and subsequently the estimation of distance is an important consideration in any new technique, and ours is no exception. We describe below some experimental results to meet this consideration. One useful observation in our application is that only metal pads are used as contact points in any VLSI design. Therefore, we have considered 10 sets of images, where each set contains images of the probe while approaching different metal pads. For each image of the probe, the value of σ is calculated as described in the previous paragraph. The mean values of all 10 sets is plotted against the distance and are shown in Figure 4.9. The deviation of σ for each distance is also shown. By observing the figure, we can conclude that for any calculated value of σ it is possible to estimate the distance of the probe and hence the required Z-drive to the probe.

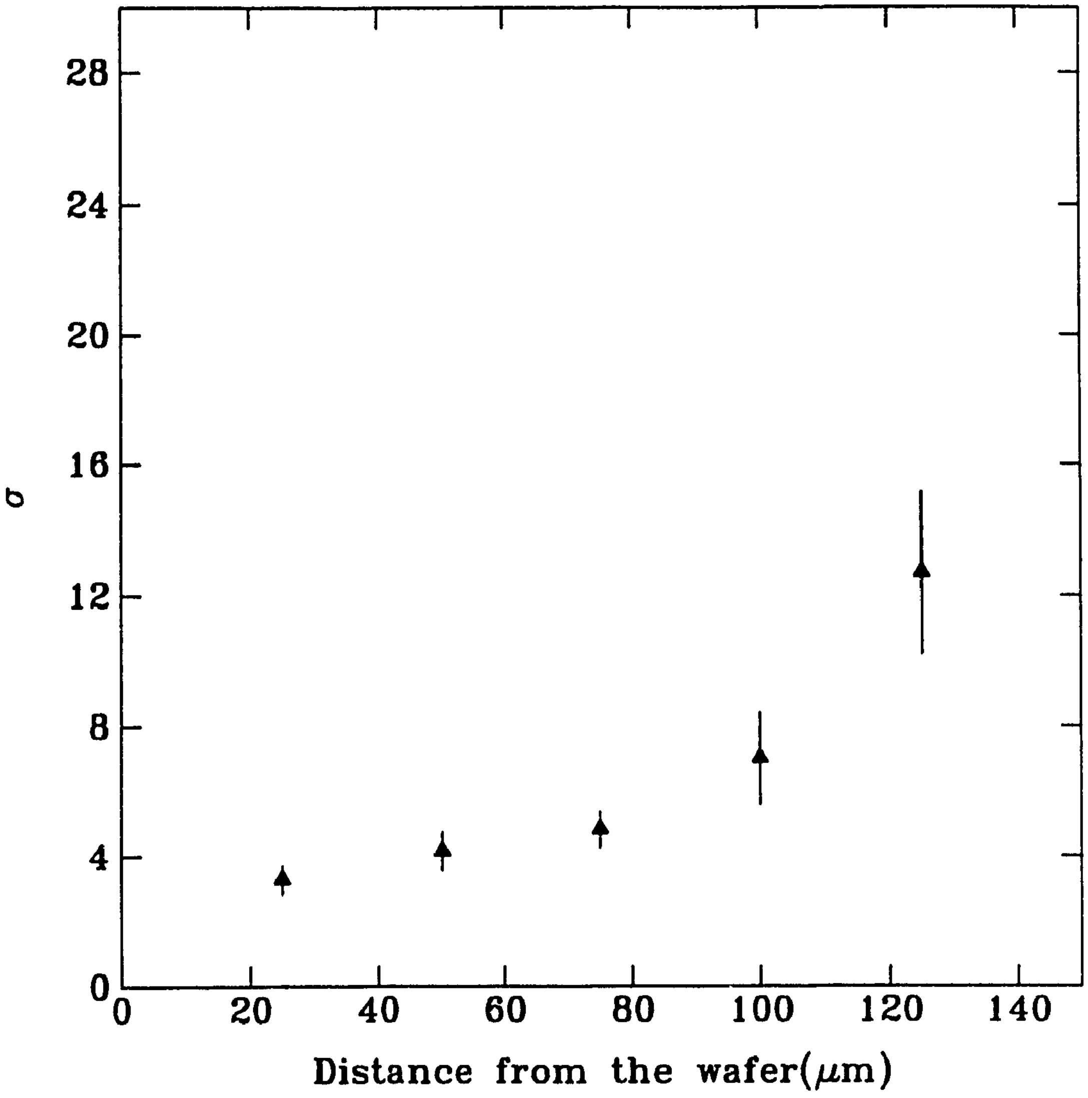


Figure 4.9 Deviation from mean value of σ.

4.7 Conclusions

We have developed a method to estimate the depth of an object using information concerning blurring around the edges. The methods suggested by Pentland [40], [41] and Subbarao [44] use second and first derivatives of the intensity profiles perpendicular to the edge. Because of the derivative calculations along the entire cross-section, these methods are sensitive to noise and are not suitable for some uncontrolled environments, like the application considered in this thesis, where the image is corrupted by noise due to incorrect illumination, motion etc. In our method, we need to find the slope of the edge-profile at one point, i.e. at zero-crossing. In order to estimate the slope of an edge, even in the presence of noise, we fit a plane surface in the neighborhood of the zero-crossing. The slope of this plane is used in calculating σ. In contrast to other methods, the measured slope in our procedure is less susceptible to noise.

In order to confirm our theory, as well as to test it in a real environment, we have applied it to determine the distance of a probe from the surface of a VLSI wafer. We have considered 10 sets of images of different VLSI wafers belonging to different batches in the fabrication process. The parameter which is uniformly varied in all these images is the blur parameter σ . It can be observed from Figures 4.7 and 4.8 that similar values of σ were obtained for different backgrounds. This confirms our theory and shows that σ can be used as a measure in guiding the probe to its destination on the wafer. The deviation in the measurement of σ can be reduced by considering a higher number of grey levels (e.g. 256) than in the present set-up (64 levels). Another concern is in the measurement of δ (the height of the step edge in (4.3.15)). The values of δ need to be calibrated for different settings of the lamp illumination. This will further improve the measurement. Due to the above mentioned limitations, we use this method for coarse measurement of distance of the probe from the wafer. Hence, this method is not suitable for the detection of touch (when σ =0). We shall use another method for this purpose [48]. This is described in the next chapter

Another interesting observation is that a smoother variation of σ is observed for a uniform background, e.g. a single layer of the background material (Figure 4.2), than for non-uniform backgrounds, e.g. one having several layers (Figure 4.3). Hence a blurred object on a uniform background gives a more accurate measure of the distance from the object to the background. Also, we observe that the slope of the plane near zero-crossings, for different backgrounds, varies very significantly and agrees with common experience that a blurred object looks sharper against a high contrast background than against a low contrast one. This type of information may also be used for identifying boundaries between different regions. Our human visual system can neither estimate nor identify the vertical orientation of the probe by viewing along the optical axis (see Figures 4.2 and 4.3). But it is possible through machine vision to obtain this information by measuring the blur parameter of the blurred object (Figure 4.8). This interesting result may be useful in other applications of machine perception.

Chapter 5
Detection of Touch

5.1 Introduction

One of the problems in the automation of assembling operations using industrial robots is the detection of satisfactory contact between the object and the end effector. This contact can be detected using a visual or nonvisual sensor. However, the visual sensor is ideal since it is a non-contact sensor. Other types of sensors are: touch, tactile and slip. All nonvisual sensors use force measurement and hence require physical contact with the object. This mode of sensing is not always possible and sometimes not desirable. One such application is wafer probing, where the detection of contact between the probe tip and the wafer surface is a very sensitive operation due to the thickness of the probe tip (1 micron). Moreover, the thickness of the metalization layer of the wafer is around 0.5 micron. So, implanting any of the above mentioned nonvisual sensors is not possible.

For detecting contact in wafer probing, another solution is the measurement of leakage current between the probe tip and the die (grounded). But, the leakage current will be in the order of pico amperes and its measurement is subject to errors due to infringing capacitances. Alternatively, we can measure the closed circuit resistance between the probe and the contact area of the die (also grounded). For these measurements, we have to ground either the die or the contact area of the wafer surface. Also, grounding the contact surface (in an intermediate die in the wafer) is a difficult and time consuming task. Hence, measurement of resistance for detecting the contact is not a practical solution and we have not pursued it any further. Thus, visual information for guiding the probe was selected as a viable alternative.

In order for the probe to make good contact and not destroy the metalization layer, the probe's impact velocity and force must be accurately controlled. Also, the probe has to make good electrical contact for reliable testing of the wafer [9]. Deformation of the distance of a probe from the wafer and the detection of touch are important issues in achieving the above objectives. We have evolved a two stage control strategy for the guidance of the probe to its target pad. The first stage is a coarse control (by using distance information) for the movement of the probe which was described in Chapter 4. In this chapter we describe the second stage which is to be used for fine control of the movement of the probe and detection of touch.

The approach used for coarse control is as follows: when the probe is near the wafer, the image consists of the background and the blurred probe. Using information on blurring around the edges, the algorithm described in Chapter 4 [48] estimates the parameter σ (standard deviation) in the line spread function*. As the probe comes close to the wafer, the value of the parameter σ decreases [48]. The measurement of blur using this procedure is very suitable for coarse control of the probe but is not suitable for small values of σ (e.g. less than 3 pixels, when the probe is about to touch the wafer).

Since we require a robust but very accurate sense of touch, we have devised another procedure (for fine control of the movement) using the pixel values of the blurred probe. This is made possible by the fact that when the probe is very close to the wafer, it is easy to separate the pixels of the probe from the background. We considered two sets of pixel values, the first containing the probe with the background and the second containing only the probe. The second set is obtained from the first by subtracting the background. We study the relation of the variance of the pixel values in each set with the standard deviation σ of the line spread function.

* The line spread function of the optical system is modeled as a Gaussian function, which is $\frac{1}{\sqrt{2\pi}\sigma}e^{\frac{-x^2}{2\sigma^2}}$

We have also studied the relation of the above mentioned variances (corresponding to the two sets of data) and the depth of the object in a given scene. We consider the general scenario where an object is approaching a uniform background. We obtain analytical expressions for the mean and variance of the object and study the variation of these values with respect to the distance of the object from the surface. Initially, we derive analytical expressions for variance in different sets of data in the image and relate them to the value of σ (Sections 5.2 and 5.3). Next, we prove that the variances corresponding to the two sets of data in the image are enough for the calculation of σ (Section 5.4).

In addition to the variance of pixel values of the probe, we used an experimental phenomenon for detecting touch. This is explained below. We observed that the probe "slides" as soon as it touches the surface. The permitted "sliding" has been found experimentally to be of the order of a few microns, which corresponds to a difference in the position of the tip of the probe by about 32 pixels (using a lens 3 magnification 500x). Therefore, the slide of the probe introduces considerable change in the mean and variance of the pixel values of the probe in the image. This change corresponds to the moment of contact. The experimental results are described in Section 5.5.

5.2 Variance of Intensity Values of a Blurred Object with a Focused Background

In a vision system which comprises an optical element with a small numerical aperture focused on a fixed background, the movement of an object towards or away from the background gives rise to a change in intensity values of the image. By measuring this change, it is possible to determine the distance from the object to the background. A simple method is to calculate the mean and the variance of the intensity values.

In this section, we derive a relationship between the variance of the pixel values of an object ($\sigma_I{}^2$) and σ, the "blur parameter" in the line spread function. To simplify the mathematical analysis, a cross-section of an object of a certain width is considered in Figure 5.1. Also shown in this figure are the background (*K*), the blur function $h_l(x)$, and the spread of the Gaussian (*t*) function*. (The area under the Gaussian function ($-\infty$ to $+\infty$) is equal to 1. Since we do not have ∞ points, we choose the limits as large as possible so tha: the value of the function is close to 1. For our purpose we have selected the value of the function as 0.97) The variance of the pixel values of the function $h_l(x)$ is calculated and used in Section 5.4 to derive a formula for the blur parameter σ. The theorem below explains the relationship between the variance of $h_1(x)$, the spread of the Gaussian (*t*), the width of the object (*T*) and the pixel value of the background (*K*).

Theorem 2.1 Let *f(x)* be the image (defined in the interval *–N/2* to *N/2*) consisting of an object with width *T* and a background of grey value *K*. Let $h_1(x)$ be the image of the background and the object away from the background. If the line spread function of the blurred object is modeled as a Gaussian function $g(x)$ with standard deviation σ, then the variance σ_1^2 of the pixel values of such a blurred object with a focused background is given as $K^2 \left[\frac{2T}{N} - \frac{4T^2}{N^2} - \frac{4q\sigma}{N}\right]$, where q is a constant.

Proof. As shown in Figure 5.1, the function *f(x)* is defined in the interval from *–N/2* to *N/2* as follows:

$$f(x) = \begin{cases} 0 & -T \leq x \leq T \\ K & otherwise \end{cases} \tag{5.2.1}$$

The convolution of *f(x)* with *g(x)* results in a function $h_1(x)$ which is expressed as:

$$h_1(x) = f(x) * g(x) \tag{5.2.2}$$

[*] The value of *t* is selected such that $\int_{-t}^{t} g(x)\,dx \approx 1$ where *g(x)* is a Gaussian distribution with standard deviation σ.

The function $h_1(x)$ in the interval $-(t+T) \le x \le -(T-t)$ is expressed by the following equation.

$$\int_{-t-T}^{-T+t} g(x-\tau)\, d\tau = \frac{1}{\sqrt{2\pi}\ \ \sigma} \int_{x+T}^{\infty} e^{\frac{-v^2}{2\sigma^2}} dv \\ = Q\left(\frac{x+T}{\sigma}\right) \tag{5.2.3}$$

where $Q(x) = \frac{1}{\sqrt{2\pi}} \int_{x}^{\infty} e^{\frac{t^2}{2}} dt$

Similarly we write the blur function $h_1(x)$ in the interval $(T-t) \le x \le (T+t)$ as

$$\int_{T-t}^{T+t} g(x-\tau)\, d\tau = \frac{1}{\sqrt{2\pi}\,\sigma} \int_{x-T}^{\infty} e^{\frac{-v^2}{2\sigma^2}} dv \\ = Q\left(\frac{x-T}{\sigma}\right) \tag{5.2.4}$$

Hence, the blur function $h_1(x)$ in the range *–N/2* to *N/2* can be written as follows:

$$h_1(x) = \begin{cases} K & -N/2 \le x < (-T-t) \\ KQ\left(\frac{x+T}{\sigma}\right) & (-T-t) \le x < (t-T) \\ 0 & (t-T) \le x < (T-t) \\ K - KQ\left(\frac{x-T}{\sigma}\right) & (T-t) \le x < (T+t) \\ K & (T+t) \le x \le N/2 \end{cases} \tag{5.2.5}$$

In our application, the background is uniform near the probe. Hence, the value of K can be considered to be constant. The mean or expected value of the function $h_1(x)$ can be written as

$$E\,[h_1(x)] = \frac{1}{N} \int_{-N/2}^{N/2} h_1(x)\, dx \tag{5.2.6}$$

We now evaluate the above integral for the intervals defined in (5.2.5).

$$\frac{1}{N}\int_{-N/2}^{N/2} h_1(x)\; dx = \frac{K}{N}\Bigg[\int_{-N/2}^{-T-t} dx + \int_{-T-t}^{t-T} Q\left(\frac{x+T}{\sigma}\right) dx + \int_{T-t}^{T+t} d\,x$$

$$- \int_{T-t}^{T+t} Q\left(\frac{x-T}{\sigma}\right) dx \;+\; \int_{T+t}^{N/2} dx\Bigg] \qquad (5.2.7)$$

By using the results in Appendix F to evaluate the above integrals, we get the following expression:

$$E\left[h_1(x)\right] = \frac{K}{N}\left[N - 2T\right] \qquad (5.2.8)$$

and the square of the expected value is:

$$\left[E\;\left[h_1(x)\right]\right]^2 = \frac{K^2}{N^2}\left[N^2 - 4NT + 4\,T^2\right] \qquad (5.2.9)$$

The expected value of the square of the function $h_1(x)$ can be written as follows:

$$E\left[h_1^2\;(x)\right] = \frac{1}{N}\int_{-N/2}^{N/2} h_1^2(x)\,dx \qquad (5.2.10)$$

Again, we evaluate the above integral for all intervals in (5.2.5)

$$\frac{1}{N} \int_{-N/2}^{-N/2} h_1^2(x)\, dx = \frac{K'^2}{N} \Big[\int_{-N/2}^{-T-t} dx + \int_{-T-t}^{t-T} Q^2\left(\frac{x+T}{\sigma}\right) dx + \int_{T-t}^{T+t} dx \tag{5.2.11}$$

$$-2 \int_{T-t}^{T+t} Q\left(\frac{x-T}{\sigma}\right) dx + \int_{T-t}^{T+t} Q^2\left(\frac{x-T}{\sigma}\right) dx + \int_{T+t}^{N/2} dx \ \Big] \tag{5.2.12}$$

Evaluating the above integrals (see Appendix F), we get

$$E\left[h_1^2\,(x)\right] = \frac{K'^2}{N}\ \left[N - 2T - 4q\,\sigma\right] \tag{5.2.13}$$

where q is a real value defined in Appendix F.

The variance of $h_1(x)$ can be evaluated by using the following expression:

$$\sigma_1^2 = E\left[\,h_1^2(x)\right] - \left[E\left[h_1(x)\right]\right]^2 \tag{5.2.14}$$

Using (5.2.9) and (5.2.13) we get the following expression for the variance σ_1^2.

$$\sigma_1^2 = K'^2\left[\frac{2T}{N} - \frac{4T^2}{N^2} - \frac{4q\sigma}{N}\right] \tag{5.2.15}$$

Completion of the proof.

From the above equation, we note that the variance σ_1^2 of the pixel values of the blurred probe with a background increases with decreasing values of σ, the blur parameter in the line spread function, provided that σ satisfies the following inequality.

$$0 \leq \sigma \leq \frac{2T - \frac{4T^2}{N}}{4q} \tag{5.2.16}$$

Since σ is non-negative and q is positive definite and real, the above condition is valid when $T \leq N/2$. Therefore, the above condition is valid when the object width $2T$ is less than the image size, N. Since this is always true, we conclude that the variance σ_1^2 of the blurred object with a background increases with decreasing value of σ.

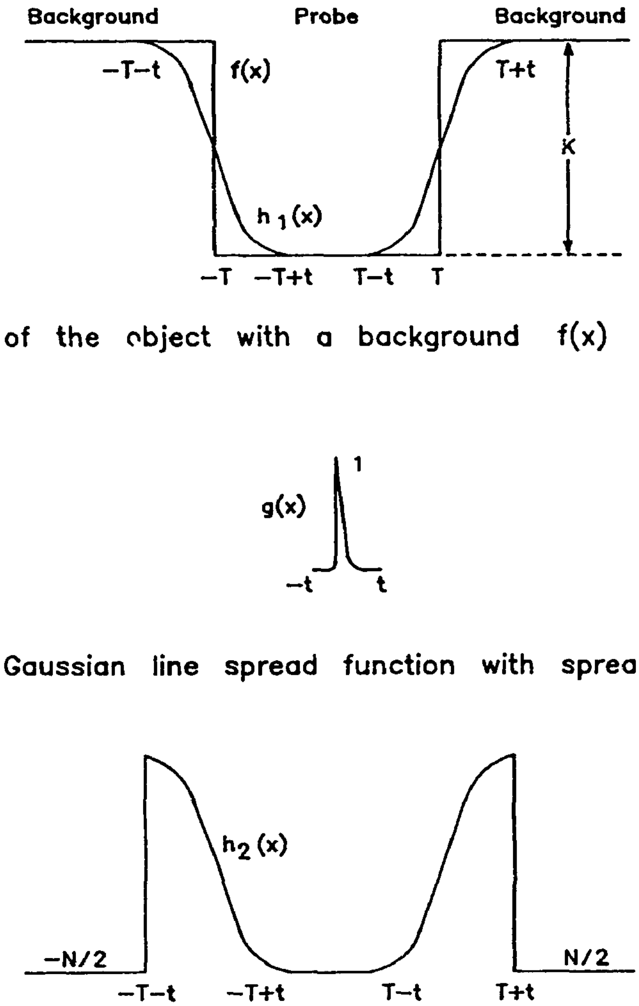
Equation (5.2.15) also contains the pixel value of the background. But, we need a relation independent of the background value. Hence, we need to derive the variance of the pixel values contributed by the blurred object by excluding the background. This procedure is explained next.

5.3 Variance of Intensity Values of the Blurred Object

In the real world, we often deal with objects (probes) having various types of uniform backgrounds (e.g. metal and diffusion layers). In this section, we formulate another variance σ_2^2 by taking into account the pixel values contributed by the probe. This is achieved by deleting the background from the blurred image of the probe with the background. Then the variance σ_2^2 is calculated for these pixel values. In this section, we derive a relationship between ${\sigma_2}^2$ and σ.

Theorem 3.2 Let a background (defined in the interval *–N/2* to *N/2*) with gray level K and size N be given and let an object of width T be close to the background. If the line spread function of the blurred image is modeled as a Gaussian function $g(x)$ with standard deviation σ, then the variance of the pixel values of the blurred object is $\sigma \left[\frac{K^2\, 2\,\alpha}{N} - \frac{K^2\, 4q}{N} - \frac{4K^2\alpha^2\sigma}{N^2}\right]$, where α and q are constants.

Proof. The blurred function $h_2(x)$ is shown in Figure 5.1. As in Section 5.2, the blurred function $h_2(x)$ can be represented in various intervals as:



a) One dimensional function of the object with a background f(x) and its blur function $h_1 (x)$.

b) A Gaussian line spread function with spread t.

c) One dimensional function of the object after subtracting the background.

Figure 5.1 One-dimensional functions of an object while in contact with a uniform background.

$$h_2(x) = \begin{cases} 0 & -N/2 \leq x < (-T-t) \\ KQ\left(\frac{x+T}{\sigma}\right) & (-T-t) \leq x < (t-T) \\ 0 & (t-T) \leq x < (T-t) \\ K - KQ\left(\frac{x-T}{\sigma}\right) & (T-t) \leq x < (T+t) \\ 0 & (T+t) \leq x \leq N/2 \end{cases} \tag{5.3.1}$$

The expected value of the above function can be obtained as follows.

$$\begin{aligned} E[h_2\ (x)] &= \frac{1}{N} \int_{-N/2}^{N/2} h_2(x)\ dx \\ &= \frac{K}{N}\left[\int_{-T-t}^{t-T} Q\left(\frac{x+T}{\sigma}\right) dx + \int_{T-t}^{T+t} dx - \int_{T-t}^{T+t} Q\left(\frac{x-T}{\sigma}\right) dx\right] \\ &= \frac{2tK}{N} \end{aligned} \tag{5.3.2}$$

Therefore, the square of the expected value is

$$[E[h_2\ (x)]\,]^2 = \frac{4t^2K^2}{N^2} \tag{5.3.3}$$

We also calculate the expected value of the square of the function as follows:

$$E\left[h_2^2\ (x)\right] = \frac{K^2}{N}\Bigg[\int_{-T-t}^{t-T} Q^2\left(\frac{x+T}{\sigma}\right) dx + \int_{T-t}^{T+t} dx - 2\int_{T-t}^{T+t} Q\left(\frac{x-T}{\sigma}\right)$$

$$+ \int_{T-t}^{T+t} Q^2\left(\frac{x-T}{\sigma}\right) dx\Bigg] \tag{5.3.4}$$

Using the results in Appendix F, we get

$$E\left[h_2^2(x)\right] = \frac{K^2}{N}\left[2t - 4q\ \sigma\right] \tag{5.3.5}$$

The variance of $h_2\,(x)$ is given by

$$\begin{aligned}\sigma_2^2 &= E\left[h_2^2\ (x)\right] - \left[E\left[h_2\ (x)\right]\right]^2 \\ &= \frac{K^2}{N}\left[2t - 4q\ \ \sigma\right] - \frac{4t^2\ K^2}{N^2} \\ &= \sigma\left[\frac{K^2\ 2\ \alpha}{N} - \frac{K^2\ 4q}{N} - \frac{4K^2\alpha^2\sigma}{N^2}\right]\end{aligned} \tag{5.3.6}$$

where we have substituted $t{=}\alpha\sigma$. This completes the proof of the theorem.

From equation (5.3.6), we see that the variance σ_2^2 of the blurred object is directly proportional to σ, the standard deviation of the Gaussian distribution for $\alpha > 2q$ and $\frac{N}{4\alpha^2} >> 1$. In fact, the value of α is set by the threshold value (described in Section 5.4) and always greater than $2q$. We have calculated σ_2^2 experimentally and we use this value for detecting the operation of touch. This method is described in Section 5.5.

Since we have two relations (5.2.15) and (5.3.6) involving σ_1^2, σ_2^2, σ, and background gray level K, we can eliminate the parameter K to obtain a solution for the value of σ in terms of σ_1^2 and σ_2^2. Such a closed-form solution is derived in the next section.

5.4 Calculation of the Blur Parameter σ

Recently, there has been considerable interest in measuring depth using blurring as described in [39]—[41] and [44]. All these researchers have formulated a relationship between the distance of an object and the amount of blur (σ in the line spread function) in the scene. In our application, we can control the guidance of the probe by measuring σ. In this section, we derive a closed form solution for the value of σ as a function of σ_1^2 and σ_2^2.

By using the expressions for the variances derived in Sections 5.2 and 5.3 we evaluate the value of σ, the blur parameter in the line spread function. From Section 5.2, the variance σ_1^2 is given by

$$\sigma_1^2 = \frac{K^2}{N}\left[2T - \frac{4T^2}{N} - 4q\,\sigma\right] \tag{5.4.1}$$

Also, from Section 5.3 the variance σ_2^2 is given by

$$\sigma_2^2 = \sigma\left[\frac{K^2\,2\,\alpha}{N} - \frac{K^2\,4q}{N} - \frac{4K^2\alpha^2\sigma}{N^2}\right] \tag{5.4.2}$$

By eliminating K from the above two equations, we get the following quadratic equation in σ.

$$\sigma^2\left[\frac{4\alpha^2}{N}\right] - \sigma\left[2\alpha + 4q\left[\frac{\sigma_2^2}{\sigma_1^2} - 1\right]\right] + \frac{\sigma_2^2}{\sigma_1^2}\left[2T - \frac{4T^2}{N}\right] = 0 \tag{5.4.3}$$

We obtain values of σ by solving this equation. We know N (the size of the image), T (the size of the object), q (given in Appendix F) and the experimentally determined variances σ_1^2 and σ_2^2. The value of α is chosen according to the value of the threshold used in the algorithm given in Appendix G. For example, for threshold value 0.05 (which is $Q(\alpha)$), the value of α is 2.5. This also implies that the area under the Gaussian function in Section 5.2 is 0.97. This value of threshold is selected experimentally and satisfies the following criteria. The smaller the value of the threshold, the larger is the number

Theoretical	Estimated
1.0	0.771421
1.5	1.6489
2.0	1.9935
2.5	2.346
3.0	2.69
3.5	3.083
4.0	3.87

Table 5.1 Theoretical and estimated values of σ calculated using variance of the pixel values of the object.

of probe pixels which influence the measurement of σ. However, we cannot choose a smaller value of the threshold than the signal-to-noise ratio of the image.

The formulation (5.4.3) is verified using a synthetic image. An object similar to the shape of the probe is generated. Also, an uniform background is chosen for this object. Such a synthetic image is blurred with a Gaussian function with a spread σ. The variances of the pixel values with background (σ_1^2) and without background (σ_2^2) are calculated. Then, the value of σ is estimated using (5.4.3). The theoretical and estimated values of σ are given in the Table 5.1. This method of measuring blur is a useful alternative to the one described in Chapter 4. However, our goal is to achieve fine control of movement for the touch operation. Touch (contact with the wafer) takes place when σ=0. But, the Gaussian distribution is not valid at this value of σ. Hence, it is not possible to detect touch by measuring σ. Therefore, we make use of an experimental observations and the variance of intensity values of the blurred probe to detect touch. This is discussed next.

5.5 Detection of Touch

The primary goal of our computer vision system is to provide controlled guidance of the probe so that the probe makes a soft contact with the wafer. As a first step, the probe is brought to the vicinity of the wafer using the method described in Chapter 4 [48]. When the probe is very close to the wafer, the pixel values provide very useful information about the proximity of the probe to the wafer. As we showed in Section 5.3 the variance of the pixel values of the blurred probe is directly proportional to the distance of the probe from the wafer surface. So, the value of σ_2^2 decreases as the probe approaches the wafer. After touching the surface, the probe slides on the surface and the variance increases. The reason for this phenomenon is explained below.

In our experimental set-up, when the probe is approaching the wafer, the microscope is focused on the wafer surface. Hence, the image of the probe is defocused while the wafer surface remains in focus. The probe comes into focus at the moment of touch with the wafer. Any further drive of the probe results in sliding of the probe tip on the wafer. Since the probe is at an angle to the optical axis, this sliding of the probe on the wafer will bring a larger area of the probe into the image. Also, the tip of the probe which is in contact with the wafer surface will be in focus while the other end of the probe in the image will be out of focus. When the probe starts sliding, the tip will remain focused but a larger defocused region appears. Therefore, the variance of the pixel values of the probe increases. We have used this observation for detecting touch. The variance is calculated from the image data and a dip in this value corresponds to the instant of touch.

A sequence of images taken when the probe is approaching the wafer is shown in Figure 5.2. We have separated the blurred probe from the background using the algorithm in Appendix G. The change in the mean and the variance of the pixel values as the probe approaches the surface are shown in Figures 5.3 and 5.4.

5.6 Conclusions

An important task in automating wafer probing is to detect a touch of the probe with the wafer. So measurement of the distance from the probe to the wafer is essential for the guidance of the probe to the target pad. In this chapter, we have described an algorithm that can be used for fine control of the probe movement when the probe is very close to the wafer (σ less than 3 pixels). We have considered the variance of the pixel values of the blurred probe and used this information for the guidance of the probe. Since the main objective of this guidance is to achieve non-destructive contact, we have concentrated our efforts on the detection of touch. To this end, we have utilized the experimental observation that the probe slides after touching the wafer surface. This corresponds to a dip in the value of the variance at the instant of touch (see Figure 5.4). The variance of the probe pixel values decreases as the probe gets into focus. Any slide after the touch will result in a larger blurred region in the image and the variance starts increasing again. For example, Frame 4 in Figures 5.3 and 5.4 is the instant of touch. The accuracy of this procedure is in the order of a few pixels (equal to the signal to noise ratio of the image). This is well below the maximum acceptable slide ($\approx$ 32 pixels, found experimentally).

The change in variance with distance agrees well with the theoretical analysis described in Section 5.3. We have tested the algorithm in Appendix G for detection of touch for different types of tips such as tungsten and tungsten carbide and with different NMOS and CMOS processes. We have also used commercial dies, such as the Motorola 6805 and Intel 2708. We have detected touch successfully for all these varying processes, pad sizes and probe tips.

As a by-product of our touch procedure, we have derived a formula for estimating the distance of the probe using the two different variance calculations. The two variances

σ_1^2 and σ_2^2, are evaluated as in Sections 5.2 and 5.3. Next, the valuc of σ (the standard deviation of the line spread function) is expressed in terms of the two variances, and the size of the object. However, there are some limitations to the measurement of distance using this method:

i. A uniform background is assumed in this method and hence this is not a solution for the general problem.

ii. A single object of uniform intensity is assumed and this may not be true in some applications.

iii. The accuracy of the measurement of blur using this technique depends on the value of α, which in turn depends on the value of the threshold in the algorithm given in Appendix G.

However, this method can still be applied by constraining the measurement to a uniform region of the background with a homogeneous probe.

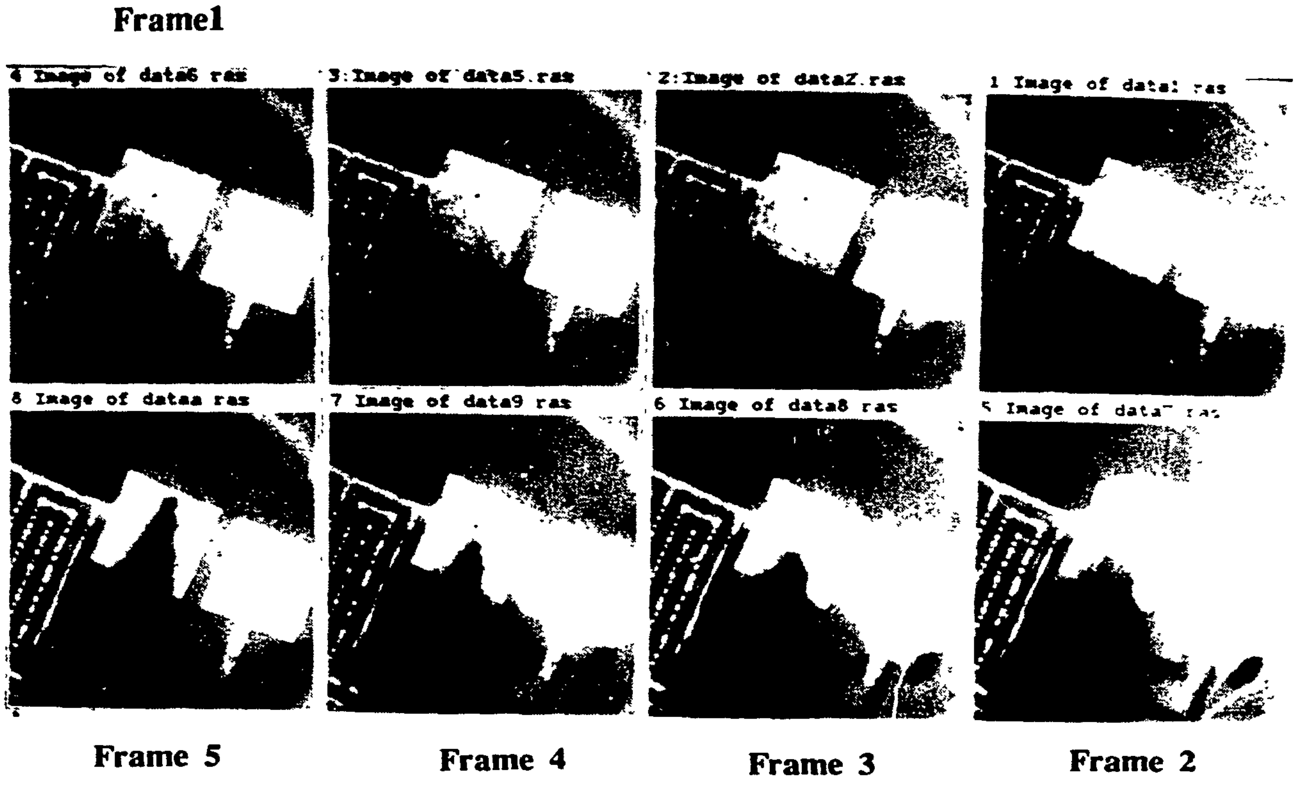


Figure 5.2 Images of the proɔe approaching and touching a pad.

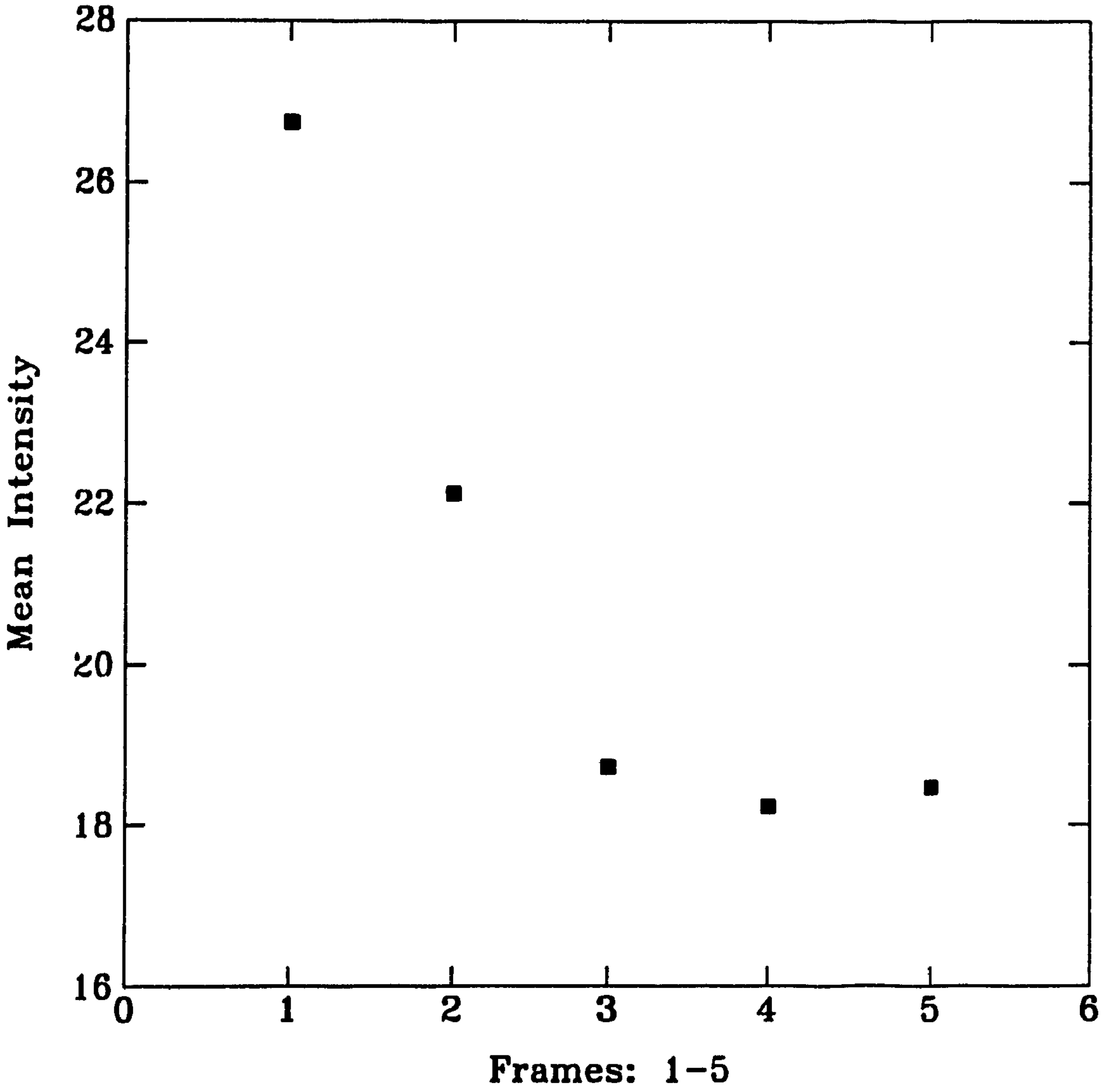


Figure 5.3 Mean of pixel values of thc probe approaching and touching a pad.

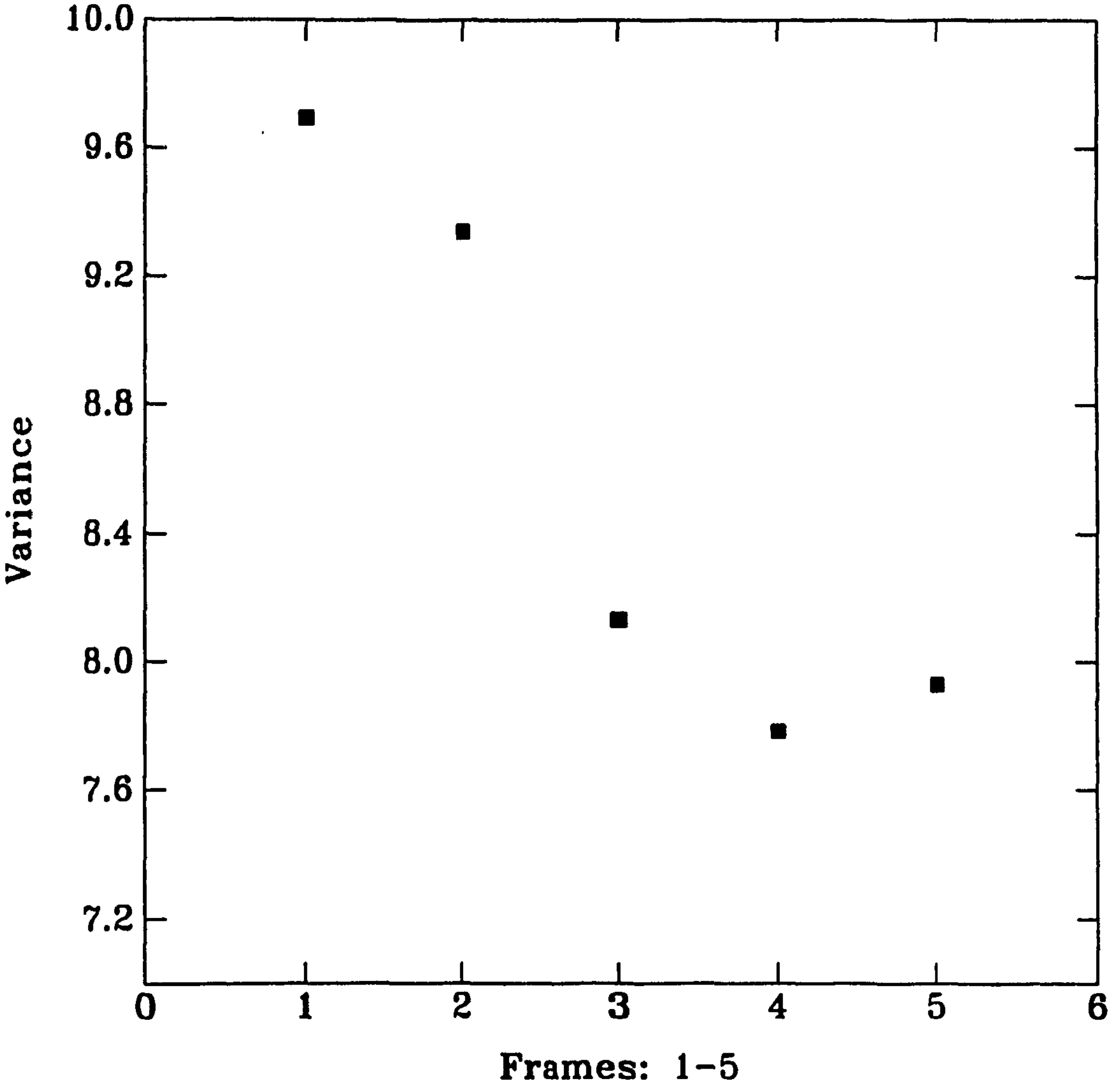


Figure 5.4 Variance of pixel values of the probe approaching and touching a pad.

Chapter 6
Low Level Vision Algorithms on Homogeneous Multiprocessor

6.1 Introduction

Many applications in image processing require a large amount of computation especially if they involve anything more than the most elementary image processing techniques. This is because image processing and pattern recognition algorithms in general are very computation intensive. Multiprocessor implementations offer the possibility of making these algorithms useful for practical, real-time applications. The complexity of such algorithms is in general a function of the size of the problem and the number of transfers between the processes cooperating in the solution. Thus a structure on which such computations can be performed efficiently requires the availability of communication pathways linking processors in a pattern that matches the one imposed by the algorithm chosen. In the following section we describe some multiprocessor architectures suitable for image processing.

6.1.1 Speed Requirements for our Computer Vision System

The computer vision system described in this thesis is required to meet the timing requirements of the wafer probing operation. These requirements are two fold. The first is related to the response time of the probes (a HOP2000 remote controlled probe manufactured by Wentworth Inc.) for each probing operation. The probe has travel speed in the range of 2 to 20 microns per second. As described in Chapters 4 and 5, we use a two stage (coarse and fine) control scheme for the guidance of the probe toward the target pad. The high travel speeds (around 20 microns per second) of the probe are suitable for coarse movement and the low travel speed range (around 2 microns per second) is suitable for the fine movement. So the sensory (vision) feedback should be available within this time frame. The second timing requirement is related to the speed of probing which, in turn, is related to the overall speed for testing a die. A typical wafer test involves checking various semiconductor parameters of the process and design specifications. To achieve this objective, we need to conduct several tests, each test involving the probing of several pads. Thus, the total time required for probing is an important consideration in the automation of wafer probing. A human operator takes 5 to 10 seconds for each probing operation. An automated probing operation should be faster than this.

The various vision algorithms used in our system are described in Chapters 3-5. The execution times (CPU time on a SUN 3/60) for these algorithms are given in Table 6.1. For each probing operation, the following sequence of operations are executed:

i. Smoothing, histograming, thresholding, edge detection and other special purpose algorithms (i.e. all the low level vision algorithms) described in Chapter 3.

ii. Estimation of proximity using the algorithm presented in Chapter 4.

iii. Detection of touch using the algorithm described in Chapter 5.

By analyzing the timing requirements of the above operations in Table 6.1, we find that the low level algorithms require the most time. Also, these algorithms perform neighborhood operations. Therefore it is likely that these operations will speedup through the simultaneous execution of the algorithms on different parts of the image. Such a task can be easily achieved using multiple processors.

In this chapter, we describe potential applications of the Homogeneous Multiprocessor [55], [56] and [57] in low level image processing. Initially, we shall review the use of multiprocessors in computer vision (Section 6.2). Next, we describe the parallelization of some well known low-level vision algorithms and the performance of these

Algorithm	CPU time (sec)
Eliminating background traces	15.559
Smoothing	2.569
Edge detection	4.439
Interpolating the edge-map	0.02
Chaining of edges	0.440
Finding points on the perpendicular (for 49532 points)	7.82
Finding slope of the plane (for 116 surfaces)	0.52
Reading a raster file and converting to a 2D array (a raster image of 256x256x8)	0.44
Converting 2D array to raster format and writing on to a file (a raster image of 256x256x8)	0.759

Table 6.1 CPU time on the SUN 3/60 for the vision algorithms.

algorithms on the Homogeneous Multiprocessor (Sections 6.4.1, 6.4.2 and 6.5). The performance was obtained analytically, and through simulation experiments on the existing simulator for the multiprocessor [58]. The results obtained by both the methods are in close agreement (Sections 6.4.3 and 6.5).

6.2 Use of Multiprocessors in Computer Vision

Several multiprocessor architectures have been proposed in recent years [55], [49]—[54]. Partitionable SIMD/MIMD Architecture (PASM) proposed by Siegel [49] was one of the first generation multiprocessors and was targetted at image processing. Several low level algorithms, like convolution, histogram generation and edge detection have been implemented on this architecture. In recent years, three kinds of architectures have become popular for the implementation of parallel algorithms for image processing. These are: i) the Mesh connected architecture, ii) the Pyramid architecture and iii) the Hypercube architecture. A survey of these architectures and their performances related to several image processing algorithms is reported in [59]. The performance of an architecture depends on the interconnection mechanism between the processors and the complexity of the vision algorithms. The computational requirements for low and high level vision algorithms are different. These have been discussed in great detail by Cypher and Sanz [59]. We shall briefly review the performance of these architectures with reference to low level algorithms.

A mesh connected computer is an array of processing elements (PEs) where each PE consists of a processor and an associated memory. Each PE has a communication link with four PEs: above, below, to the left and to the right. One example of this architecture (CLIP4) was built by University College, London [60]. The greatest strengths cf the mesh connected computer are its ease of construction and its performance on local neighborhood operations. But, the main drawback is its poor performance in solving problems involving global communication. Hence, this architecture is not suitable for high level vision algorithms [59].

Another popular architecture is the pyramid architecture. A pyramid machine consists of $\left(\frac{1}{2}logN + 1\right)$ levels, where the ith level, $0 \leq i \leq \left(\frac{1}{2}logN\right)$, is a mesh connected computer with $N/4^i$ PEs. Each level has connections to the levels above and below, giving 9 connections: 4 to its children in the level below, 4 to its nearest neighbors at the same level, and 1 to its parent in the level above. One example of the pyramid architecture is the HCL Pyramid, built by the University of Washington, Seattle [61]. Pyramid computers are more difficult to build than mesh computers [59]. They have similar performance to mesh architectures but have advantages over the mesh architectures in region growing algorithms, where the image is analyzed at different resolutions. Tanimoto [61] studied the problem of bright spot detection, where a relatively bright region of the image must be located so that it can be used as a seed for a region growing operation. In this application, the bottom most level of the pyramid holds the image and each level of the pyramid contains the same image with different resolutions. This algorithm takes *logN* (*N* is the image size) time. Tanimoto [61] also computed histograms on a pyramid, which involves performing a *log(N)* time "pass" from the bottom of the pyramid to the top for each gray level value to be computed. Again, this architecture is not very suitable for high level vision algorithms [59].

Both mesh and pyramid architectures are very suitable for neighborhood operations. However, there are several algorithms where data transfers to distant processors take place. One multiprocessor which is useful in such applications is the hypercube architecture. A hypercube multiprocessor consists of $N=2^n$ (*N* is the image size) processing cells interconnected as if each were located at one vertex of an *n*-dimensional hypercube, so that two cells share a direct connection if and only if their corresponding hypercube vertices are connected by a hypercube edge. One such multiprocessor is the Connection Machine, which is an SIMD hypercube architecture and was built in 1986 by Thinking Machines Corporation. The hypercube computer is better suited to high level computer vision because communication with distant nodes is easy in this architecture. The reason is that each processor is connected to *log(N)* processors. Some interconnection networks for this purpose are described in [59]. A final observation is that each of the above mentioned architectures can simulate the other architectures and this helps in designing ideal algorithms for specific architectures [52]

In this chapter we shall investigate the application of the Homogeneous Multiprocessor [55, 58] which is a closely coupled MIMD architecture that provides nearest neighbor communication. Execution of image processing algorithms such as smoothing, edge detection and relaxation benefit from the use of such an architecture.

6.3 The Homogeneous Multiprocessor and the H-Network

As shown in Figure 6.1, the Homogeneous Multiprocessor [55] is a tightly coupled MIMD architecture, composed of *k* *(k>3)* processing elements, *k* memory modules, *k+1* interbus switches s_i isolating the processing elements from each other and the H-Network which is a fast local area network used for point-to-point and broadcast mode communications. The architecture can be considered to be composed of two parts: namely the Homogeneous Multiprocessor Proper incorporating the processors and the interbus switches, and the H-Network [57].

Each processing element P_i owns its local memory module M_i and accesses it via its local bus b_i; it also has the exclusive use of the respective network station HS_i. Local buses are separated by the intervening switches s_i. These switches provide each processor P_i with the ability to access the memory modules of either one of its two immediate neighbors by requesting the appropriate switch to close, thus creating an "extended bus". Also, for I/O or data transfers to and from distant processors, each processor may utilize the H-Network. Although the H-Network could have been used for data transfers to and from distant processors (especially in histogram calculations), the results reported in this work were obtained through the utilization of the "extended bus" communication mechanism. It was felt that this mechanism was considerably faster for short to medium distances since no task switching is involved. Nevertheless, we expect further improvement in the performance by employing the H-Network for long distance transfers when it becomes operational. The Homogeneous Multiprocessor is currently under implementation using the 8–MHz MC68000 processor. The performance results reported in this study therefore pertain to an implementation of the multiprocessor using the aforementioned processor. Before describing the results of our implementation, we will discuss the need for speeding up the execution of the vision algorithms in our system.

6.4 Neighborhood Operations

The Homogeneous Multiprocessor, being a closely coupled MIMD architecture, is perfectly suited for context dependent algorithms. The often used image processing algorithms like smoothing, histogram computation, and edge detection are considered here. In the following sections wc present the method we used to run these algorithms on the Homogeneous Multiprocessor as well as results showing their performance.

6.4.1 Image Averaging

Averaging (smocthing) operations [30] are primarily used for diminishing noise. The raw image usually has sharp edges, but it is also noisy and may contain small spiky artifacts. One of the most commonly used algorithms for smoothing is local averaging. Given an $M \times M$ image *f(x,y)*, the smoothed image *g(x,y)* is obtained by averaging the grey level values of the pixels of the original image contained in a predetermined neighborhood *S(x,y)* of *(x,y)*.

$$g(x,y) = \frac{1}{n} \sum_{i,j \in S(x,y)} f(i,j) \qquad (6.4.1)$$

where n is the total number of points in the set *S(x,y)*.

The operation is performed for each pixel in the image with the possible exception of the edge pixels. We implement the smoothing algorithm on the Homogeneous Multiprocessor as follows. The $M \times M$ image is divided into N strips with each strip having $M \times M/N$ pixels. Each processor is allotted a strip of the $M \times M$ image and smoothes the pixels in the strip located in its local memory using equation (6.4.1). Each processor needs to communicate with a neighboring processor only for the calculation of the smoothed image at the boundaries of the strip. This communication is

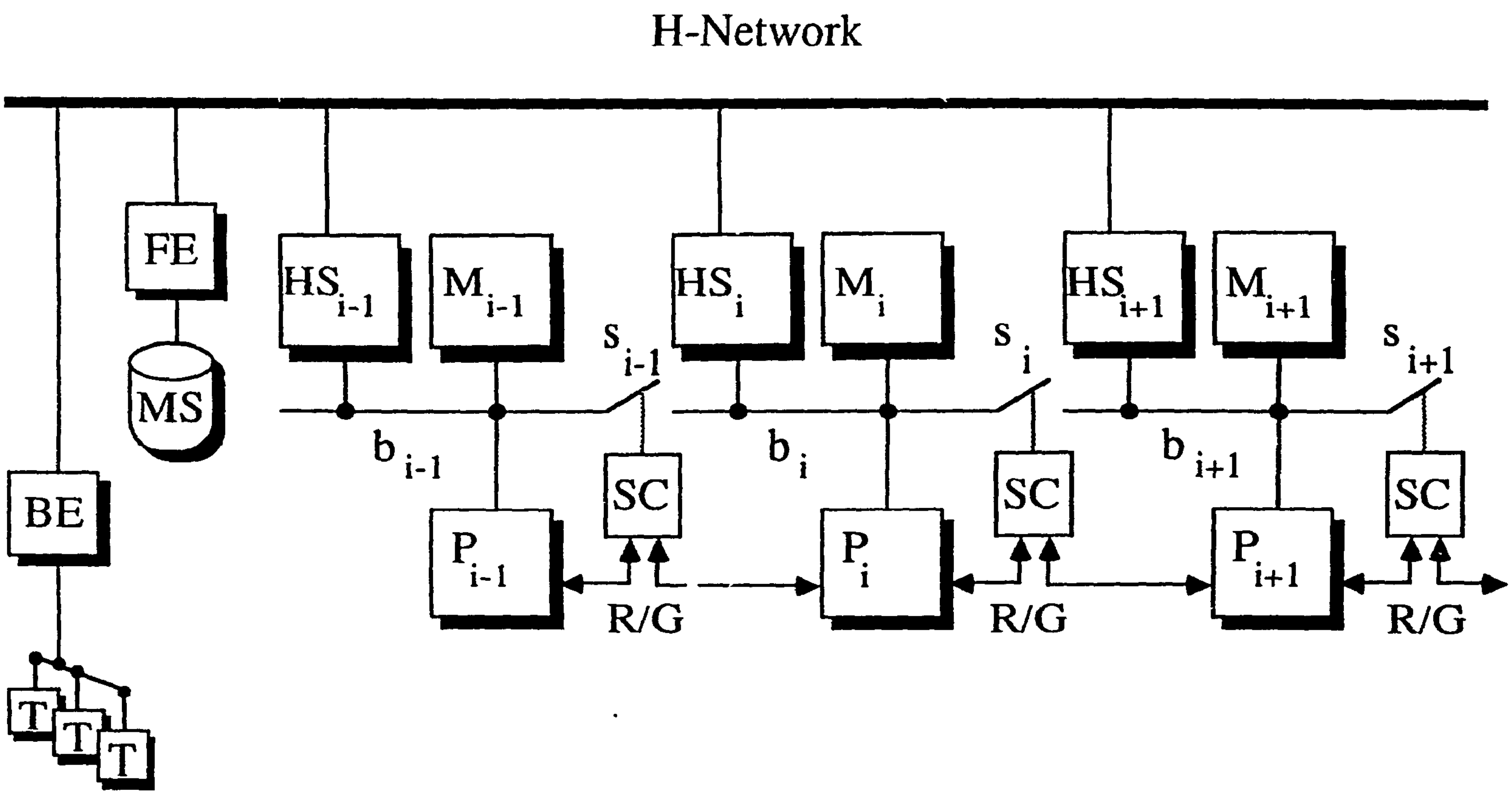


Figure 6.1 The Homogeneous Multiprocessor architecture

P:	Processor	M:	Memory Module	s:	Bus Switch
FE:	Front End	BE:	Back End	SC:	Switch Controller
b:	Local Bus	T:	Terminal	MS:	Mass Storage
HS:	H-Network Station	R/G:	Bus Request/Grant		

accomplished directly through the "extended bus" mechanism described in Section 6.3, and is transparent to the programmer. Figure 6.2 shows the plot of the speedup factor obtained against the number of processors involved in the computation for two image sizes. The number of grey levels is fixed at 64 for both cases.

6.4.2 Edge Detection

Edge detection plays an important role in segmenting an image. Some of the edge detection algorithms are very attractive for parallel implementation, e.g. mask operators. In fact, we have used a mask operator, the Laplacian, for the edge detection in our computer vision system. This operator is omni-directional and hence useful in detecting edges in all directions. But, there are also directional mask operators for edge detection which are effective in identifying objects in automated visual inspection [26]. One such operator, the Sobel, estimates the partial derivatives in four directions (one operator for each direction), and is given in Figure 6.3. For edge detection, the response in all four directions is computed and the one giving the maximum response is considered as the edge direction. Hence, the Sobel operator is more time consuming than the Laplacian. In order to evaluate the performance of the Homogeneous Multiprocessor for edge detection, we have considered the Sobel operator. The implementation details are given in this section.

Each edge detector is described by a set of templates whose application on an intensity function *f(x,y)* results in a set of gradient arrays. The above procedure can be implemented on the multiprocessor as described below. The $M \times M$ image is divided into N strips having $M \times M/N$ pixels. Each processor calculates the gradient arrays of its strip, i.e., the corresponding edge enhanced values of each pixel. In a similar fashion as in smoothing, the edge enhanced values of the strip boundaries are calculated based on the values of the pixels in the neighboring processors. This is repeated for all the directional mask operators. The partitioning algorithm is similar to smoothing. Figure

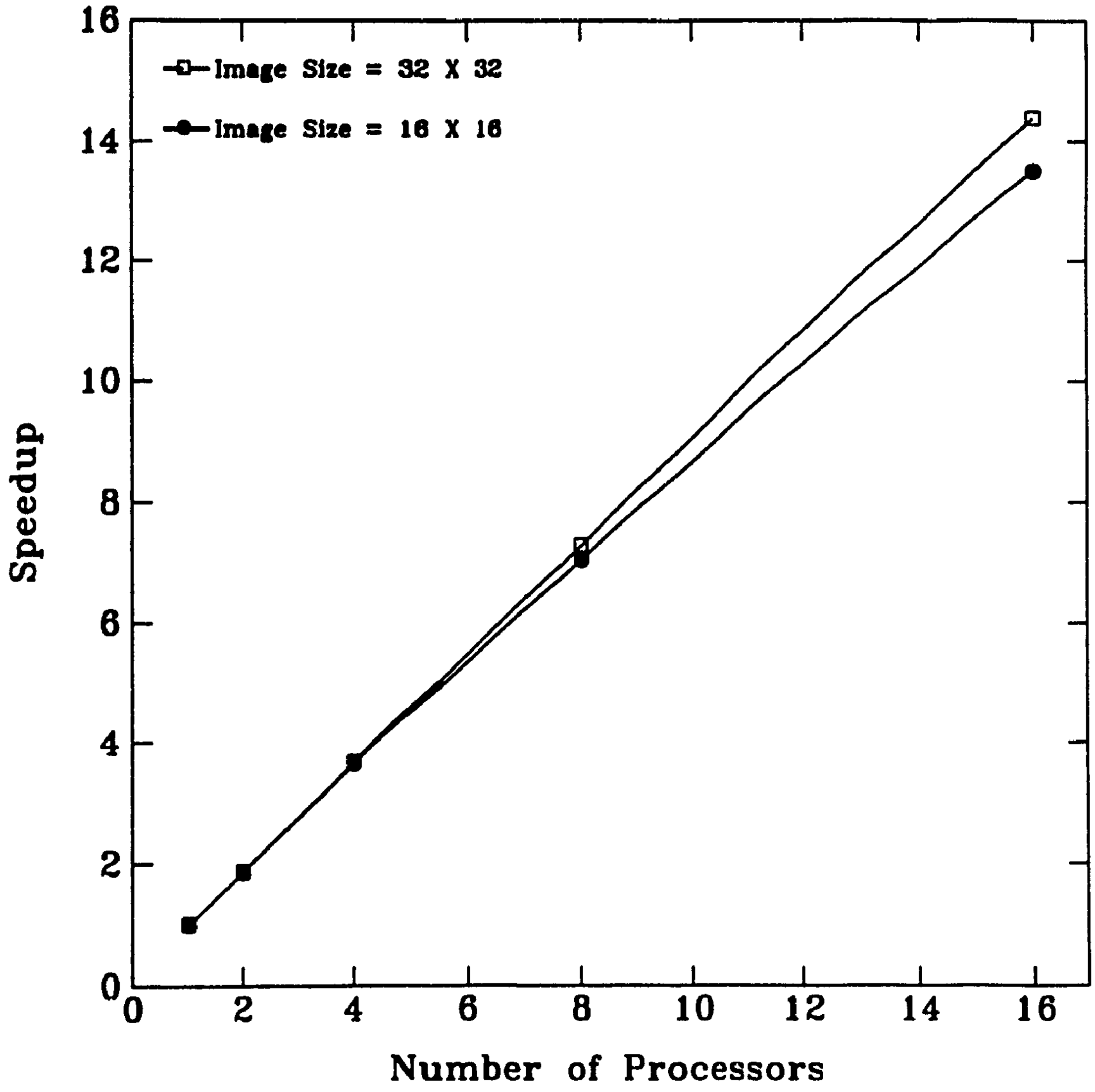


Figure 6.2 Speedup vs. the number of processors for the distributed averaging algorithm as obtained through simulation.

6.4 shows the speedup factor against the number of processors for two image sizes. The number of grey levels is fixed to 64 for both cases.

6.4.3 Speedup Calculation for Local Operations

Several image processing algorithms can be executed by repeating a set of operations on a well defined neighborhood of each pixel. Examples are the local averaging and edge detection algorithms mentioned previously. These algorithms can be effectively ported to the Homogeneous Multiprocessor by assigning to each of the N nodes a strip of *MxM/N*. If we denote by τ_{op} the time necessary to perform the set of operations on the pixel neighborhood, by τ_{tr} the time needed to transfer a single pixel from a neighboring memory module, and by α the percentage of pixels in the neighborhood of a pixel lying at the edge of a strip, then we can estimate the achievable speedup as follows. The time to carry out the algorithm on a uniprocessor is given [62] by $T = M^2\tau_{op}$, while the execution time for the Homogeneous Multiprocessor is $T_{PAR} = \frac{M^2}{N}\tau_{op} + 2\frac{M^2}{N}\alpha\tau_{tr}$. Therefore, the speedup is given as

$$S = \frac{T}{T_{PAR}} = \frac{M^2\tau_{op}}{\frac{M^2}{N}\tau_{op} + 2\frac{M^2}{N}\alpha\tau_{tr}} \tag{6.4.2}$$

It is easy to see that the speedup equal to $\frac{N}{1+2\alpha\frac{\tau_{tr}}{\tau_{op}}}$.

6.5 Histogram Generation

A histogram of grey level content provides a global description of the appearance of an image. Histograms are frequently used in thresholding images while interactive histogram modification is useful for enhancing the picture quality. We have used histograms of grey values for the thresholding operation. This was discussed in Chapter 3. In this section we describe the implementation details of the histogram algorithm on the Homogeneous Multiprocessor.

$$E = \begin{bmatrix} -1 & 0 & 1 \\ -2 & 0 & 2 \\ -1 & 0 & 1 \end{bmatrix} \qquad W = \begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$N = \begin{bmatrix} 1 & 2 & 1 \\ 0 & 0 & 0 \\ -1 & -2 & -1 \end{bmatrix} \qquad S = \begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

Figure 6.3 The Sobel mask operators for east, west, north and south.

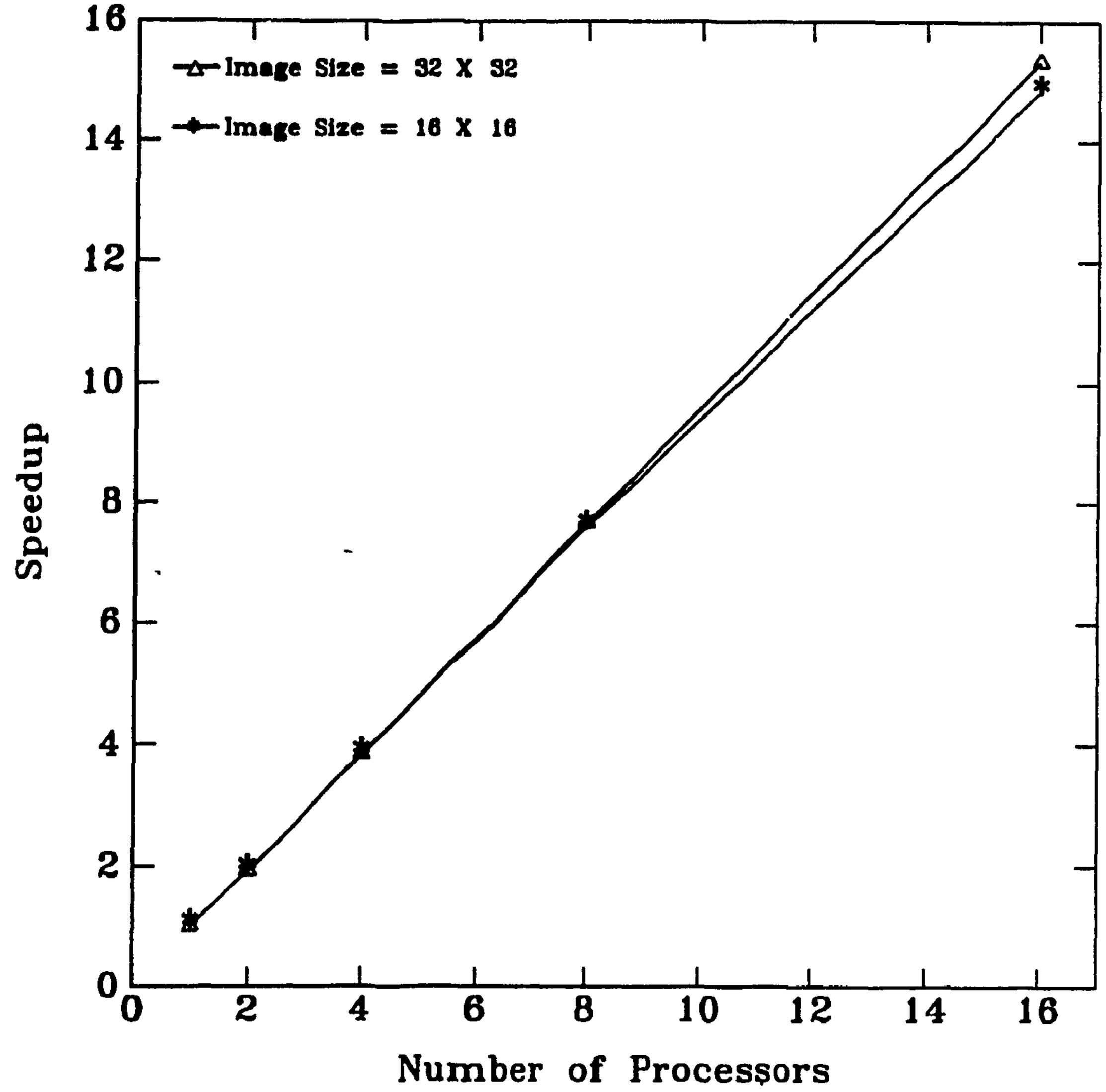


Figure 6.4 Speedup vs. the number of processors for the distributed edge detection algorithm (using the Sobel operator) as obtained through simulation.

It is assumed that there are $N\left(N=2^{k}\right)$ nodes in the Homogeneous Multiprocessor. The image is divided into N strips. Each strip is loaded in the memory of each of the nodes, which calculates the partial histogram of the region assigned to it.

The next step accomplishes the merging of the partial histograms. This is done through a form of recursive doubling [49], [63], [64]. Suppose that there are B grey levels in the image. Initially, processors $P_{2l+1} : l = 0,1,2,.....(N/2-1)$ merge the B/2 least significant bins of the partial histograms contained in their own memory as well as those of their neighbors to the right. Similarly, processors P_{2l+2} merge the B/2 most significant bins located in their own memory as well those of their neighbors to the left. At the end of these operations, nodes P_{2l+1} hold the least significant halves of the merged partial histograms, while their neighbors to the right (nodes P_{2l+2}) hold the most significant halves.

Next, nodes $P_{4l+1}; l = 0,1,2,...(N/4-1)$ transfer the B/2 least significant bins of their merged histograms to nodes P_{4l+2}, and similarly nodes P_{4l+4} transfer the B/2 most significant bins to nodes P_{4l+3}. At this point, nodes P_{4l+2} and P_{4l+3} contain partial B bin histograms and the process is repeated. The final completed histogram is to be found in node $P_{N/2}$. Under this algorithm, the partial histograms are merged on a tree structure of processors embedded on the Homogeneous Multiprocessor as depicted in Figure 6.5.

Observe that, as the process continues, partial histograms are located in nodes that are progressively further away from their neighbors. Their merging requires the transfer of data between distant processors. This is done through the implementation of a form of a "bucket brigade" to efficiently transfer long vectors between distant processors. Thus, in order to transfer a B-bin vector from processor P_i to processor P_j, the intervening processors form a pipeline through which the B-vector is transferred in *O(j-i+B)* steps. Since neighboring nodes communicate by accessing each others memory modules, it is possible to form a pipeline consisting of alternate nodes. Each node in the pipeline moves data from the memory module of its left neighbor to the memory module of its right neighbor. The following example will clarify the strategy. Consider the transfer of a vector of B elements from node 5 to node 8 (Figure 6.5.). The transfer is accomplished by forming a two stage pipeline consisting of nodes 6 and 8 (In this case node 8 transfers data to itself since it is the last stage of the pipeline). Thus node 6 transfers data from the memory module of node 5 to the memory module of node 7 while node 8 completes the pipeline by transferring the data from the memory of node 7 to its own. The total transfer requires $B+2-1$ steps. This result can be generalized as $B+[(j-i)/2]-1$ for transfers between nodes i and j.

The histogram merging algorithm outlined above is carried out in $log(N)$ iterations. Each iteration consists of a partial merging step requiring $B/2$ additions and $B/2$ transfers together with $B/2$ transfers needed to locate the merged histogram in the appropriate node. Iterations 2 to $log(N-1)$ require the pipelining of intermediate nodes and each iteration requires $B + 2^{(m-2)} - 1$ transfers (m is the iteration number). Thus, if we denote by τ_a the time required to perform a single addition, and by τ_{tr} the time required for a single transfer from the memory module of the left neighbor to that of the right neighbor. Then the time required for the histogram merging algorithm can be calculated as* [62].

$$\begin{aligned} T_{MRG} &= \sum_{m=1}^{log\,N} (\tau_a + 2\tau_{tr}) B/2 + \tau_{tr} \sum_{m=2}^{log\ N-1} \left[B + 2^{(m-2)} - 1\right] \\ &= \left(\tau_a \frac{B}{2}\right) log\ N + \tau_{tr} \left[\frac{N}{4} + (2B-1)\,logN - 2B + 1\right] \end{aligned} \qquad (6.5.1)$$

Now, given an MxM image, the time required to obtain the N partial histograms on a Homogeneous Multiprocessor consisting of N nodes is given as $T_{HIST} = \frac{M^2}{N}\tau_a$. Thus, the total time required for the parallel algorithm is obtained as

* It is assumed that the time spent to merge each of the $B/2$ bins of two partial neighboring histograms is $\tau_a + \tau_{tr}$ which corresponds to the transfer of one bin from the neighboring memory module and the addition to the corresponding bin in the local memory module.

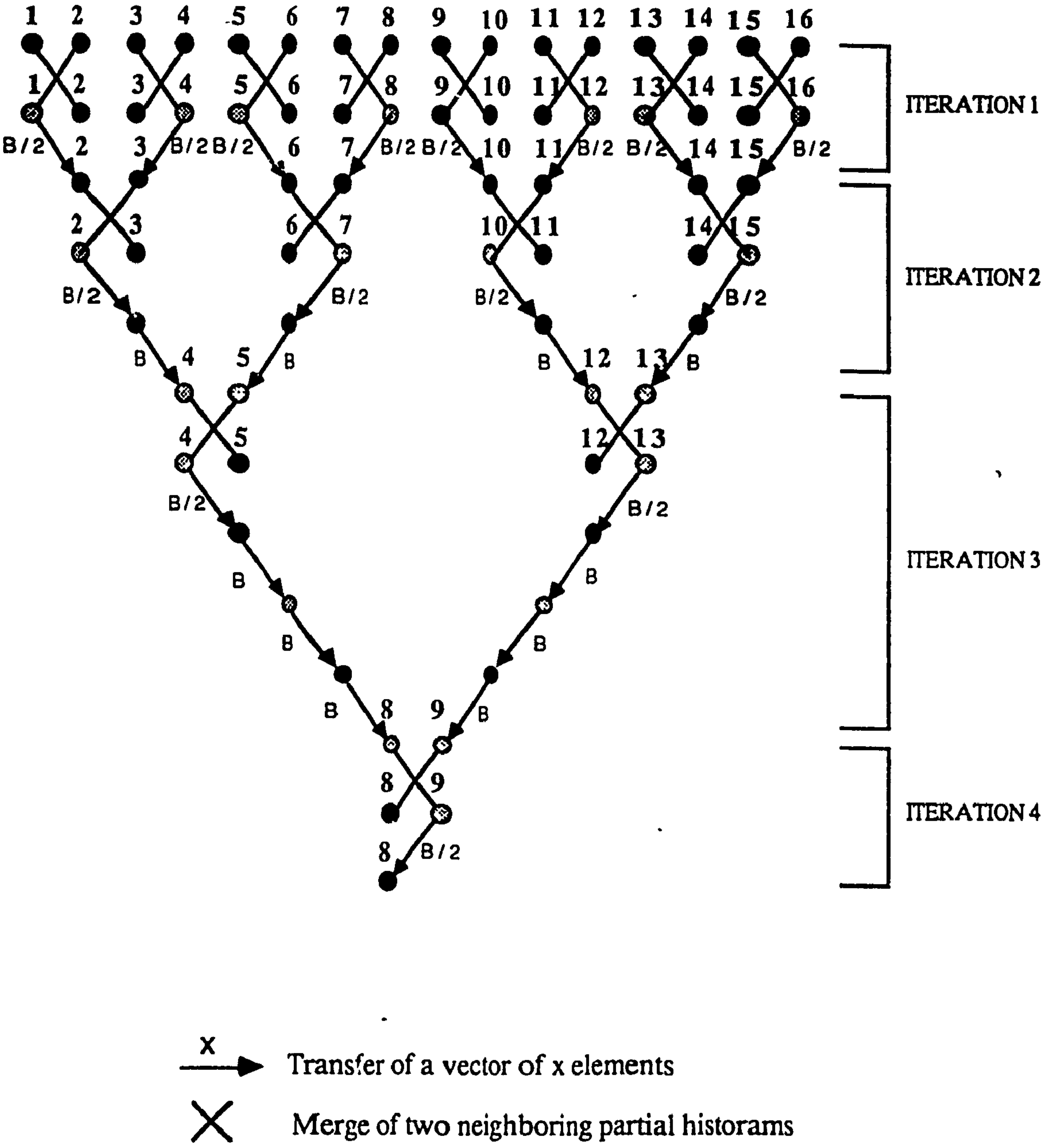


Figure 6.5 An example of the distributed merge algorithm on sixteen processors.

$$T_{PAR} = T_{HIST}+T_{MRG} = \tau_a\left(\frac{M^2}{N}+\frac{B}{2}logN\right)+\tau_{tr}\left[\frac{N}{4}+(2B-1)\,logN-2B+1\right] \tag{6.5.2}$$

Similarly, the total time required to obtain the histogram of an *MxM* image on a uniprocessor is given as $T = M^2\tau_a$. The speedup factor can therefore be obtained as

$$S = \frac{T}{T_{PAR}} = \frac{M^2\tau_a}{\tau_a\left[\frac{M^2}{N}+\frac{B}{2}logN\right]+\tau_{tr}\left[\frac{N}{4}+(2B-1)\,logN-2B+1\right]} \tag{6.5.3}$$

$$= \frac{M^2}{\left[\frac{M^2}{N}+\frac{B}{2}logN\right]+r\left[\frac{N}{4}+(2B-1)\,logN-2B+1\right]} \tag{6.5.4}$$

where $r = \frac{\tau_{tr}}{\tau_a}$ is the ratio of the transfer over the add times. The efficiency of the algorithm is given by

$$e = \frac{S}{N} = \frac{M^2}{\left[M^2+\frac{B}{2}NlogN\right]+r\left[\frac{N^2}{4}+(2B-1)\,NlogN+N\,(1-2B)\right]} \qquad .5.5)$$

Figure 6.6 shows the speedup factor obtained both through simulations and equation (6.5.3) plotted against the number of processors for histogram calculation for images with varying sizes. The number of grey levels used is set at 32.

6.6 Conclusions

The final goal of the computer vision system described in this thesis is to attain real-time operation of wafer probing. Towards the achievement of this goal, we have investigated the parallelization of low level algorithms. We have used the Homogeneous

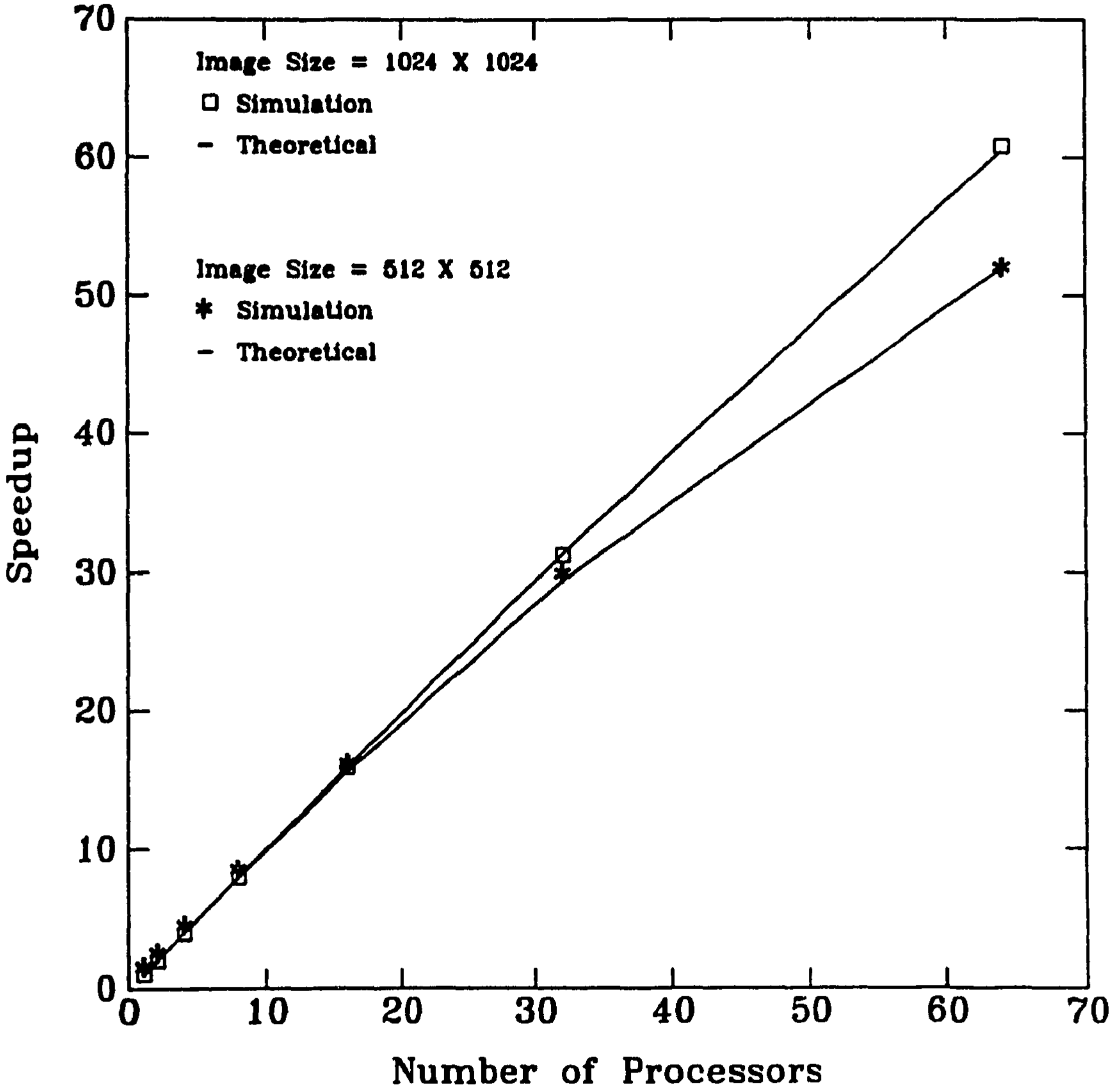


Figure 6.6 Speedup vs. the number of processors for the distributed histogram algorithm as obtained through simulation and analysis.

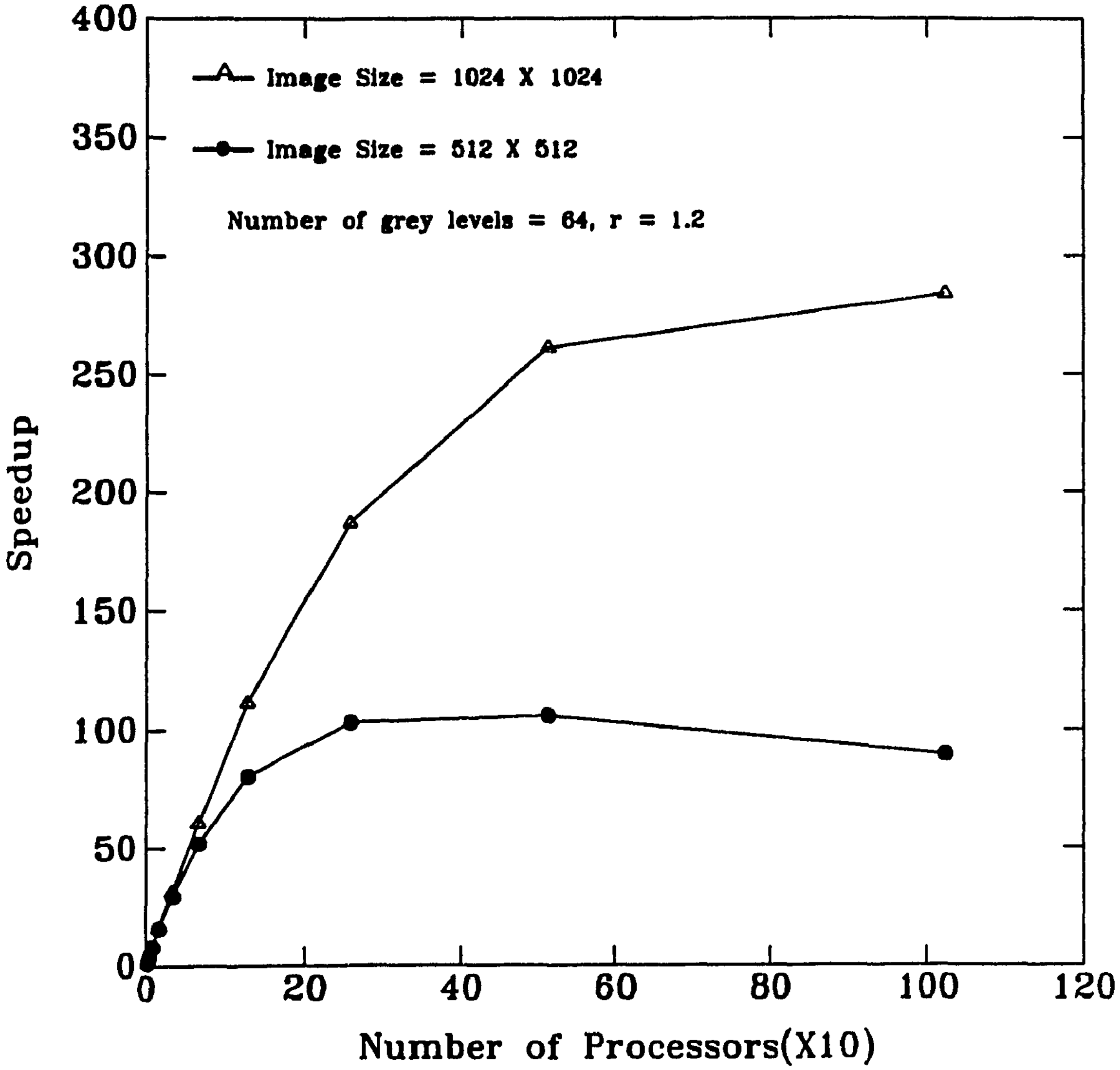


Figure 6.7 Speedup vs. the number of processors for the distributed histogram algorithm as obtained through analysis.

Multiprocessor for this investigation. In this chapter we presented the performance of the following low-level vision algorithms: (i) Smoothing, (ii) Edge Detection and iii) Histogram Generation. The results were obtained through simulation experiments run on the simulator developed for the Homogeneous Multiprocessor and are presented in Figures 6.2, 6.4 and 6.6.

As can be seen in Figures 6.2 and 6.4 both smoothing and edge detection algorithms show almost linear speedup with the number of processors involved. This is to be expected, since these algorithms require very little interaction between processors, apart from the occasional exchange of the values of the pixels located at the boundaries of the strips allocated to the processors. This is also in agreement with (6.5.2) which describes the behavior of the speedup for local algorithms implemented on the Homogeneous Multiprocessor.

Also, almost linear speedup was obtained for large images for the histogram algorithm presented in Section 6.5. Figures 6.6 and 6.7 show substantial agreement between the theoretical calculation (as given by equation (6.5.2)) and the simulated results.

On the other hand, as can be seen from Figure 6.7, the speedup for the histogram generation algorithm slows down and actually reverses itself when a large number of processors is used. This was also to be expected, since the algorithm used for the distributed merging of the partial histograms requires O(n) transfers. Hence for every bin size there exists an optimum number of processors beyond which the speed-up drops.

Our distributed merge algorithm can be improved if the pipelined transfer of the partial histograms is started immediately after the merging of the neighboring histograms. Further improvements may be achieved if the H-Network were to be used instead of the bucket brigade scheme for long range transfers. This transfer mode would become efficient when the transfer time over the H-Network becomes less than the transfer time obtained through the bucket brigade mechanism.

Nevertheless we were able to show substantial speedups for both cases of local algorithms (such as local averaging and edge detection) as well as nonlocal ones (e.g. histogram generation).

The analysis and simulations carried out in this work helped us to understand the performance of the Homogeneous Multiprocessor for image processing, parallel implementation of image processing algorithms for a given architecture, and selection of an optimum number of processors for a given operation. In addition, the use of a multiprocessor can be beneficial for the real-time execution of the control algorithms and trajectory planning algorithms required for the micro-manipulator.

Chapter 7
Conclusions and Further Work

The semiconductor industry is one of the principal beneficiaries of computer vision research. It is also a major buyer of vision hardware and software. The high precision required by this industry can only be achieved using high resolution optics and vision systems, particularly since the introduction of VLSI. The increased complexity and small feature size of VLSI circuits have created challenging research problems in several manufacturing areas. Machine vision has been successfully applied to wafer inspection, die and wire bonding, and testing. However, there are several areas such as automatic wafer probing where further research of a more interdisciplinary nature is required. In wafer probing, metallic probes are used to contact the surfaces of the wafer. Compliant and reliable probing is a highly skilled operation which has to be repeated with good accuracy. In this chapter, we summarize the contributions of this thesis and discuss the practical applicability of the proposed methods. The application we have chosen involves multiple disciplines and further work is required towards realizing an end product of this research, as we shall discuss in the latter part of the chapter.

This thesis describes a method for automating the wafer probing operation. The primary objective of this research was to investigate various aspects of guiding a probe to its metal target pad. Another objective was the detection of probe contact for non-destructive probing. In addition, it was also intended that the computational time required for these solutions be within the reach of real-time operation. All these objectives were achieved through analytical solutions of the problem and by experimentation with real data.

The contributions of this thesis are two-fold. So far, to our knowledge, no one has reported automatic wafer probing using sensory feedback. Probing the wafer without damaging the wafer surface and the probe tip, is a major difficulty in wafer probing. We have solved this problem using vision as feedback. Hence, the first contribution of this thesis is a step towards the development of completely "hands-off" wafer probing system. The second contribution is a simple and workable solution to the problem of depth perception in a scene with a single point of view. This method is new and significantly more robust in comparison to existing methods. In addition to the above mentioned contributions, this thesis investigated the use of a multiprocessor to achieve real-time operation.

Our experimental set-up consists of a probing station, a camera, an image grabber, and a computer for processing the images. We selected the Wentworth VLSI test station for our experimental set-up. We have modified the optics of the station and also used a solid state camera for capturing images of the VLSI patterns. We developed an in-house frame grabber for the camera. Due to budget constraints we have developed a custom-made image grabber for image acquisition. A user-friendly image processing software package was developed in our laboratory. This software package was very useful at various stages of the development for manipulation of images.

The images from the camera contain specular noise. We eliminated this noise using the neighborhood averaging technique. We were able to separate the three regions (probe, background, and metal layer) by thresholding the smoothed image. This is possible because of high contrast between these three regions. By applying the Laplacian edge operator on the thresholded images, we were able to obtain the edge map of the probe and the metal pads. These edge maps are used for further analysis such as estimating the proximity of the probe to the the wafer and detecting the contact between the probe and the wafer. Some of the problems resulting from the physical constraints in our experimental set-up were addressed. These are unconstrained illumination, high intensity of light, specularity due to the metal layer, noise due to multiple layers of the VLSI patterns, the shadow cast by the probe, vibration of the wafer chuck, etc. Special purpose algorithms were developed to resolve these difficulties.

Accurate determination of the distance from the probe to the wafer is important for the successful guidance of the probe to its destination. We have developed a method to estimate this distance using the information from the blurring of the probe image around the edges. This method used the slope of the edge profile at the zero-crossing. A theory was formulated for this method and a closed form solution was derived to obtain a measure of the distance of the probe from the wafer. This formula was verified using experimental data. It was found that the method works very well at the large distances. Its accuracy at smaller distances is affected by quantization errors and noise in the image, but can be improved by increasing the resolution of the camera. We have also observed that the slope at the zero-crossings, for different backgrounds, varies very significantly and agrees with common experience that a blurred object looks sharper against a highly contrasting background than against a low contrast one. This type of information can be used for identifying boundaries between different regions. Another interesting observation occurs in the determination of the probe's orientation. Our human visual system cannot determine the orientation of the probe in the vertical direction by viewing along the optical axis. By calculating the blur parameter along the edge of the probe, we were able to determine the inclination of the probe with respect to the wafer surface. This interesting result may be useful in other applications of machine perception.

The final operation in the guidance of the probe is the detection of touch. We have solved this problem using a method involving an experimental observation. In this method, we used the observation that the probe slides horizontally after touching the wafer. We calculated the variance of the intensity values of those pixels consisting of the probe and noticed a dip in this value at the moment of touch. The slide can be detected within the accuracy of a few pixels. This is well within the allowable limit of accuracy (experimentally found as 0.1 μm$\approx$ 32 pixels). We have derived analytical formulations for this variance value of the probe and proved theoretically that the variance of pixel values decreases as the probe approaches the wafer. As a by-product of the above formulations, we have derived another formula for the calculation of depth. These formulations are generic and have applications where only a single view is possible.

Our ultimate goal is to use the computer vision system described in this thesis for real-time operation. The algorithms described above are computationally very expensive. We have investigated the use of the Homogeneous Multiprocessor for speeding up the low level vision algorithms. The performance of smoothing, edge detection, and histogram generation algorithms were studied by running simulation experiments on a simulator developed for the Homogeneous Multiprocessor. The algorithms for smoothing and edge detection showed a linear speedup with respect to the number of processors involved. This is understandable since these algorithms require very little interaction between processors, apart from the occasional exchange of the values located at the boundaries of the image data allocated to the processors. On the other hand, the speedup of the histogram generation algorithm slowed down and reversed when a large number of processors were involved. This is expected since the algorithm used for the distributed merging of the partial histograms requires *O(n)* transfers. Hence, for every bin size (number of gray levels in the image) there exists an optimum number of processors beyond which the speedup drops. Theoretical formulations were derived for the speedup of neighborhood operations and histogram generation. The experimental results agreed well with the theoretical formulations. These formulations will be helpful in designing parallel algorithms to achieve the desired speedup.

Further Work

Research in the development of completely automated VLSI wafer probing poses challenges in the areas of robot vision, plan generation and robot control. One aspect, measurement of the distance from the probe to the wafer, was investigated in this thesis. Future advances in the area of automation of VLSI wafer probing will require contributions from several interdisciplinary areas. The following are a few proposed areas of endeavor.

Development of a Motion Controller

Presently, human manipulation is used to control the guarded motion of the probe. A computer vision system for the measurement of distance is described in this thesis. This information can be used to control the Z-drive of the probe. Recently a computer controlled probe has been acquired and its software driver is being developed.

Development of an Intelligent Plan Generator

An expert system for vision and plan generation for the micro-manipulator is required for "hands-off" VLSI wafer probing. The expected characteristics of the expert system are the following; Given a specific test (from a menu of available tests), the present locations of the probes are identified (through vision). Then the probes are moved, suitable optical magnification is set, visual patterns are recognized, and electrical measurements are done through specific test pads. Plans like SEE, ZOOM, FOCUS, MOVE, CHECK, FIND, TOUCH, LIFT, and TEST are generated in sequence according to the goal specified by the selected test and the knowledge base. This intelligent plan generator can provide the overall strategy for generating various plans to test a VLSI chip.

Study of Vision Algorithms on the Homogeneous Multiprocessor

We have studied the implementation of some low level algorithms on the Homogeneous Multiprocessor. Further work will involve the study of the performance of the Homogeneous Multiprocessor for the remaining algorithms in the vision system. New parallel algorithms need to be developed to implement the depth perception schemes.

Study of Blurring at High Numerical Apertures

We have considered the Gaussian function as the point spread function for modeling the blurring in the microscope. The change in intensity distribution near the focal plane for high and low numerical apertures is reported by Mansuripur [65], [66]. The choice of an appropriate point spread function (PSF) for the defocusing system is important to achieve accurate measurement of the distance from the probe to the wafer. Hence, we need to obtain analytical expressions for the relationship between exact intensity distributions and distances, taking into account high numerical apertures.

Neural Networks for Depth Perception

It would be interesting to study the use of neural networks for understanding the depth perception problem. Human vision can easily identify the depth of various objects in a given scene. This is due to the training we acquire in the real world. Such training is based on a context-dependent interpretation of objects and boundaries. In our application, the human operator obtains distance information using a single view combined with his past experience. Neural networks are capable of retaining such experience through repeated training and thus provide a viable alternative to algorithmic solutions.

References

[1] K. L. Harris, P. Sandland, and R. M. Singleton, "Automated inspection of wafer patterns with applications in stepping, projection and direct-write lithography," *Solid State Technology*, pp. 159–179, February 1984.

[2] K. L. Harris, P. Sandland, and R. M. Singleton, "Wafer inspection automation: Current and future needs," *Solid State Technology*, pp. 199–205, August 1983.

[3] L. Goldstein, "Controllability/observability analysis for digital circuits," *IEEE Transactions on Circuits and Systems*, vol. CAS-26, no. 9, pp. 685–693, 1979.

[4] Y. Y. Hsieh and K. S. Fu, "An automatic visual inspection system for integrated circuit chips," *Computer Graphics and Image Processing*, vol. 14, pp. 294–343, 1980.

[5] B. K. Horn, "A problem in computer vision: Orienting silicon integrated circuit chips for lead bonding," *Computer Graphics and Image Processing*, pp. 294–303, April 1975.

[6] M. Mese, I. Yamazaki, and T. Hamada, "An automatic position technique for VLSI assembly," in *Proc. of 5th International Joint Conference on Artificial Intelligence*, pp. 685–693, 1977.

[7] M. L. Baird, "SIGHT-I : A computer vision system for automated IC chip manufacture," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. SMC-8, pp. 133–139, 1979.

[8] R. G. Bennetts, *Design of Testable Logic Circuits*. New York: Addison Wesley, 1984.

[9] C. Beddoe-Stephens, "Semiconductor wafer probing," *Test and Measurement World*, pp. 33–35, November 1982.

[10] M. L. Baird, "EYESEE: a machine vision system for inspection of integrated circuit chips," in *Proceedings of IEEE Conference on Robotics and Automation*, pp. 444–448, June 1985.

[11] A. M. Wallace, "Industrial applications of computer vision since 1982," *IEE Proceedings-E*, vol. 135, pp. 117–136, May 1988.

[12] R. T. Chin and C. A. Harlow, "Automated visual inspection: A survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-4, pp. 562–565, November 1982.

[13] H. Yoda, Y. Ohuchi, Y. Taniguchi, and M. Ejiri, "An automatic wafer inspection system using pipelined image processing techniques," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 10, pp. 4–16, January 1988.

[14] T. W. Williams, *VLSI Testing*. Amsterdam: North-Holland Publishing Company, 1986.

[15] E. J. McCluskey and F. Buelow, "IC quality and test transparency," in *Proceedings of IEEE International Test Conference*, (Washington DC), pp. 295–301, September 1988.

[16] E. Wolfgang, R. L. P. Faxekas, and H.-P. Feuerbaum, "Electron-beam testing of VLSI circuits," *IEEE Transactions on Electron Devices*, vol. ED-26, pp. 549–559, April 1979.

[17] E. Wolfgang, "Electron beam testing," in *Handbook of Advanced Semiconductor Technology and Computer Systems*, (New York), pp. 148–180, 1988.

[18] Material Development Corporation, *Information Brochure*, 1988.

[19] I. T. Report, *Introduction to Microanalytical Probing*. Wentworth Laboratories, California, 1988.

[20] I. T. Report, *Probe Cards*. Wentworth Laboratories, California, 1988.

[21] I. T. Report, *Probe Card Application Note*. Wentworth Laboratories, California, 1988.

[22] R. L. Waters and J. K. Logan, "Methods of trace cutting for diagnostic probing of semiconductor devices," in *Proc. of the International Symposium for testing and Failure Analysis*, (Los Angeles, California), June 1984.

[23] M. M. Trivedi and A. Rosenfeld, "On making computers "see"," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, pp. 1333–1335, November 1989.

[24] C. Beddoe-Stephens, *Personal Communication*. Wentworth Laboratories Inc., 1983.

[25] Hewlett Packard Application Note, *DC Parametric Analysis of Semiconductor Devices*, March 1982.

[26] M. D. Levine, *Vision in Man and Machine*. New York: McGraw-Hill Book Company, 1985.

[27] D. Stewart, *Transfer of Data between HP9000 and SUN 3/60*. Concordia University, August 1987. Summer project report.

[28] D. Stewart, *An Image Processing Package*. Concordia University, August 1988. Summer project report.

[29] P. V. C. Hough, *Method and Means for Recognizing Complex Patterns*. No. 3 069 654, U. S. patent, 1962.

[30] R. C. Gonzalez and P. Wintz, *Digital Image Processing*. Reading, Massachusetts, USA: Addison-Wesley Publishing Company, 1982.

[31] D. Marr, *Vision: A Computational Investigation into the Human Representation and Processing of Visual Information.* New York: W. H. Freeman and Company, 1982.

[32] D. E. Dudgeon and R. M. Meresereau, *Multidimensional Digital Signal Processing.* Englewood Cliffs, New Jersey: Prentice-Hall Inc., 1985.

[33] B. K. P. Horn, *Robot Vision.* Cambridge, Massachusetts, USA: The MIT Press, 1986.

[34] V. S. Nalwa and T. O. Binford, "On detecting edges," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. PAMI-8, pp. 699–714, November 1986.

[35] D. H. Ballard and C. M. Brown, *Computer Vision.* Englewood Cliffs, New Jersey, USA: Prentice-hall, Inc., 1982.

[36] R. Krishnapuram and D. Casasent, "Hough space transformations for discrimination and distortion estimation," *Computer Vision, Graphics and Image Processing*, vol. 38, pp. 299–316, 1987.

[37] A. Rosenfeld and A. C. Kak, *Digital Picture Processing.* Academic Press, 1981.

[38] S. B. Marapane and M. M. Trivedi, "Region-based stereo analysis for robotic applications," *IEEE Transactions on Systems, Man, and Cybernetics*, vol. 19, pp. 1447–1464, November/December 1989.

[39] T. Hwang, J. J. Clark, and A. L. Yuille, "A depth recovery algorithm using defocus information," in *IEEE Computer Society International Conference on Computer Vision and Pattern Recognition*, (San Diego, California), June 1989.

[40] A. P. Pentland, "A new sense for depth of field," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 9, pp. 523–531, July 1987.

[41] A. P. Pentland, T. Darrel, M. Turk, and W. Huang, "A simple real-time range camera," in *IEEE Computer Society International Conference on Computer Vision and Pattern Recognition*, (San Diego, California), pp. 256–261, June 1989.

[42] M. Subbarao, *Efficient depth recovery through inverse optics*. New York: Acadamic Press, 1989.

[43] M. Subbarao, "Parallel depth recovery by changing camera parameters," in *IEEE Computer Society International Conference on Computer Vision*, (Florida), pp. 149–155, December 1988.

[44] M. Subbarao and G. Natarajan, "Depth recovery from blurred edges," in *IEEE Computer Society International Conference on Computer Vision and Pattern Recognition*, (Ann Arbor, Michigan), pp. 498–503, June 1988.

[45] R. V. Dantu, N. J. Dimopoulos, R. V. Patel, and A. J. Al-Khalili, "Micromanipulator vision for wafer probing," *IEEE Transactions on Semiconductor Manufacturing*, vol. 2, pp. 114–117, August 1989.

[46] G. Hausler and E. Korner, "Simple focusing criterion," *Applied Optics*, vol. 23, August 1984.

[47] P. Grossmann, "Depth from focus," *Pattern Recognition Letters*, vol. 5, pp. 63–69, January 1987.

[48] R. V. Dantu, N. J. Dimopoulos, R. V. Patel, and A. J. Al-Khalili, "Depth perception using blurring and its application to VLSI wafer probing," *International Journal of Machine Vision and Applications*, 1990. Accepted for publication.

[49] L. J. Siegel, H. J. Siegel, and P. H. Swain, "Performance measures for evaluating algorithms for SIMD machines," *IEEE Transactions on Software Engineering*, July 1982.

[50] V. Cantoni and S. Levialdi, "Multiprocessor computing for images," *Proceedings of IEEE*, vol. 76, pp. 959–969, August 1988.

[51] M. Maresca, M. A. Lavin, and H. Li, "Parallel architectures for vision," *Proceedings of IEEE*, vol. 76, pp. 970–981, August 1988.

[52] Q. F. Stout, "Mapping vision algorithms to parallel architectures," *Proceedings of IEEE*, vol. 76, pp. 982–995, August 1988.

[53] L. Uhr, *Parallel Computer Vision*. Orlando, Florida: Academic Press Inc., First ed., 1987.

[54] G. Conte and D. D. Corso, *Multi-Microprocessor Systems for Real-time Applications*. Dordrecht, Holland: D. Reidel Publishing Company, 1985.

[55] N. J. Dimopoulos, "On the structure of the Homogeneous Multiprocessor," *IEEE Transactions on Computers*, vol. C-34, pp. 141–151, February 1985.

[56] N. J. Dimopoulos, K. F. Li, E. C. W. Wong, R. V. Dantu, and J. W. Atwood, "The Homogeneous Multiprocessor system: An overview," *Proceedings of 1987 International Conference on Parallel Processing*, pp. 592–599, August 1987.

[57] N. J. Dimopoulos, K. F. Li, E. C. Wong, R. V. Dantu, and J. W. Atwood, "The Homogeneous Multiprocessor system: A status report," *Journal of Computer Systems Science and Engineering*, vol. 4, pp. 227–239, October 1989.

[58] K. F. Li, *Simulation Study and an Operating System Design of the Homogeneous Multiprocessor*. PhD thesis, Concordia University, Montreal, Canada, 1987.

[59] R. Cypher and L. C. Sanz, "SIMD architectures and algorithms for image processing and computer vision," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 37, pp. 2158–2174, December 1989.

[60] M. J. B. Duff, "CLIP4: A large scale integrated circuit array parallel processor," in *Proceedings of IEEE International Joint Conference on Pattern Recognition*,

pp. 728–733, November 1976.

[61] S. L. Tanimoto, *Multiresolution Image Processing and Analysis*, pp. 136–145. New York: Springer-Verlag, 1986.

[62] D. V. Ramanamurthy, N. J. Dimopoulos, K. F. Li, R. V. Patel, and A. J. Al-Khalili, "Parallel low level vision algorithms on the Homogeneous Multiprocessor," in *Proceedings of International Conference on Computer Vision and Pattern Recognition*, (Florida), pp. 421–423, June 1986.

[63] P. M. Kogge and H. S. Stone, "A parallel algorithm for the efficient solution of a general class of recurrence equations," *IEEE Transactions on Computers*, vol. C-22, pp. 786–793, 1973.

[64] H. J. Siegel, "PASM: A partitionable SIMD/MIMD system for image processing and pattern recognition," *IEEE Transactions on Computers*, vol. C-30, pp. 934–947, 1981.

[65] M. Mansuripur, "Distribution of light at and near the focus of high-numerical aperture," *Journal of Optical Society of America*, vol. 3, December 1986.

[66] T. Wilson and C. Sheppard, *Theory and Practice of Scanning Optical Microscopy*. London: Acadamic Press, 1984.

Appendix A
Program for HP 9000 for Capturing the Image using a DMA Controller

```
1 !*** IMAGE GRABBER AND DISPLAY OF GREY LEVEL IMAGE ON
2 ! 16-COLOR HP TERMINAL ***
6 ! AUGUST 18, 1986
7 ! Version 0.80
9 !
10 ! This program will gather a digitized 64-gray-level image from a
11 ! camera through the GPIO bus. The number of frames are
12 ! limited by the size of the buffer which can vary
13 ! depending on available memory. Each frame will be displayed, one
14 ! at a time. The user has to press <RETURN> between each frame
15 ! (allowing enough time for the computer to process the frame)
16 ! for the next frame to be displayed. The process continues until
17 ! either no more data is found (an error is reported), or the user
18 ! interrupts the program. This program drives the circuit diagram
19 ! described in Appendix C. Note that
20 ! this program will display a grey level image
20A ! on the HP color Terminal.
21 ! The terminal is only capable of displaying sixteen possible
22 ! levels, thus no quantization is needed.
24 !
25 OPTION BASE 0
26 INTEGER Nx,Ny,N,I,J,K,Irow,Icol,Jcol,Jj,Ni,Nj
```

```
27 !The control line CR1 (Appendix C) will set the second D flip-flop to zero
28 ! to inhibit the pulses going into the GPIO bus, then
29 ! wait for one second and set CR0 to the HI state
30 !in order to keep ready the input of the monostable.
32 CONTROL 12,2;2
33 WAIT 1
34 CONTROL 12,2;0
35 WAIT 1
36 Nx=255
37 Ny=255
38 ALLOCATE INTEGER Gray(Nx,Ny),Gr(Nx,Ny),Gr2(Nx,Ny),Hist(64)
39 ! GET DATA
40 Gpio=12 !select code of GPIO
41 !set cr0 to 0 and bring back to 1 to start the
42 !transfer event. Subsequently the 1 to 0 transition
43 !of the input to the monostable will cause a 5 second
44 !delay. The 0 to 1 transition of the monostable
45 !output will make the Q output of the flip-flop which will start the DMA
transfer.
47 CONTROL 12,2;1
48 WAIT 1
49 CONTROL 12,2;0
50 ASSIGN @Buff TO BUFFER [256000] !Create a buffer as large as
51 ! memory permits
52 ASSIGN @Camera TO Gpio
53 !
```

```
54 !Transfer one frame from the camera which consists of 256 Pixels
55 !per line (COUNT 256) and 256 lines per frame (RECORDS 256).
56 !The END should be redundant and is signaled by the cameras
57 !ROW line. This should only occur at the same time that the
58 !256th Pixel on a row is sent.
59 !
60 DISP "WAITING TO CONNECT"
61 TRANSFER @Camera TO @Buff
62 DISP "TRANSFER STARTED"
63 WAIT FOR EOT @Camera
64 BEEP
66 DISP "TRANSFER FINISHED, LOOKING FOR START OF FRAME"
67 !
68 ! Set graphics terminal
69 !
70 GINIT
71 PLOTTER IS CRT,"INTERNAL";COLOR MAP
72 GCLEAR
73 FOR I=0 TO 15 ! Set the sixteen displayable gray levels
74 SET PEN I INTENSITY I/15.0,I/15.0,I/15.0
75 NEXT I
76 !
77 !
78 !
79 ALLOCATE Image$(256)[256]
   !Store 256 lines of 256 pixels/line
```

```
80 ALLOCATE Line1$[256],Ch$[1]
81 INTEGER X,Xmin,Xmax,P
82 Xmin=255
83 Xmax=0
98 !
99 !
100 ! *** STORE FROM BUFFER INTO AN ARRAY ***
114 ENTER @Buff USING "%,256A";Image$(*)
    !Copy buffer into an array.
117 !
118 DISP "CONVERTING DATA"
119 N1=0
120 FOR I=0 TO Nx
121 Ni=255-I
123 FOR J=0 TO Ny
131 N=NUM(Image$(I)[J+1;1])
132 Gr2(I,J)=N
133 !N=N-64 !also remove the offset of 64 which was
134 !created by the frame line going high
137 Gray(Ni,J)=INT((N-64)/4) ! scale value to range 0..15
138 NEXT J
139 NEXT I
145 DISP "CREATING IMAGE, ";N1;" DATA BYTES LOST"
146 !
147 ! The following lines will create a new matrix, Gr(*), which is in
148 ! a form that the GLOAD command can use directly.
```

```
    ! The form is described
149 ! in the BASIC 4.0 Language Reference Manual,
    ! Page 154, 'storage format'
150 ! for the HP-9000 model 236 color computer.
    ! This description is also valid for
151 ! the HP-9000, model 300 color terminal
    ! except that the display is 1024 x 400
152 ! (Monitor model 35741A). Because of the difference in horizontal and
153 ! vertical resolution, there will be two horizontal pixels and one vertical
154 ! pixel used to display each pixel of the image.
155 FOR Irow=0 TO Nx
156 Jcol=0
157 FOR Icol=0 TO Ny STEP 4
158 ! See Page 154 of Language Reference Manual for description of
159 ! P1 to P8
160 P1=Gray(Irow,Icol) !P1 and P2 have the same value = P1
161 P3=Gray(Irow,Icol+1) !P3 and P4 have the same value = P3
162 P5=SHIFT(Gray(Irow,Icol+2),-4) ! P5 and P6 have same value = P5
163 P7=SHIFT(Gray(Irow,Icol+3),-4) ! P7 and P8 have same value = P7
164 Gr(Nx-Irow,Jcol)=SHIFT(P5,-8)+SHIFT(P1,-8)+P5+P1
165 Gr(Nx-Irow,Jcol+1)=SHIFT(P7,-8)+SHIFT(P3,-8)+P7+P3
167 Jcol=Jcol+2
168 NEXT Icol
169 NEXT Irow
170 ! The image is now ready to be GLOADed
171 BEEP
```

```
172 !INPUT "PRESS <RETURN> TO DISPLAY IMAGE",Dummy
173 DISP " "
174 GLOAD Gr(*) !Display image
175 PEN 15
177 FOR I=0 TO 255
178 FOR J=0 TO 255
179 P=Gr2(I,J)-64
180 Hist(P)=Hist(P)+1
181 NEXT J
182 NEXT I
183 BEEP
184 !INPUT "PRESS <RETURN> TO HISTOGRAM DISPLAY",Dummy
185 !PLOTTER IS CRT,"INTERNAL"
186 !GRAPHICS ON
187 VIEWPORT 70,130,50,100
188 FRAME
189 WINDOW 1,64,0,10000
190 AXES 1,1,1,1,5,5,3
191 FOR P=1 TO 64 STEP 1
192 PLOT P,Hist(P)
193 NEXT P
194 !INPUT "PRESS <RETURN> TO STORE THE FILE",Dummy
195 DISP " "
196 DISP "STORING DATA ON FILE"
197 ASSIGN @File TO "DATAFILE2:MEMORY,0,0";FORMAT ON
198 OUTPUT @File;Gr2(*)
```

```
199 ASSIGN @File TO *
200 !GOTO 100 !Finished processing, go get next frame to display
201 BEEP
202 BEEP
203 BEEP
204 BEEP
206 END
```

Appendix B
Procedure for Uploading Images from HP 9000 to SUN 3/60

The following procedure describes the uploading process from the HP 9000 to SUN 3/60.

1. Since the data collected from the camera is in the RAM of HP 9000 we transfer the data to the hard disk. This is done using the following command:

   copy "datafile2:Memory,0" to "data1:HP9133,700"

2. Enter the HP 9000 set-up using the function keys and select the following options.

   Set EOF as 'LF EOT'

   Set EOR as 'CR'

   Choose SOURCE file name (same as the file name in the hard disk of the HP 9000)

   Set device control as DC1/DC3 protocol.

3. Enter SUN UNIX OS using the function keys

   Set the terminal to 'tandem' mode using **stty tandem** command.

   Set the terminal to 'cbreak' mode by using **stty cbreak** command.

   Set the terminal echo off by using **stty echo** command.

   Keep SUN ready for collection of data. This is done by **cat > file name** where 'file name' is the name of the destination file.

4. Enter the HP 9000 set-up again and the choose function key 'FILES'. Start the uploading by pressing the function key 'UPLOAD'. When the transfer is finished the terminal will go back to the set-up menu.

5. Enter SUN UNIX OS and stop the uploading process by **^Z.**

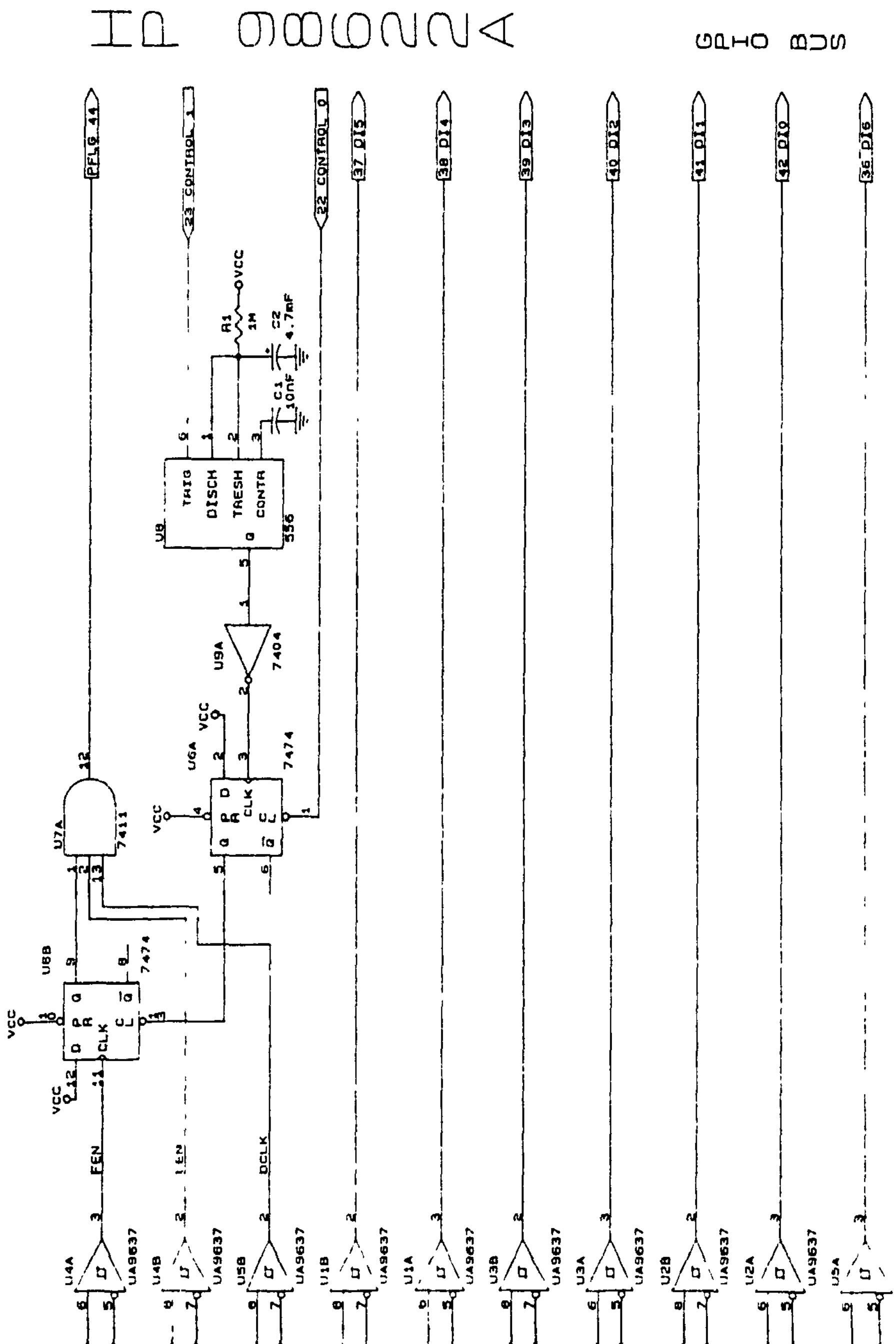
   Set the terminal echo by using **stty echo** command.

Remove the 'cbreak' mode by using **stty —cbreak** command.

Resume the uploading process by **fg** %+.

Physically close the file using **^D**. The uploading process is now complete.

Appendix C
Circuit Diagram for Camera Interface to HP98620 DMA Controller



Appendix D
Algorithm for Alignment of Images

Inputs: A window of size *window_size* is selected at the address of *search_origin*. The parameters *search_length, search_breadth* denote the length and breadth of the search. *Threshold* is the limit for the change in gray values due to misalignment. This value is set above the signal to noise ratio of the image. *Window_gray_value* is the total gray value of the window. This depends on the material of the wafer area under consideration. Hence this parameter is set experimentally. I_0 indicates the image at the instant 0 and I_i indicates the image under consideration.

Outputs: The parameters *row_shift* and *column_shift* are calculated from this algorithm and indicate the number of rows and columns shifted due to the misalignment.

```
1. Set the parameters of search_length, search_breadth, search_origin, threshold.
2. Set the window gray_value, window_size.
3. For i from -search_length to search_length
      Begin
      For j from -search_breadth to search_breadth
         Begin
         Set sum_gray_difference=0
         For all rows and columns in the window from search_origin to
         search_origin+window_size
            Begin
            Calculate the difference= $I_i(row+i,column+j)-I_0(row,column)$
            If difference is greater than threshold then
            sum_gray_difference=sum_gray_difference+difference
            End
         If sum_gray_difference is greater than the window_gray_value
            Begin
            Set window_gray_value=sum_gray_difference
            row_shift = i
            column_shift = j
            End
         End
      End
```

Appendix E
Algorithm for Elimination of Background Traces

Inputs: The size *neighborhood* is set according to the thickness of the background traces to be eliminated. I_i is the image consisting of background traces. The parameter *threshold* is used to test whether the pixel under consideration belongs to the probe or to the traces of the background. If its neighborhood is empty or nearly empty, then it is considered to be belonging to the traces of the background.

Cutputs: All pixels corresponding to the traces of the background are set to 0 in the image I_i.

```
Set the value of neighborhood=15
Set the value of threshold=0.4
For all pixels of the image $I_i(p,q)$
    Begin
        Set the value of the surround to 0
        For ps from p-neighborhood to p+neighborhood
            Begin
            For qs from q-neighborhood to q-neighborhood
                Begin
                If ps is equal to (p-neighborhood) OR (p+neighborhood)
                (q-neighborhood) OR (q+neighborhood)
                then Add $I_i(ps,qs)$ to surround
                End
            End
        Set average_surround equal to surround/((4)*(2 * neighborhood))
        If average_surround is less than threshold * $I_i(p,q)$
        then $I_i(p,q)$ equal to 0
    End
```

Appendix F
Integrals of *Q(x)* and *Q²(x)*

Integral of a Q Function

The integral of the Q function is evaluated as follows.

$$\begin{aligned}\int_{-T-t}^{t-T} Q\left(\frac{x+T}{\sigma}\right) &= \int_{-t}^{t} Q\left(\frac{v}{\sigma}\right) dv \\ &= \int_{-t}^{0} Q\left(\frac{v}{\sigma}\right) dv + \int_{0}^{t} Q\left(\frac{v}{\sigma}\right) dv \\ &= t \quad (since\ Q(-x) = 1 - Q(x))\end{aligned} \tag{1}$$

Similarly we can prove that

$$\int_{T-t}^{t+T} Q\left(\frac{x-T}{\sigma}\right) = t \tag{2}$$

Bounded Values of Integrals of *Q(x)* and *Q²(x)*

Lemma 1. For all x>0, the values of integrals $\int_0^x Q(p)\,dp$ and $\int_0^x Q^2(p)\,dp$ are bounded.

Proof: The upper bound of *Q(x)* can be written as $\frac{1}{2}e^{\frac{-x^2}{2}}$. Hence, the upper bound of the above integrals can be written as $\int_0^x \frac{1}{2}e^{\frac{-p^2}{2}}dp$ and $\int_0^x \frac{1}{2}e^{-p^2}dp$ respectively. But, we know that $\int_0^\infty e^{\frac{-p^2}{2}}dp = \frac{\sqrt{\pi}}{2}$. Therefore it follows that the integrals $\int_0^x Q(p)\,dp$ and $\int_0^x Q^2(p)\,dp$ are bounded.

The Integral of the Square of a Q Function

Lemma 2.

$$\int_{-T-t}^{t-T} Q^2\left(\frac{x+T}{\sigma}\right) dx = \int_{T-t}^{t+T} Q^2\left(\frac{x-T}{\sigma}\right) dx \tag{3}$$

$$= t - 2q\sigma$$

Proof:

$$\begin{aligned}\int_{-T-t}^{t-T} Q^2\left(\frac{x+T}{\sigma}\right) dx &= \int_{-t}^{t} Q^2\left(\frac{v}{\sigma}\right) dv \\ &= \int_{-t}^{0} Q^2\left(\frac{v}{\sigma}\right) dv + \int_{0}^{t} Q^2\left(\frac{v}{\sigma}\right) dv\end{aligned} \tag{4}$$

$$= \sigma \int_{\frac{-t}{\sigma}}^{0} Q^2(p)\, dp + \sigma \int_{0}^{\frac{t}{\sigma}} Q^2(p)\, dp \tag{5}$$

$$= \sigma \int_{0}^{\frac{t}{\sigma}} [1 - Q(p)]^2\, dp + \sigma \int_{0}^{\frac{t}{\sigma}} Q^2(p)\, dp \tag{6}$$

We can further reduce the above expression to

$$\int_{-T-t}^{t-T} Q\left(\frac{x+T}{\sigma}\right) dx = t - 2\sigma \int_{0}^{\frac{t}{\sigma}} Q(p)\, dp + 2\sigma \int_{0}^{\frac{t}{\sigma}} Q^2(p)\, dp \tag{7}$$

Since $\int_0^{\frac{t}{\sigma}} Q(p)\, dp = q_1$ (a bounded value) and $\int_0^{\frac{t}{\sigma}} Q^2(p)\, dp = q_2$, (a bounded value) we can write

$$\int_{-T-t}^{t-T} Q^2\left(\frac{x+T}{\sigma}\right)dx = t - 2\sigma(q_1 - q_2) \tag{8}$$
$$= t - 2q\sigma$$

where $q=(q_1—q_2)$ (the value of q is calculated by numerically evaluating the integrals for q_1 approximated to 0.42, and q_2 approximated to 0.16). Similarly, we can also show that

$$\int_{T-t}^{t+T} Q^2\left(\frac{x-T}{\sigma}\right)dx = t - 2\sigma(q_1 - q_2) \tag{9}$$
$$= t - 2q\sigma$$

Appendix G
Algorithm for Detection of Touch

Inputs: The pixel values corresponding to the background are inputted using the variable *background*(the value K in Figure 5.1). When the probe approaches this background, the image contains the background and the probe. The pixel values corresponding to this image is inputted using the variable *with_probe*(the function $h_1(x)$). If the difference of these two values is greater than the *threshold* then the pixel in the image is considered as contributed by the probe. The value of *threshold* is chosen to be more than the signal to noise ratio(SNR) of the image.

Outputs: The pixel values contributed by the probe(*probe* and $h_2(x)$ in Figure 5.1) are collected from the image containing background and the probe. This algorithm calculates the mean (Equation 5.3.2) and variance (Equation 5.3.6) values of the pixel values of the probe.

```
Begin
Set the value of threshold=5% of the maximum grey value in the image
   Begin
      read(background)
      read(with_probe)
      For all pixels
         Begin
         difference=background—with_probe
         If (difference>=threshold)
            probe= with_probe
         else probe=0
         End
   End
Calculate_mean(probe)
Calculate_variance(probe)
End
```