

BLDSC :- DX 171774

LOUGHBOROUGH
UNIVERSITY OF TECHNOLOGY
LIBRARY

AUTHOR/FILING TITLE

ZHANG, J

ACCESSION/COPY NO.

036000319

VOL. NO.

CLASS MARK

~~- 1 MAR 1993~~

LOAN COPY

~~- 1 JUL 1994~~

30 JUL 1995

~~11 NOV 1993~~

28 JUN 1996

~~- 1 JUL 1994~~

25 JUN 1999

07 OCT 1994

26 JUN 1998

036000319 2



BADMINTON PRESS
18 THE HALFCROFT
SYSTON
LEICESTER LE17 8LD
ENGLAND
TEL: 0533 602917
FAX: 0533 696636

**ON FLEXIBLY INTEGRATING
MACHINE VISION INSPECTION SYSTEMS
IN PCB MANUFACTURE**

by

JINGBING ZHANG

A Doctoral Thesis
Submitted in partial fulfillment of the requirements
for the award of
Doctor of Philosophy
of the Loughborough University of Technology

Loughborough University
Department of Manufacturing Engineering

March 1992

© by Jingbing Zhang

Loughborough University	
of Technology Library	
Date	Aug 92
Acc No	036000319

w 9921256

TO

My Parents and My Wife

ACKNOWLEDGEMENTS

The author wishes to thank:

Professor R. H. Weston for his kind supervision, encouragement and support.

Mr. J. M. Edwards, Mr. I. S. Murgatroyd, Mr. I. A. Coutts, Mr. J. D. Gascoigne, Mr. P. Clements, Mr. G. P. Charles, Mr. D. Walters for their interest, technical support and friendship.

Mr. X. D. Chen, Mr. J. Wang, Mrs. Y. Gu, Mr. X. T. Yan, Mrs. Y. H. Li, Mr. Y. Zhang, Mr. X. Z. Ren, Mr. J. H. Zhao, Mrs. J. Zhang, Mr. A. S. Goh, Mr. C. B. Wong, Mrs. H. Jiao, Miss F. Zheng, Miss G. H. Li, Dr. J. S. Pu for their friendship.

Mrs. M. E. Carden and all the staff from the Department for their kindness and administrative work.

Finally, my motherland, the People's Republic of China, and the British Council for providing financial support.

SYNOPSIS

The objective of this research is to advance computer vision techniques and their applications in the electronics manufacturing industry. The research has been carried out with specific reference to the design of automatic optical inspection (AOI) systems and their role in the manufacture of printed circuit boards (PCBs).

To achieve this objective, application areas of AOI systems in PCB manufacture have been examined. As a result, a requirement for enhanced performance characteristics has been identified and novel approaches and image processing algorithms have been evolved which can be used within next generation of AOI systems. The approaches are based on gaining an understanding of ways in which manufacturing information can be used to support AOI operations. Through providing information support, an AOI system has access to product models and associated information which can be used to enhance the execution of visual inspection tasks. Manufacturing systems integration, or more accurately controlled access to electronic information, is the key to the approaches. Also in the thesis methods are proposed to achieve the flexible integration of AOI systems (and computer vision systems in general) within their host PCB manufacturing environment. Furthermore, potential applications of information supported AOI systems at various stages of PCB manufacturing have been studied.

It is envisaged that more efficient and cost-effective applications of AOI can be attained through adopting the flexible integration methods proposed, since AOI-generated information can now be accessed and utilized by other processes.

CONTENTS

<i>Declaration</i>	i
<i>Acknowledgements</i>	ii
<i>Synopsis</i>	iii
<i>Contents</i>	iv
<u>Chapter 1</u>	<u>Introduction</u>
	1
<u>Chapter 2</u>	<u>Literature Survey</u>
	3
2.1	Introduction
	3
2.2	The Role of Computers in Manufacturing Industries
	3
2.2.1	Uses of Computers in Manufacturing Industries
	3
2.2.2	The Need for Integrating Manufacturing Systems
	5
2.3	Modelling of Manufacturing Systems
	6
2.4	Computer Integrated Manufacture
	9
2.5	Methods and Tools for Manufacturing Systems Integration
	12
2.5.1	Networking and Communication Standards
	12
A.	ISO/OSI Standards and the ISO/OSI Reference Model
	13
B.	IEEE 802.X Standards
	14
C.	MAP/TOP
	14
D.	SNA
	17
2.5.2	Manufacturing Information Handling
	19
A.	Information Administration Systems
	19
B.	Standards for Information Representation
	21
	IGES
	22
	PDES and STEP
	22
	EDIF
	24
2.5.3	Manufacturing Systems Integration Architectures
	24
A.	Three architectures
	24
B.	ESPRIT CIM-OSA
	28
2.6	Computer Vision Technology and Its Applications in Manufacturing
	28
2.6.1	The Building Elements of a Computer Vision System
	31
2.6.2	2-D Image Processing and Analysis Techniques
	32
A.	Preprocessing
	35
B.	Segmentation
	37

	C. Feature Extraction	41
	D. Recognition	41
2.6.3	3-D Computer Vision Techniques	43
2.6.4	Industrial Applications of Computer Vision	43
	A. Inspection	44
	B. Guidance	44
2.7	Applications of Computer Vision in Electronics Manufacturing Industry	45
2.7.1	Automatic Optical Inspection	46
2.7.2	AOI Techniques and Their Applications in PCB Product Inspection	46
	A. Reference Based Techniques	48
	B. Non-Reference Based Techniques	49
2.8	Summary	50
<u>Chapter 3</u>	<u>The Requirement for Integrated Machine Vision Application in PCB Manufacture</u>	53
3.1	Introduction	53
3.2	A Conceptual Model of PCB Manufacturing Industry	55
3.2.1	Design	55
3.2.2	Manufacture	57
3.2.3	Monitoring	59
3.3	Automatic Optical Inspection and Its Role in PCB Manufacturing	60
3.3.1	A Comparison between AOI and ATE	62
3.4	Present Generation AOI	63
3.4.1	Limitations of Present Generation AOI systems	66
3.5	CAD Reference for AOI	67
3.5.1	Benefits of CAD Data Reference	69
3.5.2	Problems Associated with CAD Reference	70
3.6	Information Support for AOI	71
3.6.1	Specifications of a New Generation AOI system	72
3.7	Summary	74
<u>Chapter 4</u>	<u>The Integration of Machine Vision and Robotic Systems</u>	76
4.1	Introduction	76
4.2	Description of the Facilities Used	77
4.2.1	The Matrox Machine Vision System (MVS)	77

4.2.2	The AdeptOne Robot Manipulator System	80
4.3	Vision-Robot Communication	81
4.3.1	Overview	81
4.3.2	Specific Constraints	82
4.3.3	Description of the Approach Adopted	83
4.3.4	The Information Requirements of the Two Systems	85
A.	PCB Inspection	85
B.	Vision Guidance for Robot	86
4.3.5	Information Representation Format (IRF)	88
4.4	An Example Application of the Integrated System	91
4.5	Discussions	94
4.6	Summary	97
Chapter 5	<u>Using CAD Information to Support AOI Operations</u>	98
5.1	Introduction	98
5.2	Systems/Tools Utilized	99
5.2.1	The P-CAD System	101
5.2.2	The Matrox Machine Vision System	103
5.2.3	SUN Workstation and the LEX/YACC Tools	104
5.3	General Issues of CAD Information Support for AOI Applications	106
5.4	Designing of a Software Information Generator Using LEX/YACC	110
5.5	The Utilization of the CAD Information	111
5.6	Discussions	116
5.7	Summary	118
Chapter 6	<u>A Product Model Based Approach to Integrating Vision Systems in the CIM of PCB</u>	120
6.1	Introduction	120
6.2	Notion Relating to a Product Model Based Approach	121
6.2.1	Use of Terminology	121
6.2.2	A Proposed Product Model Based Approach	124
6.3	Some Issues Relating to Product Model	128
6.3.1	Two Types of PCB Product Models	129
A)	The Conceptual Product Model	129
B)	The Physical Product Model	131

6.3.2	Two Views of the Product Models	132
A)	Global Conceptual and Physical Product Models	132
B)	Local Conceptual and Physical Product Models	133
6.3.3	Theoretical Model vs Calibrated Model	134
6.4	Pre- and Post-processors	136
6.4.1	Pre-processors	137
6.4.2	Post-processors	137
6.5	Summary	138
Chapter 7	<u>Suggested Architectural Design of an AOI System</u>	139
7.1	Introduction 1	139
7.2	Characterizing Typical AOI Applications in PCB Manufacture	141
7.2.1	Further Examination of AOI Applications and Their Generic Characteristics	144
A.	Panel Inspection (PI)	144
B.	Solder Paste Application Inspection (SPAI)	145
C.	Component Placement Inspection (CPI)	147
D.	Solder Joint Inspection (SJI)	149
7.3	AOI Task Decomposition	152
7.3.1	Decomposing General AOI Tasks	153
7.4	Hierarchical Design of AOI Algorithms	157
7.4.1	The proposed Hierarchical Reference Architecture	157
A.	Level_0: The Primitive Routines (PRs)	160
B.	Level_1: Element Attribute Extraction (EAE) Algorithms	162
C.	Level_2: Transitional Feature Generation (TFG) Algorithms	164
D.	Level_3: Global Board Analysis (GBA) Algorithms	167
E.	Level_4: Application Oriented Algorithms (AOAs)	169
7.5	The Data Flow Hierarchy	171
A.	The Raw Image Data (RID)	171
B.	The Elementary Local Attributes (ELAs) and the Transitional Region Features (TRFs)	172
C.	Global Board Features (GBFs) and Application-Specific Reports (ASRs)	173
7.6	Summary	175

<u>Chapter 8</u>	<u>Description Of A Prototype Information Supported Machine Vision System</u>	177
8.1	Introduction	177
8.2	Description of the Software Platform SPADAT	177
8.3	Towards Achieving Flexible Information Support for AOI	181
8.3.1	Implementing a Global Physical Product Model	183
8.3.2	Software Development	185
A.	Software to Extract Global Information from a CAD Database	185
B.	Software to Provide Information Support for Local Applications	186
8.4	Implementation of the AOI Reference Architecture	187
8.4.1.	The Primitive Routines	189
8.4.2.	The Element Attribute Extraction (EAE) and Transitional Feature Generation (TFG) Algorithms	194
8.4.3	The Global Board Analysis (GBA) Algorithms and Application Oriented Algorithms (AOAs)	202
8.5	The Vision-Robot Integrated System and its Calibration	206
8.5.1	The Communication Link for the Vision-Robot System	206
8.5.2	Calibration of the Vision-Robot System	209
8.6	Experimental Results: A Case Study	211
8.6.1	Introduction	211
8.6.2	System Configuration	211
8.6.3	Inspection/Location of Fiducial Marks	213
8.6.4	Inspection of Pads	215
8.6.5	Problems of Noise in the Inspection Process	216
8.6.6	The Role of the Proposed AOI Reference Architecture in This Case Study	217
8.7	Summary	218
<u>Chapter 9</u>	<u>Conclusions and Recommendations</u>	229
9.1	Conclusions and Contributions to Knowledge	229
9.2	Contribution to Knowledge	230
9.2.1	The Product Model Based Approach	230
9.2.2	A Proposed AOI Reference Architecture and Its Partial Implementation	236
9.3	Recommendations for Future Work	240
<u>References</u>		242

<u>Appendix A</u>	<u>List of Routines Implemented</u>	262
<u>Appendix B.1</u>	<u>An Example of PDIF Files</u>	266
<u>Appendix B.2</u>	<u>YACC Code for Software Information Generators</u>	279

Chapter 1

Introduction

This thesis is concerned with advancing studies and applications of machine vision technology/systems in PCB (printed circuit board) manufacture. More specifically, it is concerned with evolving methods of achieving better AOI (automatic optical inspection) performance by means of providing information support through flexibly integrating the operations of AOI systems with those of other relevant product realisation processes/systems involved at various stages of PCB manufacture.

A prerequisite of research in this field requires a good understanding of background concepts and available knowledge in the areas of systems integration and computer vision technology. Thus chapter 2 reviews literature on computers applications in manufacture and more specifically computer vision techniques. Further detailed discussions on the potential role and application areas of machine vision systems in PCB product inspection is discussed in chapter 3. A growing range of applications of AOI systems, especially at critical processing stages in PCB manufacturing, have been identified which are important to the success and survival of modern PCB manufacturing organisations in markets where competition is fierce. Information supported AOI is considered by the author to be the right way forward.

Chapter 4 describes the author's study of interaction between a Matrox vision system and an AdeptOne robot manipulator system, and subsequent integrated operations of the two systems. The approach used to integrating these two systems is of a bespoke nature, i.e. customized communication protocols are deployed for the data and information interchange between the two systems. In addition, custom designed data structure and information representational format was specified which satisfies requirements of the two end systems.

Chapter 5 presents the author's work of integrating a CAD (computer aided design) system and a vision machine. The main issue addressed is the re-utilisation by an AOI system of CAD generated reference information during inspection of PCB products. Other issues such as information parsing and translating are considered and recommendations made in regard to the extraction of suitable information. Thus a methodology is proposed by the author for reusing CAD reference data to support AOI applications which may be an important step forward towards achieving full information supported AOI application in the domain of PCB manufacture.

Chapter 6 describes a proposal of a product model based approach to achieving flexible integration of AOI systems within the wider computer integrated manufacturing (CIM) environment. Here, product models are considered as a means of providing a comprehensive product definition and of representing manufacturing information which can be used to support many manufacturing operations; including product inspection using AOI systems.

Chapter 7 proposes an architectural framework which can be used to structure the design and implementation of software algorithms for building the next generation AOI systems that can make better use of information available from CAD/CAM (computer aided design/manufacture) systems. The design focus comes from aiming to address the question, "given the availability of CAD/CAM created and other product-realization-related information, how should AOI systems be structured to make the best use of manufacturing information which can be used to support its operations".

Finally, chapter 8 describes a prototype software implementation of the suggested AOI system reference architecture and the proposed product model based approach to system integration. This is followed by concluding remarks in chapter 9 which outlines the major conclusions drawn from this research, the contributions to knowledge made and recommendations for future work.

Chapter 2

Literature Survey

2.1 Introduction

The literature to be reviewed in this chapter is divided up into two major parts. The first part reviews literature on the broad issue of computer applications in manufacturing; including computer based tools and methods of achieving systems integration. This serves to set up the broad frame of reference wherein all computerized manufacturing devices including computer vision systems are hosted.

The second part of this chapter reviews background concepts and knowledge about computer vision including basic techniques for digital image processing. Emphasis is placed on automatic optical inspection (AOI), which represent one specific application area of computer vision technology.

2.2 The Role of Computers in Manufacturing Industries

2.2.1 Uses of Computers in Manufacturing Industries

Computers have now become invaluable tools which are increasingly used in the operations of many different types of manufacturing industries [Powers, Jr. 1987]. Examples of computer applications can be easily found; e.g. programmable logic controllers (PLCs) [Pessen and Hübl 1979], computerised numerical control (CNC) of machine tools [Koren 1983] [Seames 1990], computer-aided design

and manufacturing (CAD/CAM) [Groover and Zimmers Jr. 1984] [Besant and Lui 1986], robotics [Craig 1986] [Fu et al 1987], computer vision [Ballard and Brown 1982], computer-based process control (e.g. control of process variables such as temperature, pressure, flow, etc.) [Holland 1983], and so on.

Along with this process of computerization of manufacturing systems/processes, the nature of modern manufacturing industry is tending to evolve into a blend of data-driven operations [Kutcher 1983]. As new and more sophisticated computer-based technologies are being continuously introduced and employed in a greater range of manufacturing processes, and as more stringent customer demands are applied to marketable products, manufacturing operations are becoming increasingly more complex [Pao 1984]. Thus the data and information required in completing shop-floor manufacturing operations in a competitive manner (as well as management and decision-making) have increased substantially to the point that computers must be used to handle it [Powers, Jr. 1987] [Charif 1986]. Consequently, the use of computer-based equipment and computer software to manipulate huge quantities of information, stored in a digital form, becomes a common feature of contemporary manufacturing enterprises [Solberg and Heim 1989].

As in many other manufacturing industries, the computerisation of electronic manufacturing industry, in particular the PCB manufacturing industry, is, on the one hand, “pulled” by market demand and strict customer requirement for delivering high quality product on time and at low cost [Wearden 1990]. On the other hand, this process of computerization is also “pushed” by technological development in computers and other related domain technology and the increasing availability of computerized equipment to automate shop-floor manufacture [Riley 1988]. Such equipment can be classified into the following broader categories, viz:

- Computer aided engineering and design (CAE/CAD) systems [Du Feu 1988] [Riley 1988] [Hansohn 1990],
- Computer aided manufacturing (CAM) systems: including for

example NC drilling machines, automatic component insertion/ onsertion (ACI/ACO) and automated assembly machines [Du Feu 1988], automated test equipment (ATE) and automated inspection stations (e.g. an AOI system), computerised process control (e.g. soldering [Cox 1988] [Spitz 1988], etching and plating [Makstein 1988a], etc.), and

- Computer-aided process planning and control (PP&C), integrated tooling and engineering control (ITEC) and management information system (MIS) [Balius 1990].

2.2.2 The Need for Integrating Manufacturing Systems

The widespread application of computers and computer-based equipment in manufacturing industries has indeed promoted productivity of various industry sectors: in terms of more efficient manufacturing operations, shortened time-to-market of products, reduced unit product costs and in terms of improved product quality [Powers, Jr. 1987].

However, since the process of computerization has been a gradual process (mirroring but lagging behind developments in computer technology), many segmented “islands of automation” have been created [Llewelyn 1989] [Harhalakis et al 1991]. As a result of these step-wise enhancements, segments of product realisation have also advanced in a stand-alone manner, meaning that they only perform a well defined range of tasks of limited scope with little assistance of other devices [Weston et al 1988]. One way of viewing this is that they are not designed to interact or share information with other segments [Hansohn 1990]. As a result, achieving synergism (i.e. greater functionality from the whole than a summation of functionality of individual parts) has traditionally been very limited. In other words, the realisation of the synergy inherent among these sub-systems may be achieved only through a full integration of these “islands of automation” [Graves et al 1988].

As manufacturing industry takes on a new data-driven (programmable) nature, tasks of data manipulation and information handling have become a common feature [Weston et al 1988] of any computerised manufacturing system. Effective and efficient processing and utilisation of all relevant manufacturing information is becoming one of the prerequisites of many industrial sectors to maintain competitive edge or even to survive in modern world of industry [Kutcher and Gorin 1983].

In view of the limitations of “islands of automation” approach to computerized-manufacturing, researchers worldwide have strived to evolve suitable means of integrating relevant activities of various sectors of manufacturing, with a view to maximising the utilization of computerised manufacturing devices [Hemond 1986] and especially the information stored in and manipulated by these devices [PE Staff Report 1986] so as to improve efficiency of factory operations and to achieve more rapid response to new marketing demands [Weston 1991a].

2.3 Modelling of Manufacturing Systems

Manufacturing systems, which often comprise many components (such as people, work stations, robots, tools, conveyer, storage, AGVs (automatic guided vehicles), etc.) [Chow et al 1985], are inherently complex systems, particularly so as their size increases; therefore in order to achieve an appropriate level of integration of the operations (processes) involved in such complex systems, it is necessary in the first place to have a good understanding of the fundamental concepts, functions and information requirements of systems and their constituent parts [Hitomi 1990].

For these reasons the modelling of manufacturing operations has become an important area of interest and research, which is expected to promote a “uniform understanding” of functions, information requirements, system necessary resources, structural organization, operational behaviours, etc. of the manufacturing enterprise [Mertins and Sussenguth 1991]. Research activities in this area have led

to a number of proposed models of manufacturing which describe manufacturing operations from various viewpoints, providing to some level of abstraction a description of real world manufacturing systems. Some examples of these models are CAM*I [CAMI 1983] [Boylin 1990], ISO [ISO 1986] [Shorter 1990], CIM-OSA [Beechman 1989] [Kosanke 1991] [Panse 1990], CIM-BIOSYS [Leech et al 1991].

While it is common and reasonable that different viewpoints are adopted when modelling particular problems of interest, the existence of an array of non-conforming conceptual models does not naturally lead to well designed manufacturing systems from a global perspective. Certainly, it is not a straight forward matter to agree on the functionality of each hierarchical layer typically found within these models or indeed even the number of hierarchical layers that typically exist. However, it is necessary and important to have a conceptual model which is generic enough to be applied to any systems with little modification. Such a universally applicable generic model could greatly assist human understanding of manufacturing systems [Dooner 1989]. Furthermore, conceptual models can also be used to provide guidelines for the design and implementation of complex manufacturing systems [Jorysz and Vernadat 1990a]. The existence of such a generic model (which is likely to be of a hierarchical nature although heterarchical models of manufacturing systems have also been specified [Solberg and Heim 1989]) relies on the fact that generic requirements are common to all manufacturing areas [Anderson et al 1990], and that every manufacturing system exhibits much the same characteristics (though different in details). A recognition of generic similarities (and underlying commonality) in all manufacturing systems can be identified and more generic solutions be sorted out through classification and analysis of typical cases [Parunak and White 1987]. This process is illustrated in Figure 2.1.

However, it has not yet been demonstrated that such a model can be identified without either being so abstract that it is not very useful or too prescriptive so that it only represents reality in a very narrow application domain [Haren and Williams 1990] [Böhms and Tolman 1990]. Nonetheless, it is worth noting that

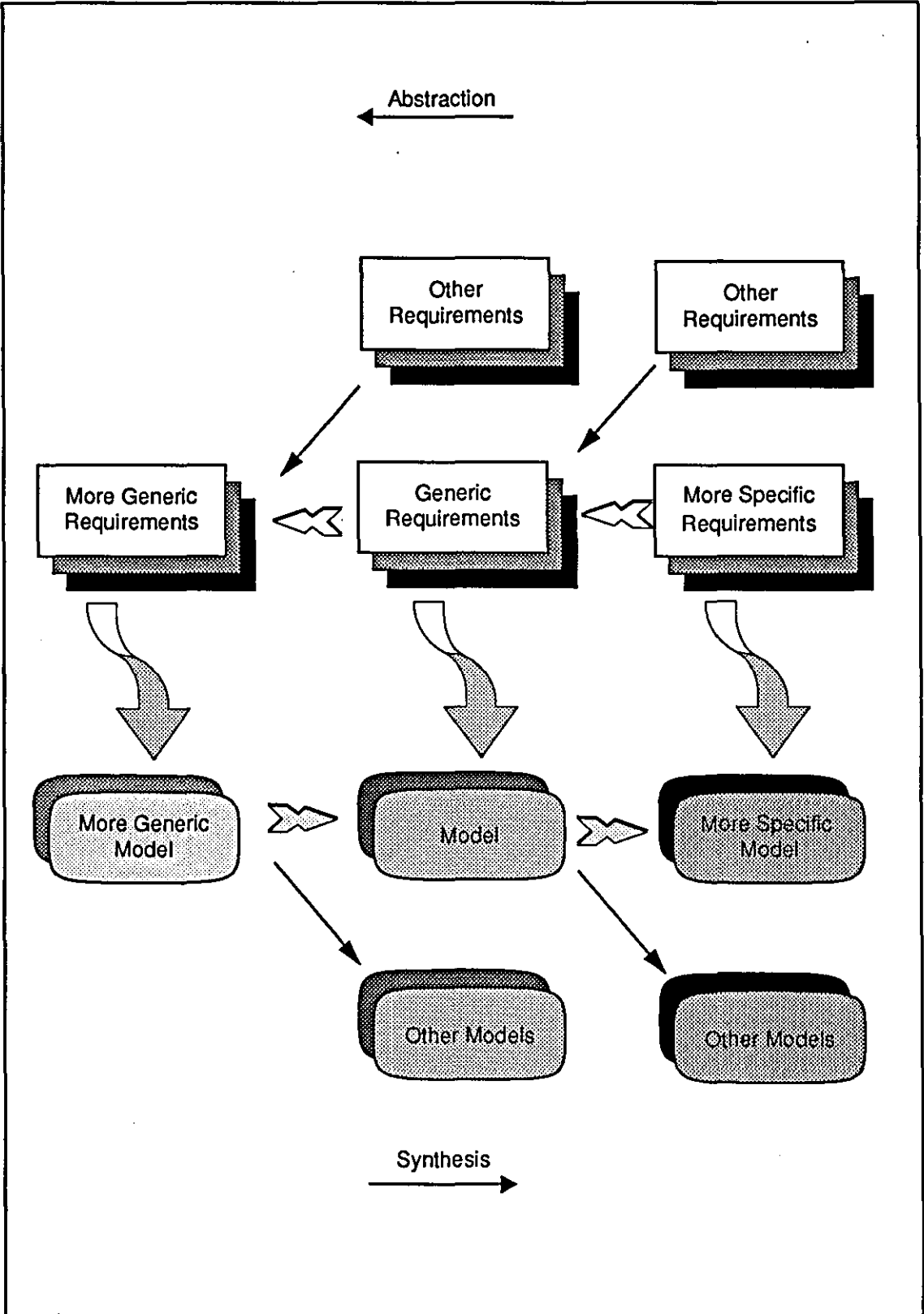


Figure 2.1 Abstraction and Synthesis [Parunak-White 1987]

among the various proposed models, the hierarchical factory reference model adopted by the international standards organisation (ISO) has gained wide international acceptance [Shorter 1990]. This model which serves to define the hierarchical operational levels within a manufacturing enterprise is illustrated in Figure 2.2.

2.4 Computer Integrated Manufacturing

Research activity in modelling manufacturing systems has advanced an understanding of fundamental concepts and requirements involved in manufacturing organizations. This enhanced understanding of manufacturing has led to a realization of the need and the importance of systems integration. Information is a valuable resource within manufacturing and the management and control of manufacturing information on an enterprise-wide basis has been considered central to the concept of computer integrated manufacturing (CIM) [Murray 1988] [Harhalakis et al 1991].

CIM aims at the complete integration of all activities and subsystems of a manufacturing organization; from the receipt of customer orders, through product specifications, product design and modification, product manufacturing and quality control (testing and inspection), to product delivery and after-sale services [Groover 1987]. See Figure 2.3. In reality, various turnkey (stand-alone) computerized subsystems, devices and tools are involved in this cycle of production, assisting with product design, development and manufacturing, as well as with organizational management. These devices and tools are building elements for CIM and are required to be interconnected or integrated in order to exploit CIM [Hemond 1986] [Foong and Hoang 1991]. Typically, such CIM elements would include: computer-aided design (CAD) systems, robots, material requirement planning (MRP), computer-aided manufacturing (CAM) stations such as NC (numerical control) drill, CNC (computer numerical control) machines, automatic optical inspection (AOI) systems, automated test equipment (ATE), etc. The trend is towards multi-vendor manufacturing, implying that for each type of devices there will be a group of suppliers; there-

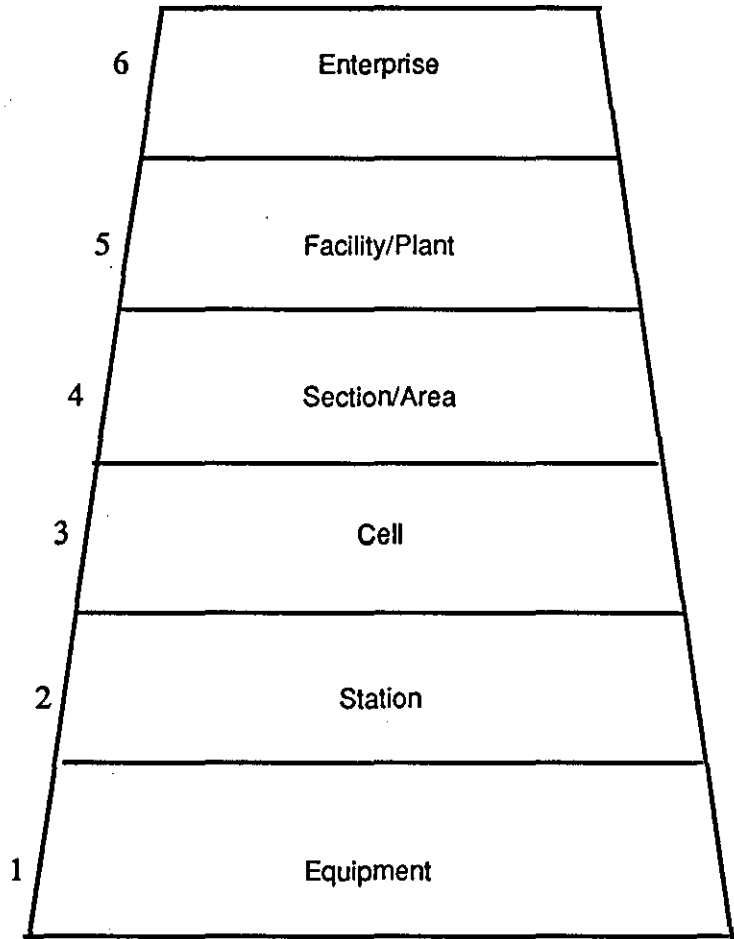


Figure 2.2 The ISO Factory Reference Model [Shorter 1990]

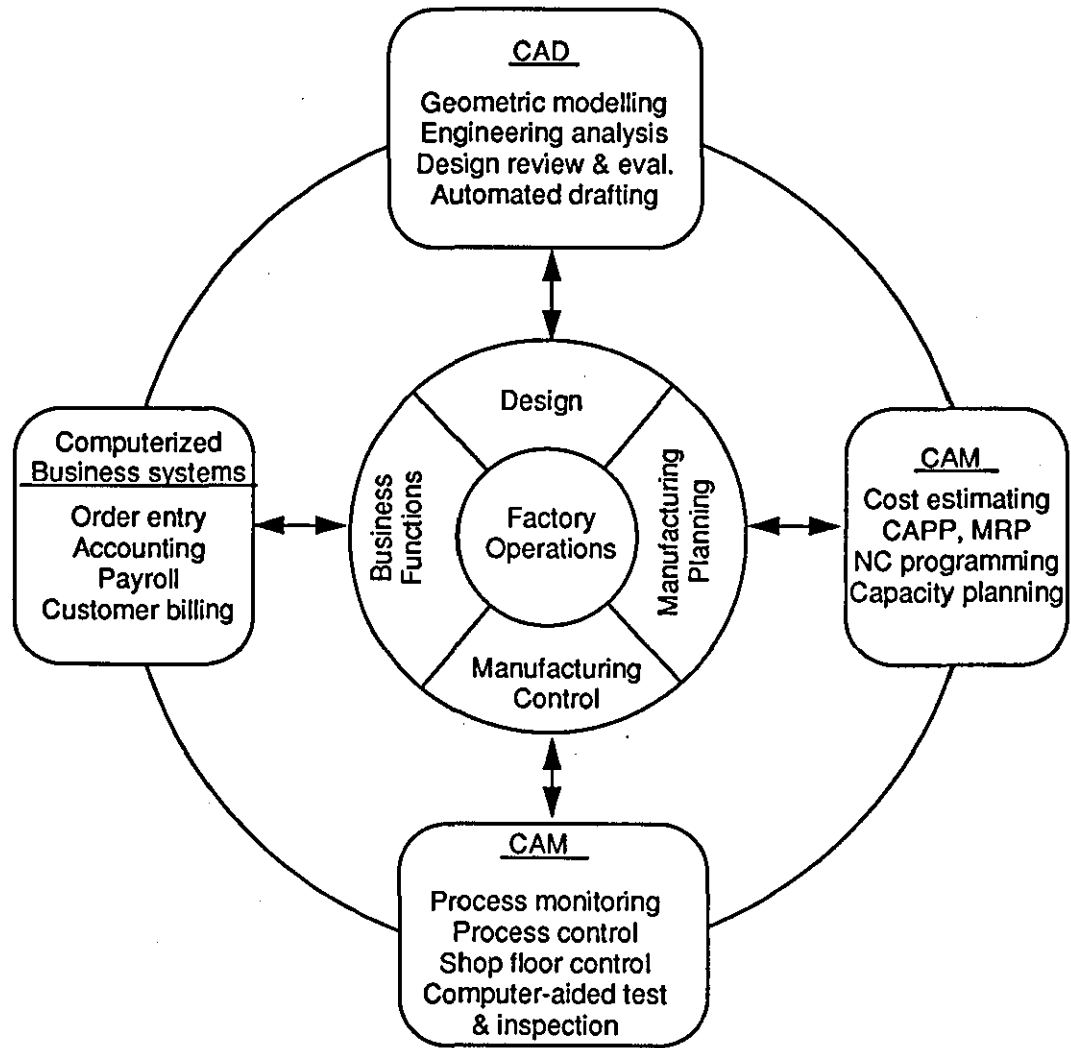


Figure 2.3 CIM Elements [Groover 1987]

fore no conformity is guaranteed for even the same type devices supplied by different suppliers, nor by the same supplier at different time frame.

2.5 Methods and Tools for Manufacturing Systems Integration

The challenge and the potential benefits of CIM have greatly stimulated world-wide interest and research activities in developing appropriate methods and tools to assist in achieving an integration of manufacturing systems. Out of these various research activities and project initiatives, a number of useful tools and underlying methodologies have emerged. Such tools and methods will be reviewed in this subsection, especially with respect to the following areas, viz:

- (1) Networking and communication standards,
- (2) Manufacturing information handling: administration systems and representation format standards, and
- (3) Architectures for manufacturing systems integration.

2.5.1 Networking and Communication Standards

A fundamental requirement of CIM is to provide means by which the physical flow of fragments of information can be realised between systems which need to communicate [Scheer and Hars 1991]. Through electronic information transfer, information sharing and interaction between manufacturing sub-systems can be enabled to allow more timely and effective decision making [Mills et al 1991]. However, in reality this physical link can be established either as a manual delivery mechanism (e.g. using manually delivered floppy disks, magnetic tapes, etc.), or as an electrical connection between communicating systems so as to facilitate electronic data interchange (EDI) [Low and Chee 1991]; e.g. using RS-232 serial transfer, or via local area networks.

The use of suitable local area network (LAN) [Maria 1986] for inter-computer communications has been proven to be an effective means of delivering information quickly and accurately [Stix 1990]. However, as has been acknowledged, data communication over LANs has only partially solved the problem, as “data transfer is one matter but information transfer is another completely” [Weston et al 1988]. Furthermore, there still exists a lack of conformity between the subsystems wishing to communicate and indeed various LAN implementations which can be used to enable communication. As a result, systems built with differing LAN interfaces (i.e. different protocols) are still unable to transfer data between them [Voelcker 1986].

Until recently, proprietary architectures and customised protocols have been the underlying basis for almost all factory networks [Jasany 1986] [Busby et al 1990]. The need for communications standards, to allow direct exchange of information between communicating systems, has stimulated the research activities in the area of communication standardization [Kaminski 1986] [Farowich 1986] [Hansohn 1990] [Frenkel 1990]. Here only a few of very important research projects in communication standards are reviewed.

A. ISO/OSI Standards and the ISO/OSI Reference Model

In 1970 the International Standards Organization (ISO) began work on series of communications standards with a number of international projects initiated under the Open System Interconnection (OSI) banner [Crowder 1985]. In 1977 a reference model for Open System Interconnection, i.e. the ISO/OSI reference model, was made public and gained international acceptance as a framework for LAN development in both the factory and the office environment [Welch Jr. 1986]. The ISO/OSI reference model functionally segments the general communication task a LAN must perform into seven different layers [Maira 1986] [Gray 1991]; therefore this reference model is also known as the “seven-layer OSI model” (see Figure 2.4). The

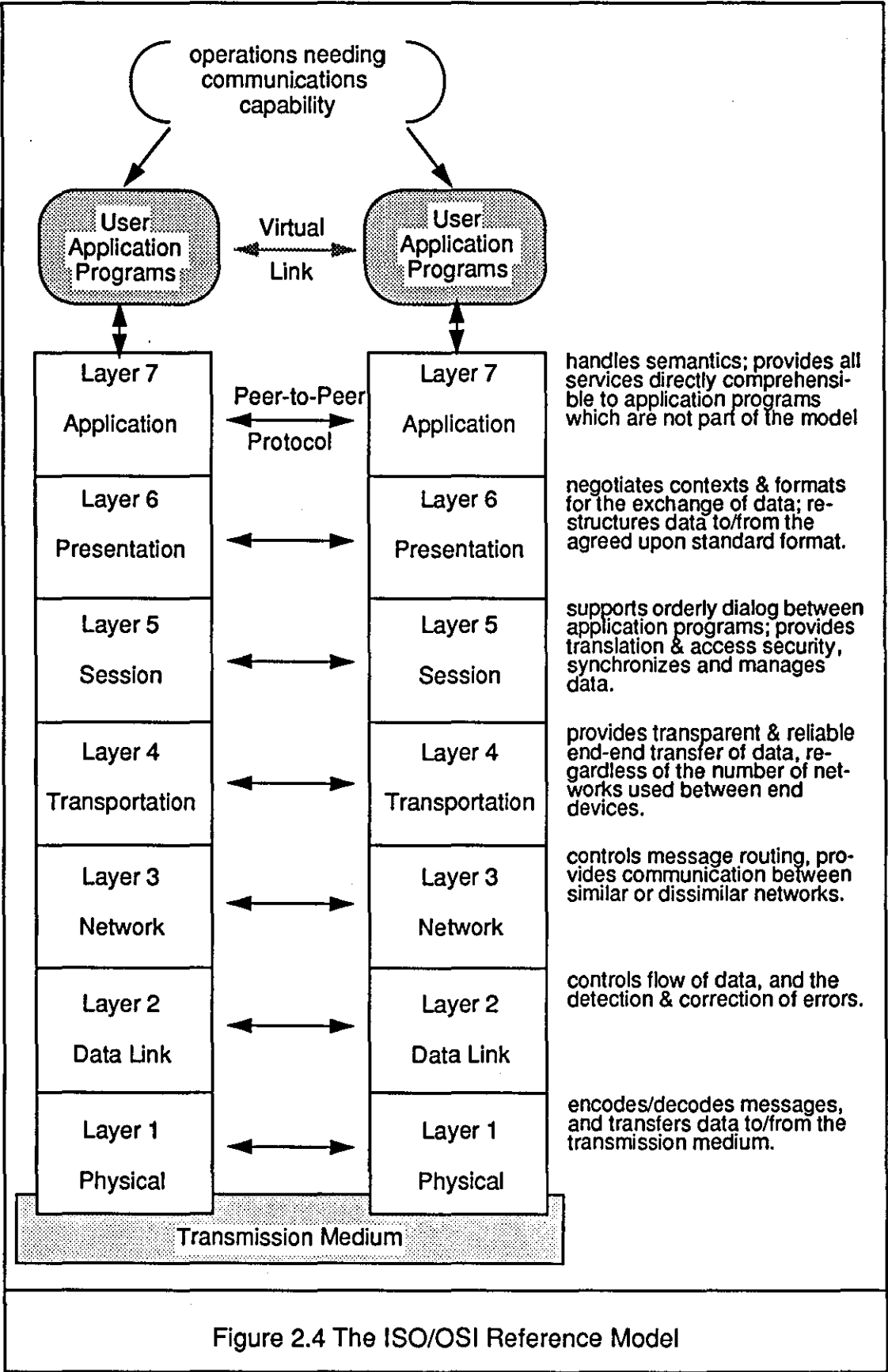
ISO/OSI model is an abstract model; it only provides a blueprint [Amam 1986], but does not specify how the functionality at each layer should be implemented [Day and Zimmermann 1983], i.e. it does not specify what protocol standards should be used at which layer. In order to accommodate the wide variety of applications, there are a number of options within the OSI standards for each of the layers [Graube and Mulder 1984]. See Figure 2.5. However, this in turn gives rise to the possibility of incompatibility between different implementation based on the same reference model [Gray 1991].

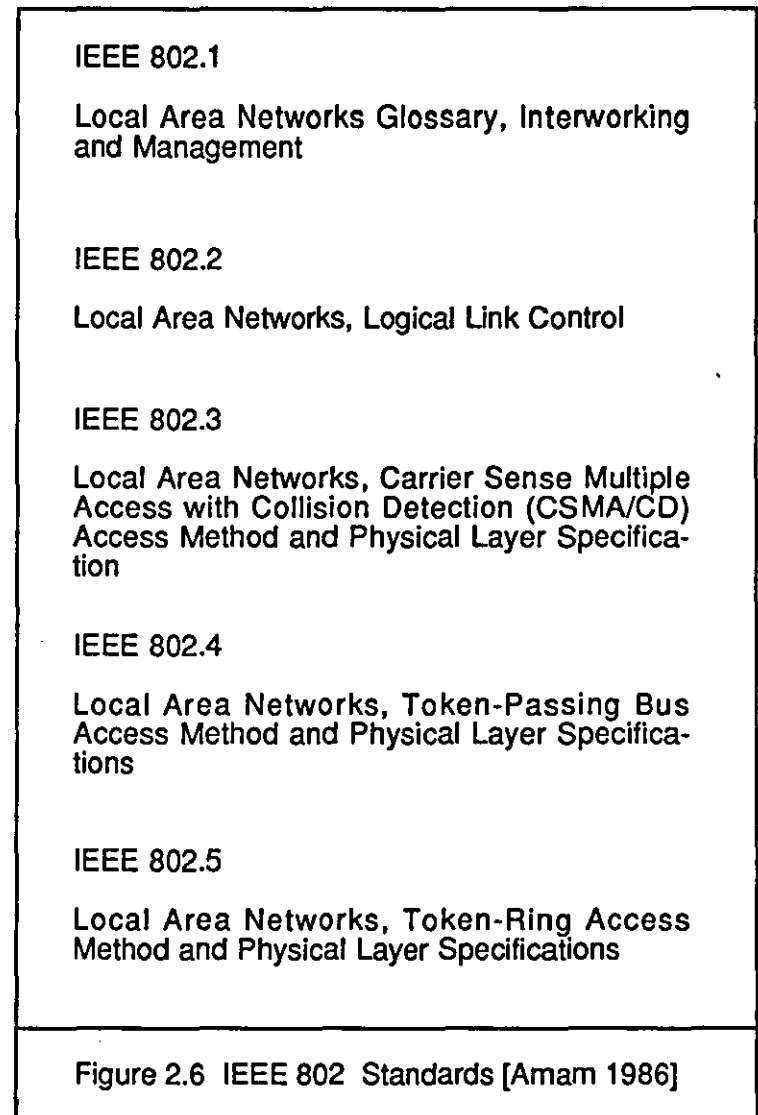
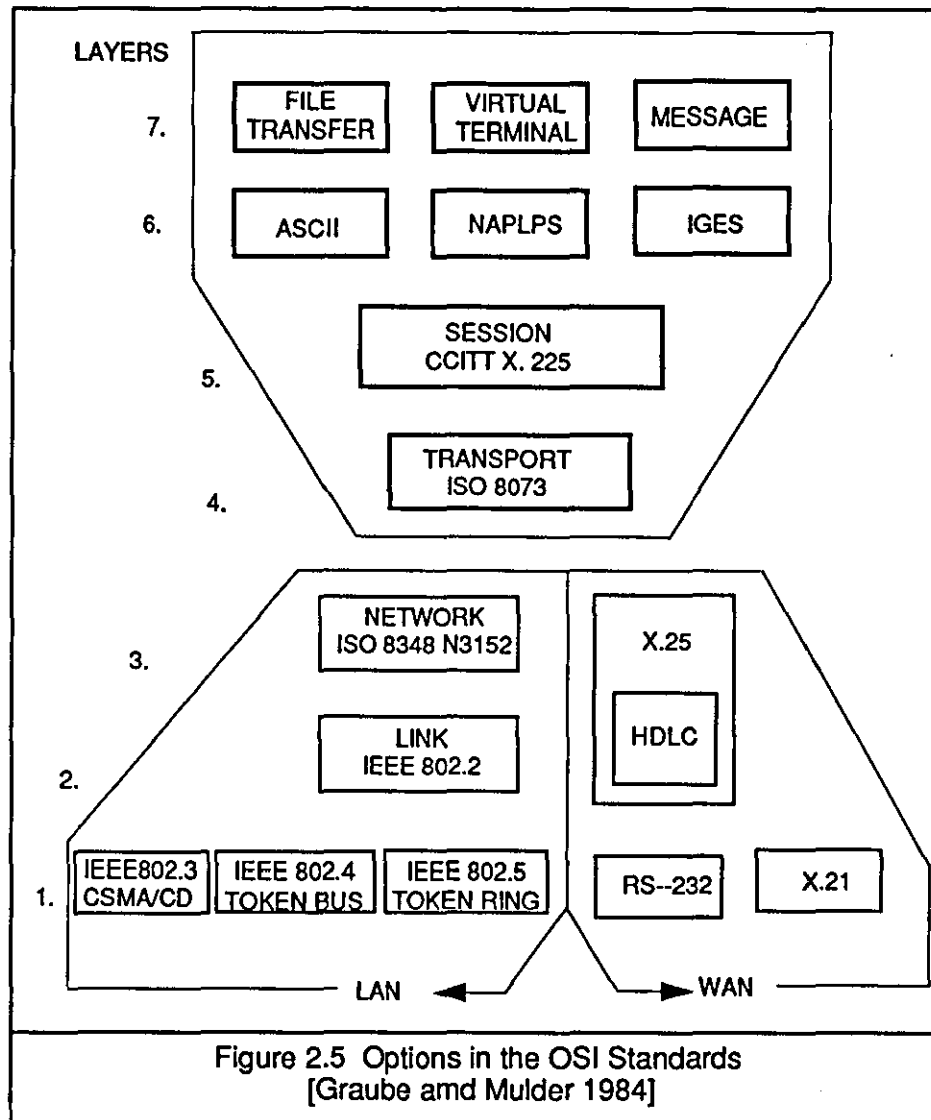
B. IEEE 802.X Standards

In 1980 the Institute of Electrical and Electronic Engineer (IEEE) formed the Project 802 committee to work on specific standards for local area networks [Maira 1986]. The results of this project is a series of LAN standards which are now gaining international acceptance [Amam 1986]. See Figure 2.6. In fact, some of the IEEE standards are chosen as standard protocols which are being included in the MAP/TOP specifications (at the lowest two layers, see Figure 2.7).

C. MAP/TOP

Manufacturing Automation Protocol (MAP) [Kaminski 1986] and Technical Office Protocol (TOP) [Farowich 1986] are two specific implementations of the ISO/OSI model developed and evolved during the last decade. Both MAP and TOP are based on ISO/OSI reference model and are committed to take the best of existing standards where available and specify new protocols where necessary [Moon 1985] [Weston et al 1988] [Breeze 1990]. For example, both MAP and TOP take ISO, IEEE, and DIS (Draft international Standards) standards for the lower layers, and specify their own protocols for the application layer (layer 7), e.g. the Manufacturing Message Specification (MMS) used in MAP 3.0. However, while MAP is intended for factory floor automation where the guaranteed delivery of messages is





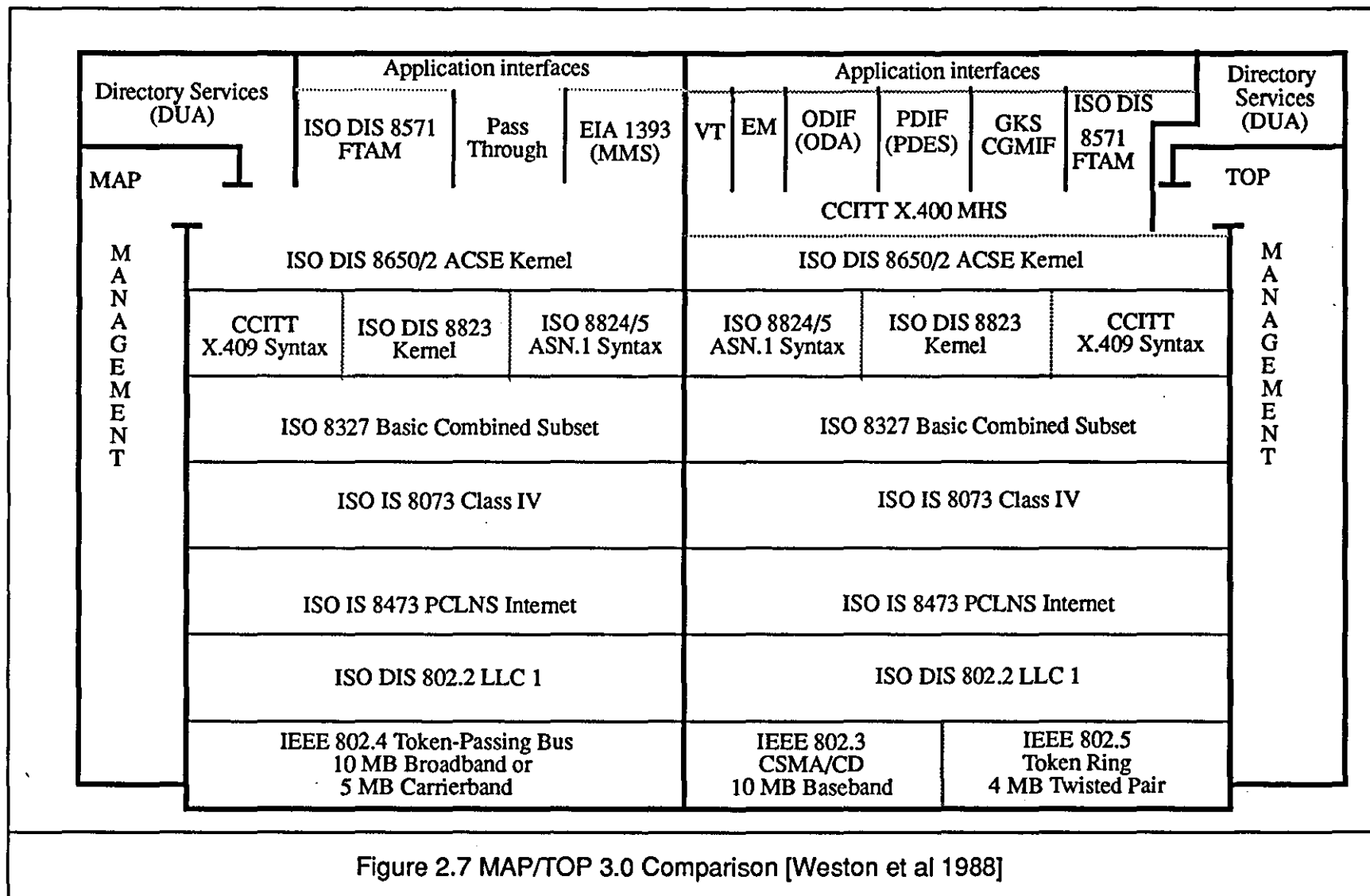
critical (e.g. for real time control), TOP is intended for technical office automation where data traffic tends to involve bursts of large blocks of data but guaranteed delivery within a short time frame is not critical [Amam 1986]; therefore at certain layers different protocols have been adopted in MAP and TOP specifications. See Figure 2.7. The difference between MAP and TOP specifications reflects the reality that communications solutions must address a variety of business applications and each of these applications may require its own coordinated suite of protocols. Moreover, as stressed in [Crowder 1985], the differences between those protocol suites will primarily lie in the lowest and highest level of the OSI reference model.

The development of MAP as the basis for rapid evolution of non-proprietary factory floor communications has received wide attention worldwide, and MAP is evolving standard messages and message handling facilities (e.g. MMS) primarily for shop floor control devices [Weston et al 1988]. However, MAP and TOP again only provide a physical data communication link between computers and computer-controlled programmable devices [Amam 1986]. In other words, MAP (based on the ISO/OSI model) serves to provide two classes of services to applications, namely, interconnection (or networking) and interworking (or interoperation) [Moon 1985] [Gray 1991].

D. SNA

Systems Network Architecture (SNA) [Corr and Neal 1979] was announced in September 1974 by IBM as a seven-layer, hierarchical and single-host network structure [Jarema and Sussengüth 1981]. Since then, the original set of functions has been enhanced to support multiple-host networking [Ahuja 1979].

SNA was designed as a proprietary communication product mainly for the purpose of allowing IBM's own computers to communicate with each other [Gray and McNeill 1979], and has for sometime been widely used as a *de facto* data communication standard, especially among IBM users [Gray 1991]. Although both



SNA and the ISO/OSI reference architecture are seven-layer hierarchical structures, the layers in SNA and OSI model have no mutual conformance. This is due to the fact that SNA preceded OSI, yet IBM were unwilling to give control of its SNA specifications and its further development to the public standard-making body [Gray 1991]. A comparison of the function of each layer in the two architectures is given in Figure 2.8, whereas Figure 2.9 lists the advantages/disadvantages of each protocol suite [Tillman and Yen 1990].

2.5.2 Manufacturing Information Handling

A. Information Administration Systems

The US Air Force ICAM programme has specified and advanced an Integrated Information Support System (I²S²) which is intended to address the problem of information sharing [Weston et al 1988] and to provide mechanisms for managing and controlling information shared between networked computers [Amam 1986].

Furthermore, a three-schema distributed database architecture is incorporated in the Integrated Manufacturing Data Administration System (IMDAS) which has evolved from the work in the Automated Manufacturing Research Facility (AMRF), established by the National Bureau of Standards (NBS) [Libes and Barkmeyer 1988].

As shown in Figure 2.10 the 3-schema information architecture identifies three types of views of a database, namely, 1) a number of external views, 2) a global conceptual view, and 3) a number of fragmented views [Date 1986] [Beyon 1990]. An external view is the view of the database as seen by the individual user, and is therefore also known as the “user” view or “application” view [Kent 1978]. The global conceptual view represents the entire information content of the integrated database and therefore is a combined view of all necessary data required to manage the whole CIM system, which the database is intended to serve. A frag-

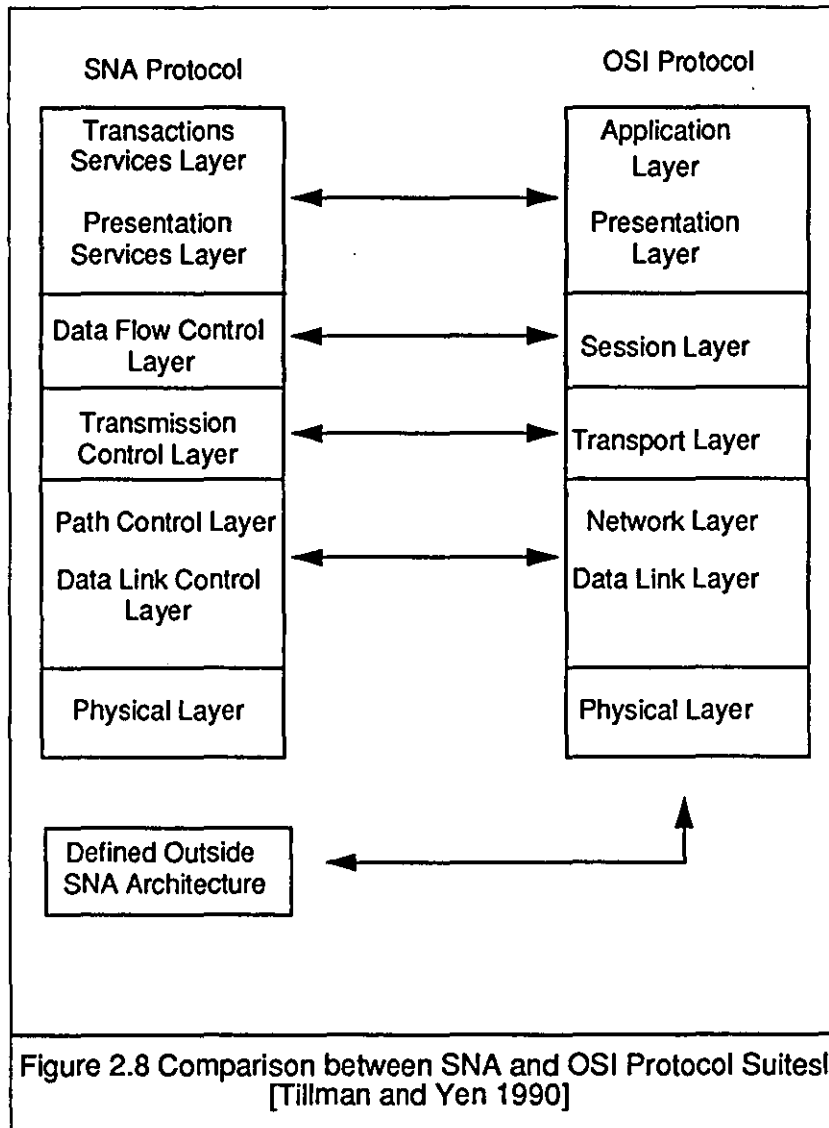


Figure 2.8 Comparison between SNA and OSI Protocol Suites! [Tillman and Yen 1990]

Advantages		Disadvantages	
SNA		SNA	
<ul style="list-style-type: none"> * High-installed base * Excellent vendor and third-party support * de facto industry standard for long-distance data communication * Good connectivity with other proprietary networks 		<ul style="list-style-type: none"> * Little connectivity with open system networks * Slow in adapting to distributed network environment * Design constraints make it difficult to update network * Applications are H/W and S/W dependent 	
OSI		OSI	
<ul style="list-style-type: none"> * International connectivity * Modularity allows for ease of update for each layer without affecting other protocols * Not hardware or program dependent 		<ul style="list-style-type: none"> * Suite not fully defined * Security (interception, alteration, interruption) is tough to maintain * Network management is difficult in open systems 	

Figure 2.9 Advantages and Disadvantages of SNA and OSI [Tillman and Yen 1990]

mented view of the database represents the physical portioning, storage and multiple occurrences of the conceptual data objects across the sub-systems of a CIM system [Weston et al 1988]. The use of three schema information architecture is deemed appropriate for information integration in a manufacturing environment as a means of managing change in manufacturing environments where fragments of information may reside on various storage media and where the location, content and even structure of that information may need to be regularly changed [Weston et al 1989d].

B. Standards for Information Representation

As previously stated, the major functionality of the MAP/TOP specification can be viewed as providing two classes of services to application programs, namely interconnection and interworking [Gray 1991]. However in reality these two classes of services have not been developed evenly, meaning that on the one hand the interconnection services tend to be mature, the interworking services on the other hand are still underdeveloped. This is partially due to the fact that the development and specification of standards for the lower layers of the communication paradigm are relatively less dependent on particular applications, whereas the specification of the higher layers are largely application-specific [Murphy 1990]. For example, the communication needs of file manipulation are quite different from that of real-time control as required by factory floor automation. In other words, the services provided by higher layers are geared more towards the semantics than the syntactic of the communication. This would require the inclusion of protocols for encapsulating production definition data, graphical data, virtual terminal access and office documents [Weston et al 1988]. In this area, a number of standardization initiatives have been focused on evolving standard (or neutral) information representation to facilitate exchange of design intent between dissimilar CAD/CAM systems [Bloor and Owen 1991]. The primary goal of all these initiatives is to facilitate the two-way transfer of data between two or more CAD systems without any loss of information [Brandli and Mittelstaedt 1989], where the neutral format acts as the

“intermediate transfer format” (Figure 2.11). The following few subsections give a brief review of some of the more important initiatives in this area together with a description of important features of resulting standards (many of which are still evolving).

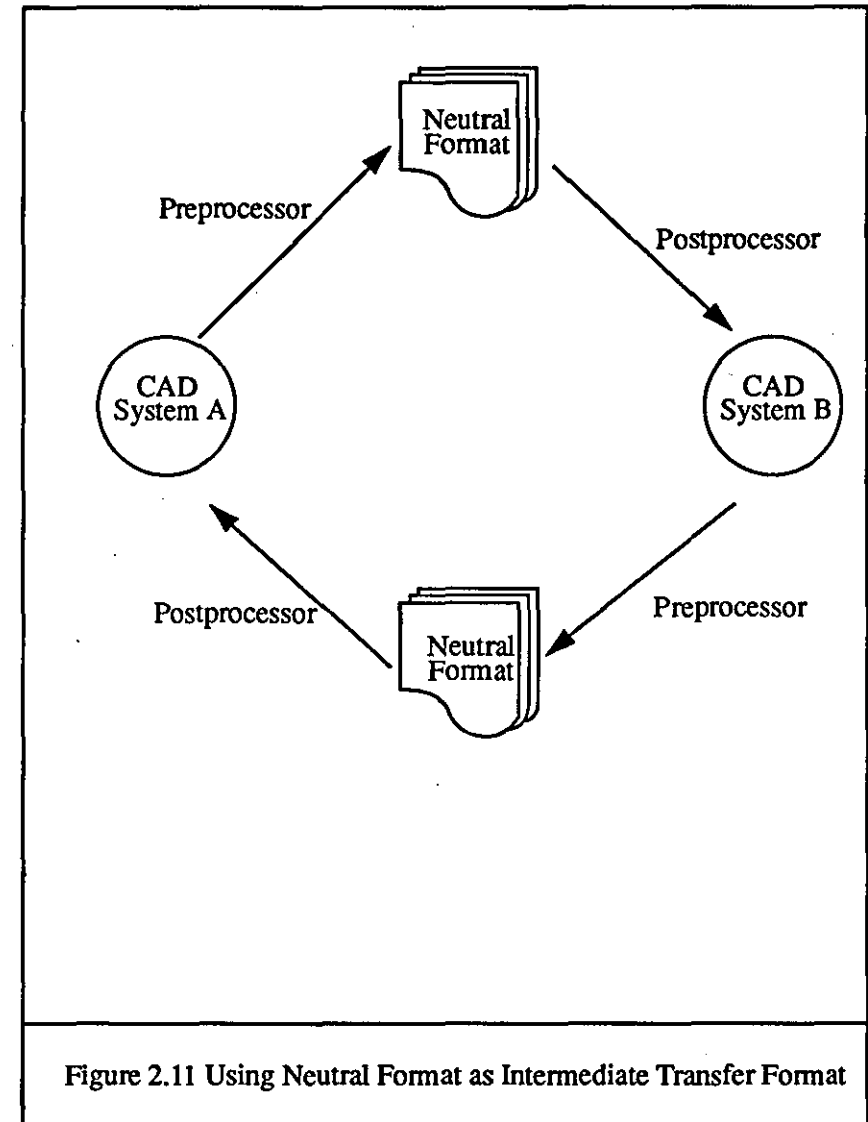
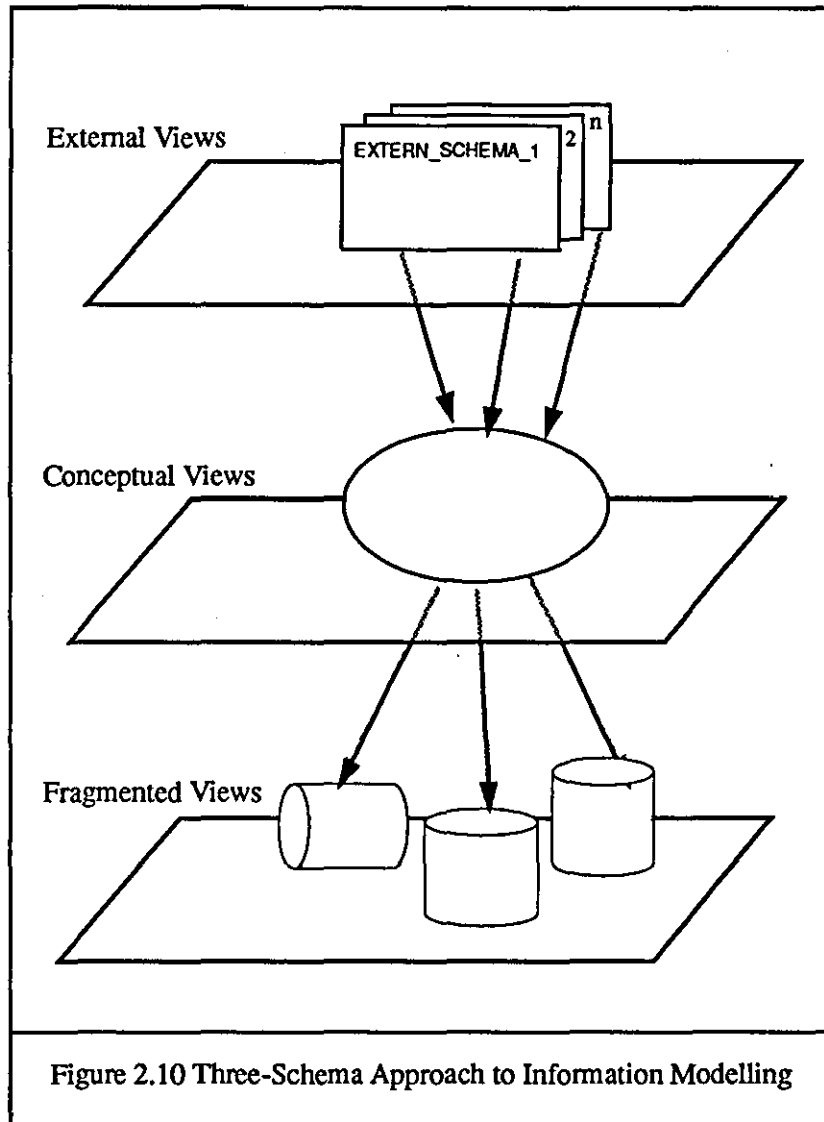
IGES

The IGES (Initial Graphics Exchange Specification) represents an early approach to establishing a neutral format [Owen and Bloor 1987] [Brandli and Mittelstaedt 1989] for describing and transmitting product design and manufacturing information. It is intended to facilitate the meaningful exchange of data which has been created and stored in a CAD/CAM system in a computer readable format. The IGES initiative started in the late 1970s supported by the US Air Force ICAM (Integrated Computer Aided Manufacturing) programme, involving ground work by the Boeing Co. and General Electric Co. [Murphy 1990].

PDES and STEP

PDES (Product Data Exchange Specification) was developed by a third project under the US Air Force programme. Drawing on the IGES, PDDI (Product Definition Data Interface), and XBF (Experimental Boundary File) work, it was developed as a successor to IGES [Owen and Bloor 1987]. It is proposed that PDES should include all product data necessary for the design and manufacture of a product [Weston et al 1988].

STEP (Standard for the Exchange of Product Model Data), first proposed in 1984, was developed under the ESPRIT CAD*I (CAD Interfaces) project which is to define a specification for the capture and exchange of surface, solid and finite element modelling data so as to support a complete representation of a product through its life-cycle [Bloor and Owen 1991]. A three-layer architecture is incorporated in STEP: the application layer, the logical layer and the physical layer.



As has been reported in literature, STEP will be the future standard in exchanging of product model data including complete product definition data and product life cycle data [Brandli and Mittelstaedt 1989] [Bloor and Owen 1991].

EDIF

The Electronic Design Interchange Format (EDIF) was originally developed by six large electronics companies in the US; with support from its European representations, it is rapidly becoming a de facto standard for the exchange of electronic CAD design data [Owen and Bloor 1987]. EDIF is designed for representing IC (integrated circuit) design data in a standard format which can facilitate information transfer from design to fabrication environment [Murphy 1990], and thus EDIF has been welcomed by many electronics companies [Eurich and Roth 1990].

Compared with IGES/PDES, EDIF uses a well defined LISP-like language for data definition. The latest EDIF version 2.0.0 was released in 1987, which has already been used to establish “links” between design and manufacturing of VLSI (very large scale integration) [Burson 1987]. Attempts were made to use the original EDIF 2.0.0 syntax (which is mainly intended for representing VLSI design data) for describing PCB layout [Racal-Redac 1987]. See Figure 2.12. However, it was found later on that this was unsuccessful due to the limitations of the EDIF 2.0.0 syntax; those triggering the research into conceptual modelling of PCB layout by the EDIF committee which has been focused on expanding EDIF syntax to include provision for describing PCB layouts [EDIF-PCB-TSC 1990].

2.5.3 Manufacturing Systems Integration Architectures

A. Three architectures

The need for separate integration architectures to enable the creation of highly configurable, visible and maintainable CIM system was identified by the Sys-

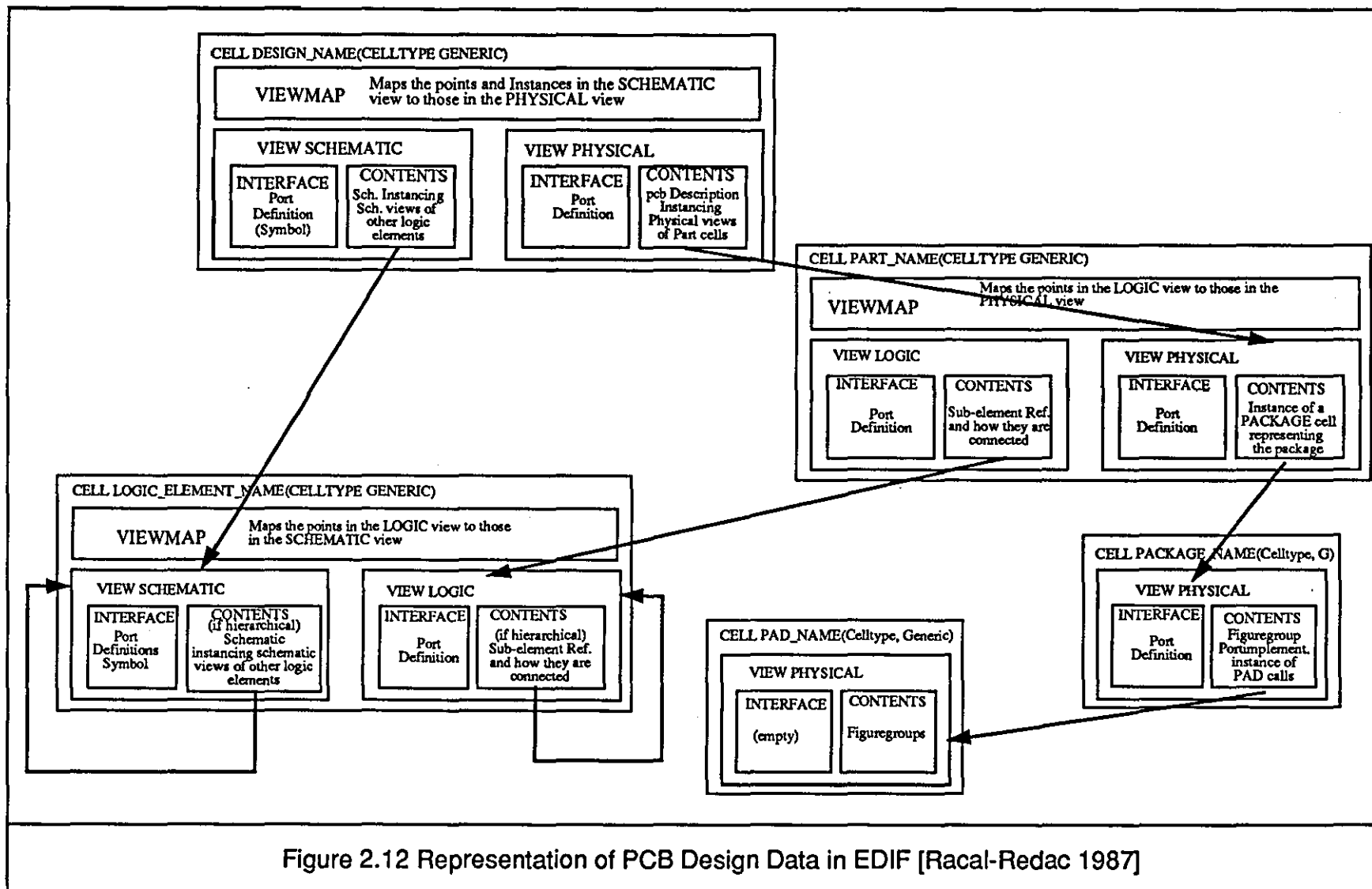


Figure 2.12 Representation of PCB Design Data in EDIF [Racal-Redac 1987]

tems Integration Group at Loughborough University of Technology (SIG/LUT). This identification has led to the notion of three integration architectures [Gascoigne 1987] [Weston et al 1989a] [Weston et al 1989c] as depicted in Figure 2.13.

According to Sumpter [Sumpter et al 1987] and Weston [Weston et al 1989a] [Weston et al 1989b], the “network architecture” is to support information exchange (sharing); the “information architecture” is to provide, e.g. by means of database technology, the required information to support various applications; the “application architecture” is to implement system management and supervisory functions, i.e. concerns with the administration of applications (or manufacturing actions) which normally involves defining, building, debugging and running manufacturing applications [Weston et al 1989a]. As such, the development in communication protocol standards (e.g. OSI standards, IEEE standards, MAP/TOP specifications, etc.) can be considered as only providing networking tools/services which can be utilized to form the “communication (network) architecture”, whereas development in information administration systems (IAS) such as I²S² and IMDAS concepts providing methodologies/tools required to form the “information architecture” [Weston et al 1988].

Recent development by the group is the notion of “soft integration platform” called CIM-BIOSYS (CIM-Building Integrated Open SYStem) [Weston et al 1990] which includes a set of configuration tools that can be used to produce, run and maintain soft integrated manufacturing systems. In other words, it provides a set of integration services which support the operation of (and interactions between) open application processes [Weston 1991b]. The platform allows conformant manufacturing devices to be “plugged” in and non-conformant devices to be connected by means of “temporary bridges” known as “alien application handlers”. It also allows the manufacturing applications to be distributed and run across a network of computers and machines [Leech et al 1991]. Software services (i.e. communications, information, control and administration) are provided by the platform to all devices plugged in or connected to it. The above-mentioned three integration architectures

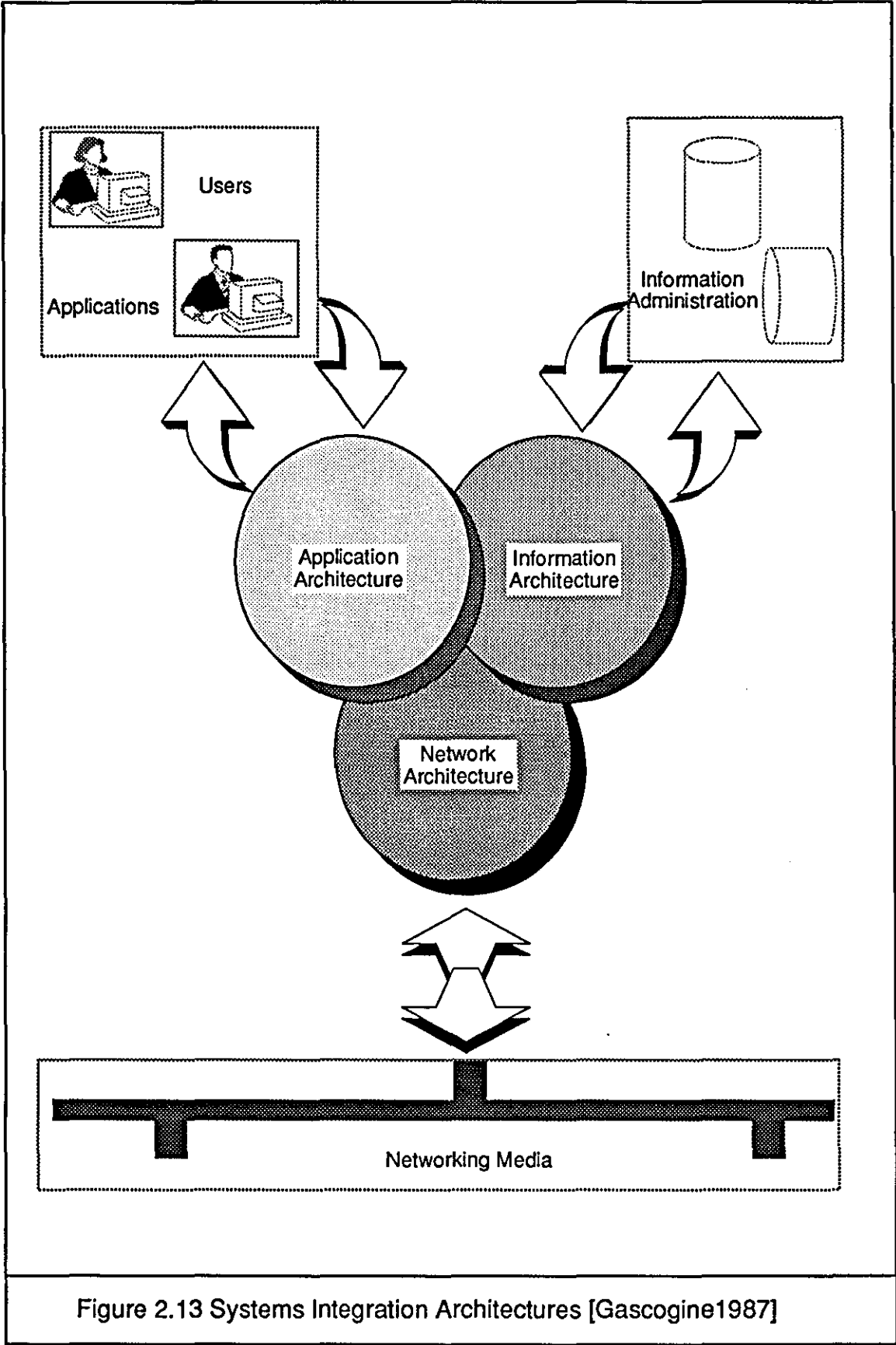


Figure 2.13 Systems Integration Architectures [Gascogine1987]

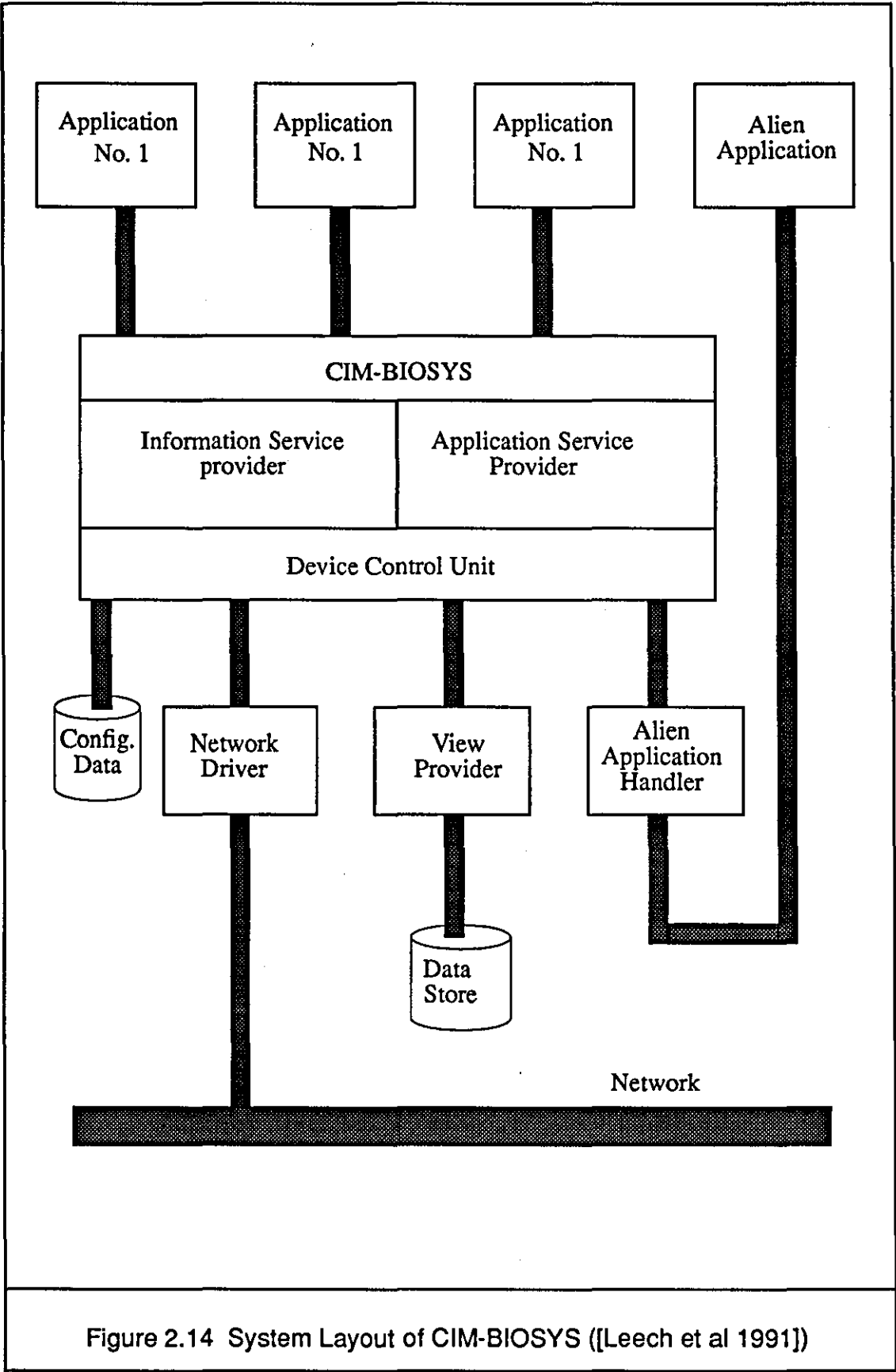
are reflected in CIM-BIOSYS by the three main sections of the platform, namely, application service provider, information services provider and device control unit. The system layout of the CIM-BIOSYS is illustrated in Figure 2.14

B. ESPRIT CIM-OSA

The European Strategic Programme for Research and Development in Information Technology (ESPRIT) was launched in 1984 [Jorysz and Vernadat 1990a], of which CIM is a key area of research. The aim of the ESPRIT CIM programme is to develop standards and technology for multi-vendor systems, with the technical goal being to develop (i) a generic CIM reference architecture for the creation and execution of enterprise models, i.e. a “modelling framework” [Panse 1990] [Jorysz and Vernadat 1990a] [Jorysz and Vernadat 1990b], and (ii) a set of rules for building CIM systems based on the architecture, i.e. an “integrating infrastructure” [Klittich 1990] that can support model engineering within heterogeneous manufacturing and information technology environments [Kosanke 1991]. The Computer-Integrated Manufacturing Open Systems Architecture (CIM-OSA) thus resulted. See Figure 2.15.

2.6 Computer Vision Technology and Its Applications in Manufacturing

Computer vision (or machine vision) is a multi-disciplinary area of study, involving electronics, optics and sensing, computer science, vision algorithms, image processing, pattern recognition and artificial intelligence [Sanz 1988]. It can be viewed as an area of pattern recognition which deals with the analysis of images and scenes, and is mainly concerned with enabling computers to handle visual (or pictorial) input data [Fu and Rosenfeld 1984]. As far as industrial application is concerned, computer vision systems can be applied in almost all manufacturing processes to provide invaluable data and information for supporting the realisation



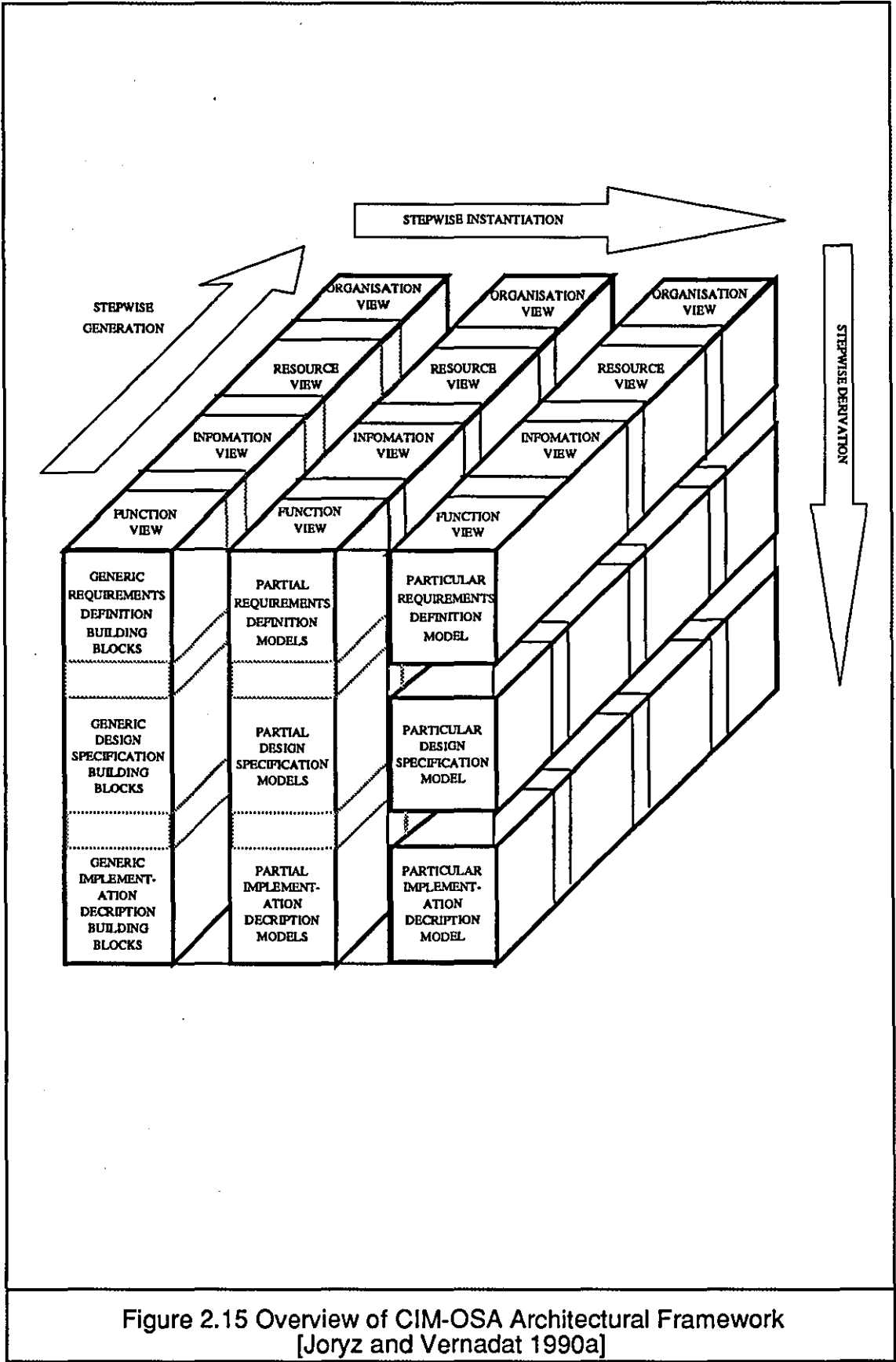


Figure 2.15 Overview of CIM-OSA Architectural Framework
[Joryz and Vernadat 1990a]

of computer integrated manufacturing [Zuech and Miller 1989], and adding a high level of flexibility to factory automation [Kent and Shneier 1986].

2.6.1 The Building Elements of a Computer Vision System

• There are potentially many ways in which a computer vision system can be applied industrially [Dunbar 1986]. However, despite the fact that varying applications may have varying requirements specifying different physical configuration of the vision system hardware and software [Welsh 1991], the underlying composition and concepts of contemporary computer vision systems are essentially similar. Moreover, in almost all cases, the generic goal of a computer vision system is to derive a description of a scene by analysing one or more images of the scene [Rosenfeld 1987], or to construct an explicit and meaningful description of physical objects from images [Ballard and Brown 1982], though apparently the content and format of the description may vary from one application to another.

Conceptually, from the system composition point of view, regardless the context of application, a computer vision system will typically include an input (sensing) device (e.g. a camera or a scanner, and an appropriate frame store), a host computer and a set of image processing algorithms and application software [Hollington 1984]. However, from a research point of view, the heart of a computer vision is its software [Rossol 1983]. The primary operations of a typical vision system comprise three functions (see Figure 2.16) [Groover et al 1986a] [Vernon 1991] [Braggins and Hollington 1986], viz:

- (1) Sensing and digitizing of the incoming data describing a scene, i.e. image formation and acquisition,
- (2) Image processing and analysis, and
- (3) Image interpretation, and application-specific action-activating and/or decision-making, based on the information generated by

function 2).

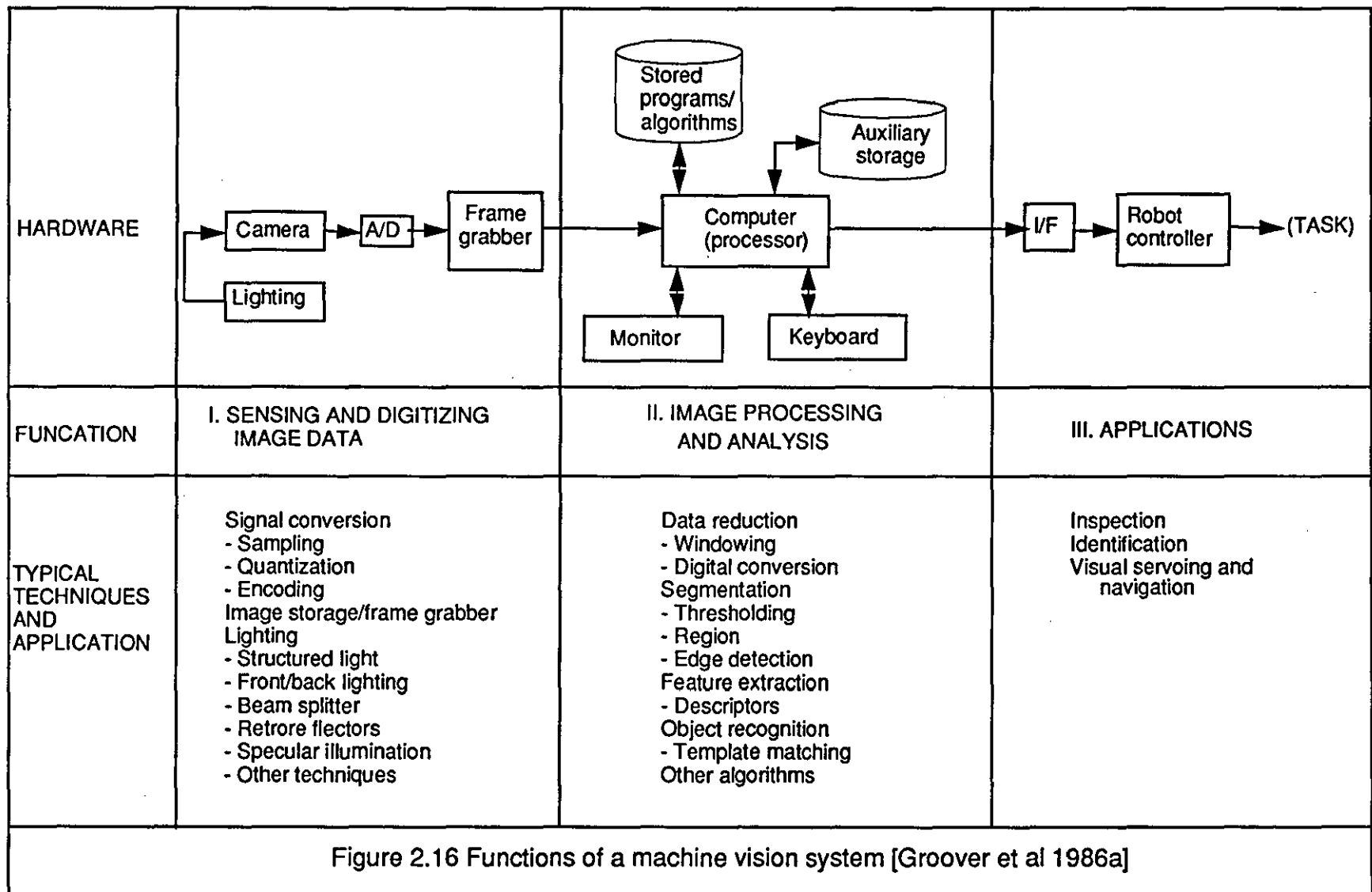
In the following subsections, literature concerning issues (2) and (3) listed above are reviewed, as these are closely related to the objective of this research. A good review of techniques concerning the issue of illumination, sensing and image acquisition techniques can be found in [Batchelor 1985a].

Although the real world is geometrically three dimensional, there are many application situations where the scene under examination is itself essentially of a two-dimensional (2-D) nature [Rosenfeld 1987] [Vernon 1991]; for example, a bare PCB or a master film under inspection, a flat part laying on a flat background (e.g a conveyer belt), and so on. Furthermore, in a computer vision system, almost all the images processed are in nature two-dimensional, typically comprising of digitized values arranged in a 2-D array. Not surprisingly, 2-D image processing techniques are relatively well-developed and form fundamental building blocks of some of the 3-D computer vision techniques.

2.6.2 2-D Image Processing and Analysis Techniques

Generally, computer vision systems are capable of dealing with 2-D images much more easily than with 3-D images. However, even with a 2-D computer vision there are still many problems which remain as bottlenecks to the development of computer vision, and thus have stimulated great interests and effort in research.

One such problem is related to the sheer amount of data to be processed by a vision machine; for example, an image with 512 by 512 pixels with 8-bit grey levels will give a total of 2,097,152 bits of data which have to be processed by the vision system in say 1/30 second (obviously the time will be dependent on the requirements of a given application). In order to reduce the amount of data to a manageable level, various techniques have evolved. The two main techniques for image



data reduction are digital conversion and windowing [Groover et al 1986a]. The former involves representing an image with a smaller number of bits (less grey levels), with the extreme being binary representation; the latter involves using only a portion of the total image.

Another class of data reduction techniques incorporate the structural transformation of the image data being processed; for example, representing an image in a more compact string-like data structure instead of the original matrix. Chain codes [Fairhurst 1988] are widely used to encode boundaries of objects in a binary image. This approach is attractive because it uses only eight numerical digits to represent the eight possible directions along the boundary of an object, yet the resultant representation not only reduces the amount of data representing the image, but also makes the process of extracting certain features (e.g. area, perimeter, moments, etc.) fairly straightforward. Another useful data reduction method in this class is the so-called run-length codes [Batchelor 1985b], which in fact adopts a line-by-line scanning approach. However, there exists an additional problem inherent in this approach, namely, that of relating one scan line to the next; therefore analysing the line-to-line relationship is much more difficult [Tanimoto 1980] with run-length codes than with the original arrays.

Traditionally, 2-D image processing techniques can be classified loosely according to the stages at which computer vision task is performed; these being:

- (a) preprocessing,
- (b) segmentation,
- (c) feature extraction, and
- (d) recognition.

A. Preprocessing

In a computer vision system, the first task starts with acquisition of an image of the scene (e.g. the natural outdoor scene, an industrial assembly, a printed circuit board, etc., depending on the application concerned). This is usually done utilizing sensors which transform the real world three-dimensional scene into two-dimensional arrays of digitized image data. For this purpose, two basic types of video cameras [Dunbar 1986] are usually utilized, namely 1) the vidicon, which is based on vacuum-tube technology, and 2) solid-state camera, which is based on the semi-conductor technology. Often, with either technology, the resultant image data is, to a certain extent, degraded by electronic noise and/or other types of distortions (e.g. optical distortions) introduced during this process of image acquisition [Ballard and Brown 1982] [Ruocco 1987].

As a result, it is often desirable that certain preliminary operations be performed on the digital image data (though in application this process may not necessarily distinctly be isolated from the succeeding stages of image processing) with a view to removing as much the noise as possible and correcting the distortions, or to enhancing certain features (e.g. the grey-level intensity uniformity [Benhabib et al 1988]) of the image so as to improve its “quality” [Rosenfeld 1969]. Such operations are generally referred to as preprocessing [Ruocco 1987], or early processing [Ballard and Brown 1982], or image enhancement [Rosenfeld 1969] [Castan 1977] [Galbiati Jr. 1990]. Arguably, these operations should not remove critical information contained in the original image [Benhabib et al 1988]. Operations for such purposes are usually considered to be at the lowest level of image processing hierarchy [Offen 1985], and are performed mainly on a pixel-by-pixel basis. In fact, most of the image pre-processing techniques are based on digital filtering techniques, which is a general term given to a class of operations transforming the image intensity values so as to enhance or depress certain features of the image [Ballard and Brown 1982]. Two classes of digital filters can be further distinguished, i.e. linear and non-linear filters.

Linear filters are usually realised by means of convolving the image with some defined “masks” [Vernon 1991]. As the convolution is being performed, a weighting factor, represented individually by a mask value, will be applied to each corresponding pixel of the image, and the resultant weighted sum will be assigned as the new grey-level of the pixel of the image which corresponds to the centre of the mask. For example, a low-pass-filter can be realised by convolving the image with an “averaging-mask”, which serves to suppress noises with high spatial frequency, and the image is thus “smoothed”. However, the noise reduction is done at the expense of bandwidth [Svetkoff et al 1987], causing the image to blur and thus the edge transition is inevitably degraded.

Equally important in the class of linear filters, are the so-called “high-pass-filters”. These are again realised by convolution, but serve to highlight intensity discontinuity of the image, resulting in a “sharpened” image; therefore it can be used as a means of enhancing the image before edge detectors are applied.

Non-linear filters are also frequently used for noise reduction and image enhancement. A good example is given as the median filtering [Castan 1977] whereby the grey-level of a pixel is replaced with the median of the pixel values in some local neighbourhood (e.g. a 3 x 3 window). Median filters can be used for noise reduction, whereas at the same time preserving sharp intensity discontinuity as normally found on edge transitions [Svetkoff et al 1987], and hence preserving certain edge shapes [Schalkoff 1989]. Thus in general, median filters are superior to the above-mentioned linear mean filters (e.g. low-pass-filters) in that image blurring is kept at minimum [Vernon 1991]. However, inherently, median filters are computationally expensive, as it requires in the first place a substantial sorting operation [Low 1991]. Moreover, serious size and shape distortions can be resulted from the use of median filters; therefore, as commented by Davies, median filters should not be applied especially in situations (e.g. industrial inspection) where precise measurements are required [Davies 1989c]. Thus median filters tend not to be used much in industrial machine vision applications [Vernon 1991].

B. Segmentation

Image segmentation is a general term applied to a variety of techniques used to extract subsets of an image that (hopefully) correspond to the relevant parts of the scene under consideration [Rosenfeld 1987]. Although a great number of techniques are available, they are almost all *ad hoc* in nature; there are no general algorithms which work for all images [Fu and Mui 1981], or, as Haralick put it, “there is no theory of image segmentation” [Haralick 1983]. The various techniques can be categorized into three classes [Fu and Mui 19981] [Groover 1986a] [Rosenfeld 1987] [Vernon 1991], namely, 1) thresholding, 2) edge detection, and 3) region growing (extraction).

Thresholding is a very popular technique which is most widely used to segment an image by means of grey-level transformation [Fu and Mui 1981] [Rosenfeld 1983] [Vernon 1991]. It is essentially a binary conversion method of which the main operation is to set all the grey levels below a certain pre-defined level (i.e. the threshold) to zero, and those above to a maximum brightness level (say 255) [Groover et al 1986a] [Low 1991]. Many preprocessing techniques (e.g. smoothing, image sharpening, etc.) can be applied prior to thresholding so as to obtain a thresholded image where certain features in the original image are highlighted or suppressed [Rosenfeld 1969]. For example, smoothing followed by thresholding [Davis et al 1975] will yield a binarized image containing less “noise points”.

A crucial problem in using thresholding-based image segmentation methods is how to select the threshold. In practice, a histogram (representing the statistical frequency distribution of grey levels in the image) is often employed to guide the selection of a proper threshold value. Where appropriate, a global threshold can be applied to the entire image, based on grey level histogram, or local properties [Fu and Mui 1981]. However, one should not assume that this will always be the case. On the contrary, there are many cases of industrial applications where it is not possible or not appropriate to use a single threshold for the whole image, necessitat-

ing the adoption of a more flexible “varying threshold” scheme [Rosenfeld 1987], or “adaptive binarization” [Castan 1977], or “adaptive thresholding” [Ruocco 1987], so as to properly segment the image. A review of threshold selection techniques is given by [Weszka 1978]; and in [Vernon 1991], these techniques are summarised as being falling into one of the following three classes, viz:

- 1) Global thresholding, in which the threshold operation is dependent only on the gray-level of the point.
- 2) Local thresholding, in which the threshold operation is dependent on the neighbourhood property of the point (e.g. the average gray-level) and on the grey-level of the point.
- 3) Dynamic thresholding, in which the threshold operation is dependent on the point coordinates (i.e. position-variant thresholding), a neighbourhood property and the grey-level of the point.

Edge detection is an image segmentation technique based on the detection of discontinuity in image intensity [Fu and Mui 1981], assuming that sharp changes in grey level (intensity) occur in pixels which lie on the boundary or edges of an object. This approach has been motivated by the following factors [Rosenfeld and Kak 1976], namely (i) most of the information of an image lies on the boundaries (or contours) between different regions [Niemann 1979], and (ii) biological visual systems appear to make use of edge detection, but not of thresholding. Edge detection operations can be further broken down into two steps, namely a) edge-element extraction and b) edge-element combination.

In step a), various local edge operators [Levialdi 1983] are utilized to extract edge elements (e.g. an edge point) from the original image by means of measuring any local discontinuity in intensity or its gradient [Ballard and Brown 1982], and a large number of operators have been proposed. For example, linear operators such as Laplacian [Pratt 1978], and non-linear operators such as Sobel

[Duda and Hart 1973], Roberts [Galbiati 1990], Prewitt and Kirsch operators [Ballard and Brown 1982], are commonly used.

The set of “edge pixels” or “edge points” (i.e. generated by applying those various edge operators on the image) seldom characterises a complete boundary due to noise present in the image (e.g. resulting from non-uniform illumination) [Gonzalez and Wintz 1987], and often containing both spurious edges and discontinuity in valid edges [Fairhurst 1988]. Thus it is necessary in step b) to further process those edge pixels so as to eliminate false elements and to merge, link and assemble genuine ones into longer edge elements [Fu and Mui 1981] or meaningful and complete object boundaries. This further processing generally involves procedures such as heuristic searching [Martelli 1976] [Levialdi 1983], curve fitting [Montanari 1971], smoothing and thinning, local and global edge grouping, and so on.

Hough transforms [Duda and Hart 1972] are frequently used for extracting high-level shape information (e.g. line and curve segments) from those edge primitives (i.e. edge points) detected by various edge operators so as to assist the detection and interpretation of object shapes [Low 1991]; therefore having a great role to play in image understanding and interpretation [Boyle and Thomas 1988]. Good introductions to the basic principle of the Hough transform can be found in the literature including, for example, [Duda and Hart 1972] [Ballard and Brown 1982] [Davies 1986] [Boyle and Thomas 1988] [Low 1991].

In general, simple and well-defined shapes (e.g. straight lines and circles, etc.) can be extracted using basic or modified Hough transforms; e.g. for detecting straight edges or circular objects in industrial inspection [Davies 1986] [Davies 1987]. More complicated (e.g. a polygon) and ill-defined shapes (e.g. an arbitrary shape) can be detected by means of Generalised Hough Transforms (GHT) [Ballard and Brown 1982] [Boyle and Thomas 1988] [Davies 1989a] [Davies 1989b]. Several practical shape detection techniques based on the Hough transforms and their variations have been given in [Low 1991]. As far as circle detection is concerned,

Hough transform is considered to be the most accurate and the only one which can generate reliable evidence from partial features [Boyle and Thomas 1988]. However, the main drawback associated with Hough transform is the computational load it introduces, especially for complex shapes with unknown size and/or orientation [Boyle and Thomas 1988] [Low 1991].

Region growing is a collection of segmentation techniques in which individual pixels are grouped into regions based on attribute of similarity [Pratt 1978] [Pavlidis 1982]. This is a method to achieve better-quality regions by requiring that the result be locally consistent. Further improvement to the resultant region quality (referred to as finding globally consistent regions) can be attained by means of multiple “split-and-merge” processes in which regions are split if they are inconsistent, and merged if their union is consistent [Rosenfeld 1987].

After the boundary of a region is formed, it would be necessary to describe each of the segmented regions with certain geometric representations, which not only further reduce the image data (with a more compact representation), but also make it easy to extract certain features [Batchelor 1985b]. A range of existing approaches to representing 2-D geometric structures has been reviewed by [Ballard and Brown 1982]. The techniques are classified into two broad categories, namely boundary and region representation. Chain coding method is found to be the most commonly used geometric representation of boundaries (object contours) [Fairhurst 1988]. For region representation, the quad trees [Samet 1980], the medial axis transform [Rosenfeld and Kak 1976], and the run-length code are often used. Note that chain code can also be used for region representation by means of specifying a closed boundary of the region. While it is convenient using chain code to represent a single edge contour, chain code does not allow different curves to be related to each other [Batchelor 1985b].

C. Feature Extraction

After the image has been segmented, and a proper representation of the isolated object has been formed, it is necessary to further abstract and extract one or more features which uniquely characterize the object concerned [Groover et al 1986a]. The features can then be manipulated to assist the recognition (or classification) of the objects and thus help achieve an understanding of the image under examination [Vernon 1991]. The techniques utilized to generate such features of the objects, often from the already segmented image, are usually referred to as feature extraction techniques. Generally, the selection of appropriate features and choice of feature extraction techniques are application-dependent [Fu and Rosenfeld 1984]. However, in industrial applications, there are some shape and texture features which are commonly used [Galbiati 1990]; for example, area, perimeter length, compactness (roundness), diameter, centre of gravity, various moments, and so on.

Primitive features (e.g. edge points or segments) can be generated at stages of edge detection. Later on, these primitive features can be processed to generate higher level features, for example, by means of Hough transforms described earlier.

D. Recognition

A further task of computer vision is concerned with recognition (or identification) of the object contained in the image, based on the information (e.g. features) extracted in preceding stages. Here the most common techniques can be classified into two major categories [Rosenfeld 1987] [Groover et al 1986a] viz:

- (1) template matching techniques, and
- (2) structural matching techniques.

Theoretically speaking, these techniques all stem from the more general pattern recognition (classification) techniques [Fu 1980]. Template matching is a

subset of the statistical pattern recognition techniques [Groover et al 1986a]; it serves to classify objects into pre-defined categories. The concept of template matching is simple and straightforward, involving comparison of the similarity between a calculated feature set and a stored feature set defining a template. In operation, this will involve the translation of the template to all possible positions in the image and evaluation of the percentage-of-match between the template and the image at that position [Vernon 1991]. Template matching is mostly suited for applications with less scene complexity. In cases where the desired object is simple and its shape is precisely known, these methods can be very effective, even without the need to extract the object explicitly out of the image [Rosenfeld and Kak 1976] [Rosenfeld 1987]. However, in cases where distortions and noise present in the image, or the shape of the objects are ill-defined, it is necessary to combine template matching techniques with other more sophisticated algorithms or to extend the template matching techniques to include the so-called “flexible template” (or “deformable template”, or “rubber template” technique) [Fu 1980].

Structural matching is a technique in which a complex object is recognized by means of analysing pieces (or portions) of it and the relationships among them [Rosenfeld 1987]. This approach has a theoretical root [Groover et al 1986a] in “structural pattern recognition” or “syntactic pattern recognition” techniques [Fu 1980] [Fu 1982] [Fu and Rosenfeld 1984] [Schalkoff 1989b]. This technique is mostly suited for applications where the object to be recognized is relatively complex in structure, making it computationally time-consuming for complete pattern recognition. Thus it is more appropriate to search for simpler regions or portions of the entire image, and upon the analysis of these simpler regions and the relationships among them, the entire image can be interpreted [Groover et al 1986a]. As will be described in later chapters, structural analysis techniques form a theoretical basis for part of this research (i.e. for analysis of PCB images in AOI applications).

2.6.3 3-D Computer Vision Techniques

The generation of 3-D information and the derivation of 3-D models of the scene is itself an important area of research. Various techniques have been developed which can be used to generate certain 3-D information [Zuech 1988]. These techniques can be generally classified into two categories, namely, 1) direct methods and 2) indirect methods.

Direct range sensing or measuring [Rosenfeld 1987] [Ruocco 1987] involves direct calculation of distances between object being sensed and the image plane. Techniques of this category include triangulation, laser range finder ("time-of-flight" approach) [Nitzan et al 1987], static range finders, structured lighting, stereo vision [Ruocco 1987].

Indirect methods derive range information by inferring from a single 2-D image (monocular image) by taking into consideration of geometric constraints or geometric characteristics of the objects to be recognized [Nitzan et al 1987]. Classes of so-called "range from . . ." methods for range measurement [Nitzan et al 1987] and "shape from . . ." techniques for surface orientation [Rosenfeld 1987] have been developed. For example, range from focusing [Jarvis 1983] [Krotkov and Martin 1986], shape from shading [Horn 1975] [Ikeuchi and Horn 1981], shape from texture [Witkin 1981].

2.6.4 Industrial Applications of Computer Vision

It is envisaged by a number of authors that computer vision will revolutionise manufacturing in two important areas [Rossol 1983] [Hollington 1984] [Vernon 1991], namely, inspection and adaptive control of robots (and other programmable machines), whereas the application domains of computer vision cover a wide variety of industries including food production, automobile, pharmaceutical, cosmetics and electronics [Gill 1990]. Furthermore, it was predicted by

[Groover et al 1986a] that the field of computer vision would be one of the fastest growing commercial areas in the remainder of the twentieth century: certainly since that time major advances have been forthcoming [Braggins 1990a].

To gain knowledge of existing and future potential applications of computer vision technology/systems in various sectors of industry, it is necessary to classify the vast variety of these application areas. However, this classification is neither exhaustive, nor intended to confine vision applications to certain categories.

A. Inspection

This is at present by far the most important and dominant application area of computer vision, accounting for at least 80% of all current applications [Vernon 1991]. Vision systems of this type are often given the name of automatic visual inspection (AVI) systems [Chin and Harlow 1982] [Foster et al 1990]. Examples can be found almost in every industry sector, such as inspection of food products [Davies 1984], verification of the presence of components in assembly [Groover et al 1986b], PCB inspection [Chin and Harlow 1982] [Landman 1988], dimensional measurement [Hollington 1984] [Foster et al 1990] [Vernon 1991], parts identification [Batchelor et al 1985], etc. In some cases, 2-D computer vision systems will suffice, whereas in other cases, 3-D vision will be demanded such as for the inspection of solder joints of PCBs, solder paste applications prior to component insertion.

B. Guidance

This is an application area where close interaction between computer vision systems and robots (or other programmable motion controllers, e.g. automatic guided vehicles-- AGV) are involved [Hollington 1984] [Groover et al 1986b]. The requirements for the computer vision systems used in guidance applications are quite different from those used for inspection [Pugh 1983] [Rummel 1989], and may necessitate more sophisticated design of the vision system. The vision system may

be required to identify and locate (i.e. to determine the position and orientation of) components on a conveyer or in a part bin, and feed back this information to a robot or machine controller so as to guide the handling of components (i.e. by a robot manipulator or motion systems, e.g. to provide AGV navigation information [Rajagopalan and Cheng 1991] [Kay and Luo 1991]). In many cases of guidance applications, real-time 3-D vision will be a requirement [Geng et al 1991].

An important and interesting area of research is to combine computer vision technology with that of robotics, thus referred to as robot vision [Pugh 1983], providing methods of evolving versatile, intelligent vision-guided robot system applications; for example vision guided intelligent robot assembly [Kohn et al 1983], robot arc-welding [Masaki et al 1983] [Clocksin et al 1983]. The coupling of a vision machine with a robot end-effector provides the opportunity of extending vision applications to include use in unfavourable environment or areas where mobility of the camera is required; for example visual controlled spray painting [Johnston 1983], automatic inspection and electronic gauging of car bodies [Macri and Calengor 1980].

In chapter 4 of this thesis, the author's work on integrating an AdeptOne robot manipulator and a Matrox vision system is presented. Part of the reason for conducting this study was to explore potential applications of vision guided robots to enable for PCB handling and inspection tasks (where the mobility of camera is utilized to enable the inspection of large PCBs at an appropriate measured resolution).

2.7 Applications of Computer Vision in Electronics Manufacturing Industry

In electronics manufacturing industry, as in many other manufacturing industries, product inspection is an important step in the process of production [Chin

1988], and it often represents dull and routine work for a person or persons. As a result, most of the applications of vision systems fall into the “inspection” category outlined in last section. Not surprisingly, the electronics industry is a very active one in applying automatic visual inspection (AVI) to products such as PCBs, integrated circuit chips, microcircuit photomasks, hybrid circuits, other electrical and electronic assemblies [Chin and Harlow 1982] [Chin 1988].

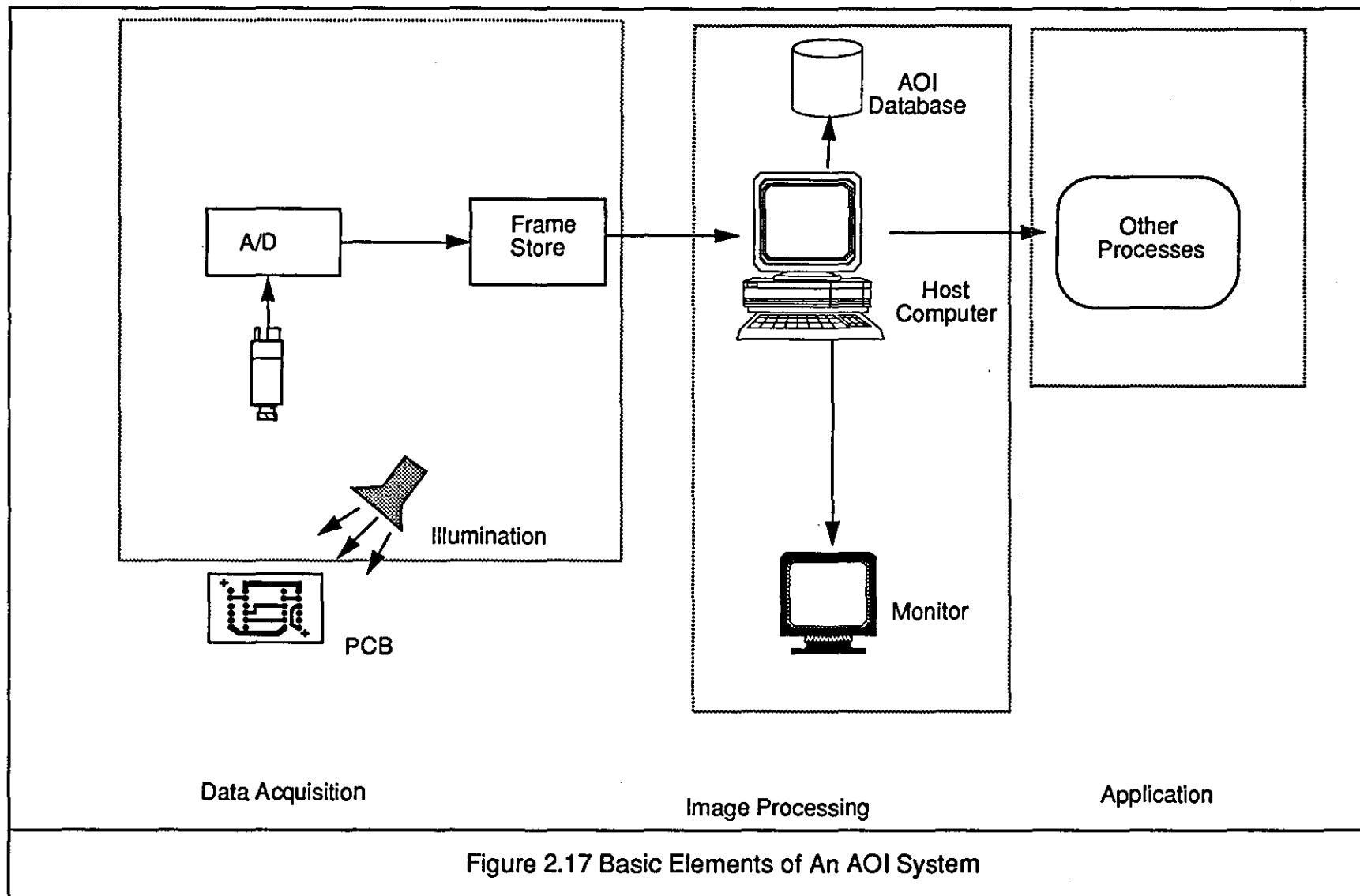
2.7.1 Automatic Optical Inspection

Automatic optical inspection (AOI) is a general name given to a class of vision systems employed by the electronics manufacturing industry for the purpose of automatic visual inspection of PCBs; including phototools, inner layers, bare boards, assembled boards and solder joints [Bartlett et al 1988], with bare board inspection being the most common applications of AOI [Keeler 1988]. In these applications, AOI systems are used as tools that maximize the benefits of visual inspection, while eliminating many of the shortcomings associated with human inspection [Gilutz 1988], with an attempt to isolate errors such as shorts, opens, missing holes, incorrect markings, over- and under-etchings so as to impose 100% quality assurance [Chin 1988].

A schematic diagram showing the building elements of a typical AOI system is given in Figure 2.17, which exhibits many of the structural characteristics of the more generalised computer vision system.

2.7.2 AOI Techniques and Their Applications in PCB Product Inspection

Much research activity, e.g. [Ejiri 1973] [Danielsson and Kruse 1979] [Bentley 1980] [Jarvis 1980] [Hara et al 1983] [West 1984] [Mandeville 1985] [Garakani and Cobb 1986] [Darwish and Jain 1988] [Hara et al 1988] [Ye and Danielsson 1988] [Benhabib et al 1990] [Lloyd 1990] [Sprague et al 1991], has aimed to address the problem of bare PCB inspection: or related problems in regard to say



the inspection of innerlayers of PCBs. Out of these activities, many techniques are proposed. Thus bare board inspection can be considered as one the most mature industrial applications of AOI systems [Chin 1988].

In almost all the techniques proposed, although they are seemingly different from each other, *a priori* knowledge about the objects to be inspected is utilized to provide strategies and criteria for the inspection process; for example knowledge about the dimensions and geometric structure as well as the engineering tolerance are frequently made use of. Furthermore, since PCBs before the insertion of components virtually have only two types of relatively flat surface (i.e. metal and substrate) binary image processing techniques are adopted in most cases [Chin 1988] with a view to reducing the amount of data which an AOI system has to process.

The techniques proposed by various research activities can be classified into two basic categories, namely reference based techniques and non-reference based techniques [Chin and Harlow 1982] [Silven 1984] [West 1984] [Keeler 1988] [Benhabib 1990] [Sprague et al 1991].

A. Reference Based Techniques

This is the method derived from the more general "template matching" techniques of object recognition (see section 2.6.2). Two subclasses of these techniques can be further distinguished, namely 1) image subtraction, which is a pixel-wise comparison methods whereby subtraction operation is performed between images of a reference board and a board under inspection; and 2) feature comparison, where high level features of the board being inspected are extracted and compared with corresponding features of the reference board. For example, Jarvis [Jarvis 1980] has used a list of (few hundreds of) 5 X 5 and/or 7 X 7 binary patterns which describe the normal conductor-substrate boundary derived from a perfect

PCB. This methods eliminates the need for precise alignment of the board under inspection, but requires careful generation of the binary patterns (templates).

B. Non-Reference Based Techniques

Non-reference based techniques can be further divided up into 1) design rule check [Chin 1988] [Landman 1988] and 2) morphological transformations [Iverson 1988].

With design rule check, features of the perfect board are represented by a set of design rules against which boards to be inspected are compared. Violations of these rules will be reported as potential defects. The most common use of design rule checks has been that for dimensional verifications of conductors and spacing requirements, see, for example, [Danielsson and Kruse 1979] and [Bentley 1980].

Morphology is the study of forms and shapes [Serra 1982]. Morphological image processing is a robust approach to image analysis. With this approach, morphological operations (transformations) are repeatedly applied on the entire image. This enhances details of interest while de-emphasizing unwanted details [Iverson 1988]. One of the first applications of morphological image processing to PCB inspection was described by Ejiri and is commonly known as the Hitachi algorithm [Ejiri 1973], typified by the simple operation of dilation-and-erosion, or expansion-and-contraction. Other applications of morphological transformations are reported by Mandeville [Mandeville 1985]. Lloyd described an analytical approach [Lloyd 1990], which is in fact based on the morphological approach.

Combinations of the reference and non-reference based methods seek to take advantage of their respective strengths and overcome their weakness [Chin 1988]. Example applications of the combined techniques have been reported in [Hara et al 1983] [West 1984].

The forgoing discussions have mainly concentrated on AOI applications in bare board inspection. Although these applications represent the main stream of AOI applications in PCB manufacture [Gilutz 1988], there are other interesting application areas; for example, in the inspection of solder joints. During solder joint inspection, properties of the surface geometry have been widely used as inspection criteria [Besl et al 1985] [Bartlett et al 1988] [Driels 1988]. Nakagawa [Nakagawa 1982] uses a structured lighting approach to obtain shape information. Compared with the surface geometry approach, this method offers the advantage of very fast processing. Capson and Eng developed a system for solder joint inspection by means of tiered illumination consisting of coloured ring lights [Capson and Eng 1988] coupled by connectivity analysis of red and blue image planes. Common soldering flaws such as no solder, insufficient/excess solder, poor wetting of component leads, etc. are reported as being detected and classified successfully. A drawback of the system is related to the complexity of processing colour images. A similar system but using monochrome lighting was reported in [Takagi et al 1991] using again a specially designed tiered illumination method. The system inspects solder joints according to the gradient of the soldered surface, which is determined by examining the intensity changes of the reflected light obtained through illuminating the soldered surface from different incident angles.

2.8 Summary

The first part of this chapter reviewed the primary application areas of computers in manufacturing industry and examines various methods and tools for achieving manufacturing integration. The needs for, and challenge and benefits of CIM have stimulated world-wide interest and research. Some of that research is classified and reviewed: for example in communication protocol standards, integration architectures, information administration systems, etc. Out of these research activi-

ties, *de facto* and international standards are emerging and facilitating moves towards the goals of systems integration.

The second part of this chapter reviews computer vision techniques and their application in manufacturing industries. Computer vision is another area in which significant research effort has been devoted during the last decade or so. The application of computer vision technology in manufacturing has been viewed as a means of achieving flexible automation. However, until recently computer vision applications have not been developed within a CIM context.

The review of automatic visual inspection, and more specifically automatic optical inspection (AOI) reveals that 1) although considerable research effort has been centred on developing suitable techniques for the inspection of PCB products, little has been done on integrating that inspection activity into its wider context, i.e. relationship with other PCB manufacturing processes; and 2) in fact, almost all the techniques developed or proposed have been focused on solving PCB inspection problems by adopting a standalone operating style, based on the traditional "golden board" approach (i.e. using a known perfect board -- referred to as "golden board" or "master board" -- for generating inspection criteria such as reference images, design rules, etc.). Any effort made on developing or implementing AOI systems using that approach will inevitably lead to the creation of an "island of information" (generated by "island of visual automation"). Whilst this approach could, to some extent, partially meet the requirements of PCB product inspection, ongoing trends towards integrated manufacturing systems have highlighted generally the limitations and drawbacks of stand-alone approaches [Powell and Carignan 1989].

The author believes that computer vision systems (including AOI systems) can be more efficiently and cost-effectively utilised in manufacturing industry if useful information residing at other "islands of computerisation" can be utilized to support their operation. Furthermore, better use of information generated by the

vision systems can be made by these other processes provided that appropriate means of sharing information resources can be enabled. Based on such a premise, this research aims to explore this possibility by (i) providing means to achieve an enhancement in the application of computer vision systems (in general), and (ii) illustrating and evaluating in proof of concept form the methods derived by targeting them at AOI systems (in particular).

Chapter 3

The Requirement for Integrated Machine Vision Application in PCB Manufacture

3.1 Introduction

Most electronic equipment/devices use printed circuit boards (PCBs) for purpose of interconnecting, both physically and electrically, the various electronic components used in the system [Kiko 1984]; from household electrical appliance (e.g. telephone, camera, television set, etc.), through office equipment (e.g. printer, fax machine), factory shop-floor systems (e.g. numerical controlled machines, programmable logic controller, robot, etc.), to military utilities such as radar and satellites. As modern society is becoming increasingly electronics-reliant in every respect, PCBs are becoming more and more vital to the operation of many aspects of the human society. Naturally, these electronic devices are required to operate with high reliability, which in turn imposes stringent requirements on the quality of the PCBs that work in the heart of all the devices.

To meet this challenge and requirement, automated and high precision equipment is being continuously introduced into the PCB manufacturing cycle where 100% product inspection is now required [Chin 1988]. This opens the door for introducing advanced automatic inspection/test systems so as to exercise tighter control of the product quality. With increasing computer-based automation of vari-

ous aspects of the PCB manufacturing industry [Weardern 1990] [Beyer 1990], electronic data is now increasingly used [Hansohn 1990] as inputs to drive various PCB production equipment (e.g. NC drill), or as information output of other processes (e.g. CAD systems, inspection/test stations, etc.). This offers the further possibility for manufacturing devices to treat information as a shared resource: it being easier to manipulate (i.e. store, transfer, reform, etc.) information stored in electronic form than using more conventional manual processing involving establishing written information recording and analysing methods. However, to turn the possibility into a reality is no easy task; this is the area of systems integration where much effort is being exercised.

This chapter presents a more detailed discussion about the use of automatic optical inspection (AOI) systems in PCB manufacture. Limitations associated with present turnkey systems will also be pointed out, highlighting the need for integrated applications of AOI systems. The discussion then leads to a way forward by making the most out of the available electronic information created and/or used by AOI systems. Much of this discussion will consider opportunities arising from using an information supported approach to AOI applications.

It is important to note that traditionally AOI has been mainly applied for the purpose of bare board (or layer) inspection; therefore the term "AOI" often means "AOI of bare board or layers" [Keeler 1988]. While bare board inspection is an important area of AOI application, other application areas are also of great significance. Thus in this thesis, the term "AOI" will be used to refer to the whole range of applications of machine vision systems (in the PCB manufacturing industry) for the purpose of PCB related inspection; including, for example, inspection of bare board, component placement, solder joint, etc.

3.2 A Conceptual Model of PCB Manufacturing Industry

In order to appreciate the potential role that AOI systems can play in the PCB industries, it is necessary to have a clear picture of PCB product realization processes. A conceptual model of these processes is therefore presented here with a view to advancing such an appreciation, and to identifying stages of PCB manufacturing where AOI systems can be utilized as a means of establishing product quality assurance.

Taking a broad view, processes involved in PCB manufacture can be conceptually divided into three distinct domains, namely design, manufacture and monitoring, as illustrated in Figure 3.1.

3.2.1 Design

This is where the design engineers express what they really expect the products to be, using media such as engineering drawings, computer graphics and/or computer documents (files, database). Such product specifications are made to align with marketing requirements and/or customer demands, as well as to ensure optimum cost, productivity and manufacturability[Kiko 1984].

In cases where computer aided design (CAD) facilities are available, the original product specifications are entered into a CAD system to enable the appropriate range of product design and/or design refinement processes to be carried out [PCAD 1989b]. On completion of these design processes, design data can be further processed and distributed to relevant processing stations for the purpose of PCB manufacture [Riley 1988]. Thus increasingly, design is viewed as an information producing function [Du Feu 1988]. The original design data is normally resident inside the CAD system, in the form of CAD files or in a database, and stored on various storage media [Hansohn 1990]. This design information is usually considered as describing a perfect PCB product (though one can not rule out the possibility of

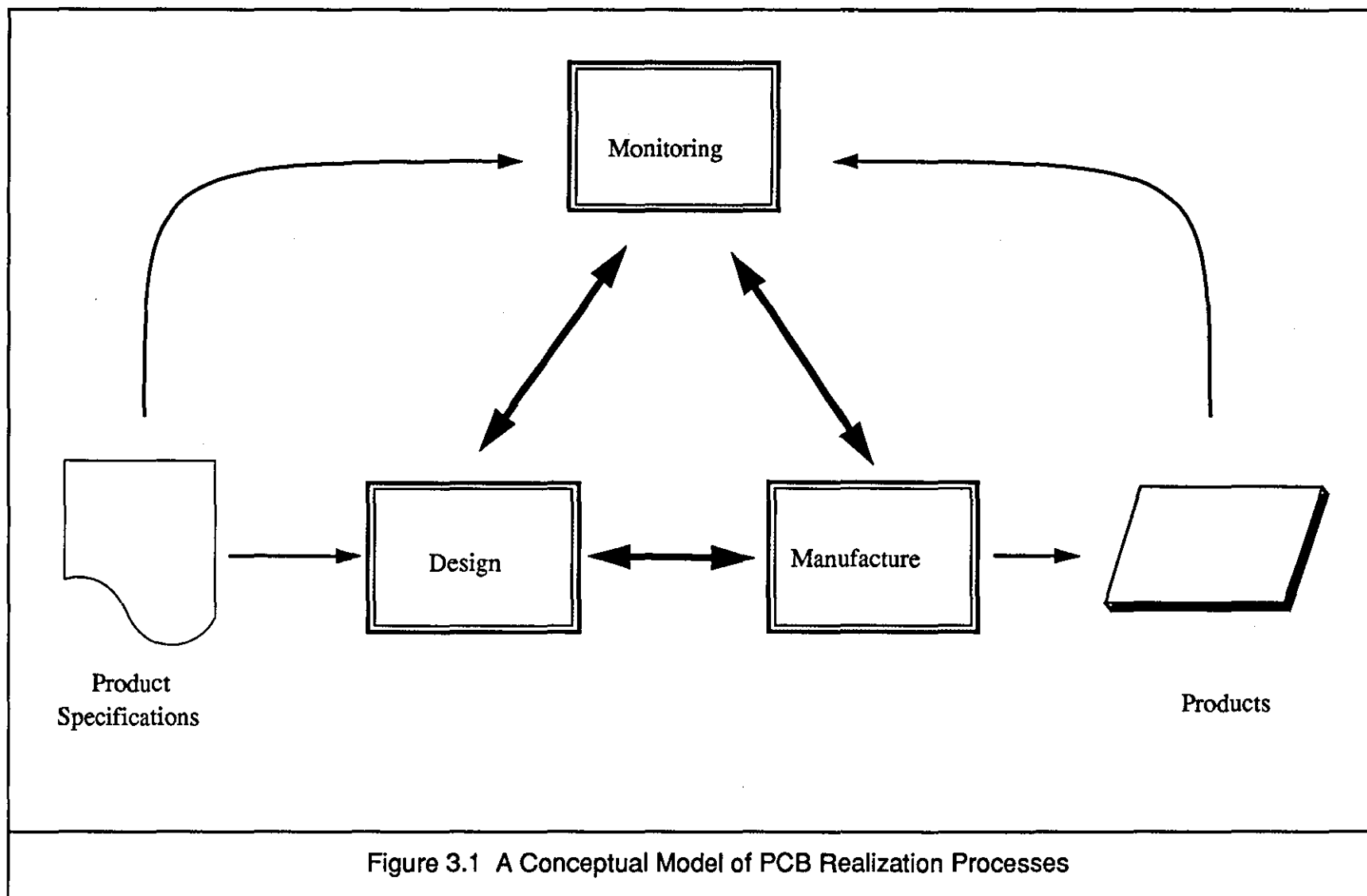


Figure 3.1 A Conceptual Model of PCB Realization Processes

errors being introduced by the design). All products subject to inspection should be checked against this design information (specifications), and products which do not live up to these specifications should be flagged as potentially defective. Usually such deviations in product quality is the consequence of distortions introduced by succeeding manufacture processes [Wright 1990] following product design.

3.2.2 Manufacture

Here a variety of manufacturing devices/processes function together to fulfil the goal of the whole PCB realisation cycle, that is to make products [Stoll 1990]. Although, over the past years, much advance has been made in PCB manufacturing technology, the elementary manufacturing processes generally remain the same [Farrimond 1988]. For example, typical processes for bare board manufacture include master film generation and phototooling, copper clad preparation, innerlayer manufacture (drilling, printing, exposure, etching, plating, bonding), bonding and outer layer manufacture (printing, exposure, drilling, etching, and plating) [Kiko 1984].

After bare boards are made satisfactorily, they reach the stage of PCB assembly whereby various components (needed to perform certain defined functions) are assembled on the base bare board, and the required interconnections between them are realized by means of the tracks, vias, pads, etc. built in the board and other additional conductive adhesives such as solder paste [Corey 1990] (followed by a soldering process to form solder joints to secure the interconnection). Thus for PCB assembly, typical processes will include component insertion/onsertion [Markstein 1988b], soldering, cleaning, packaging, etc.

In cases of using surface mounting technology [Mangin and McClelland 1987] [Buckley 1989] [Buckley 1990a], solder paste application will be required prior to component placement. Figure 3.2 illustrates those processes involved in

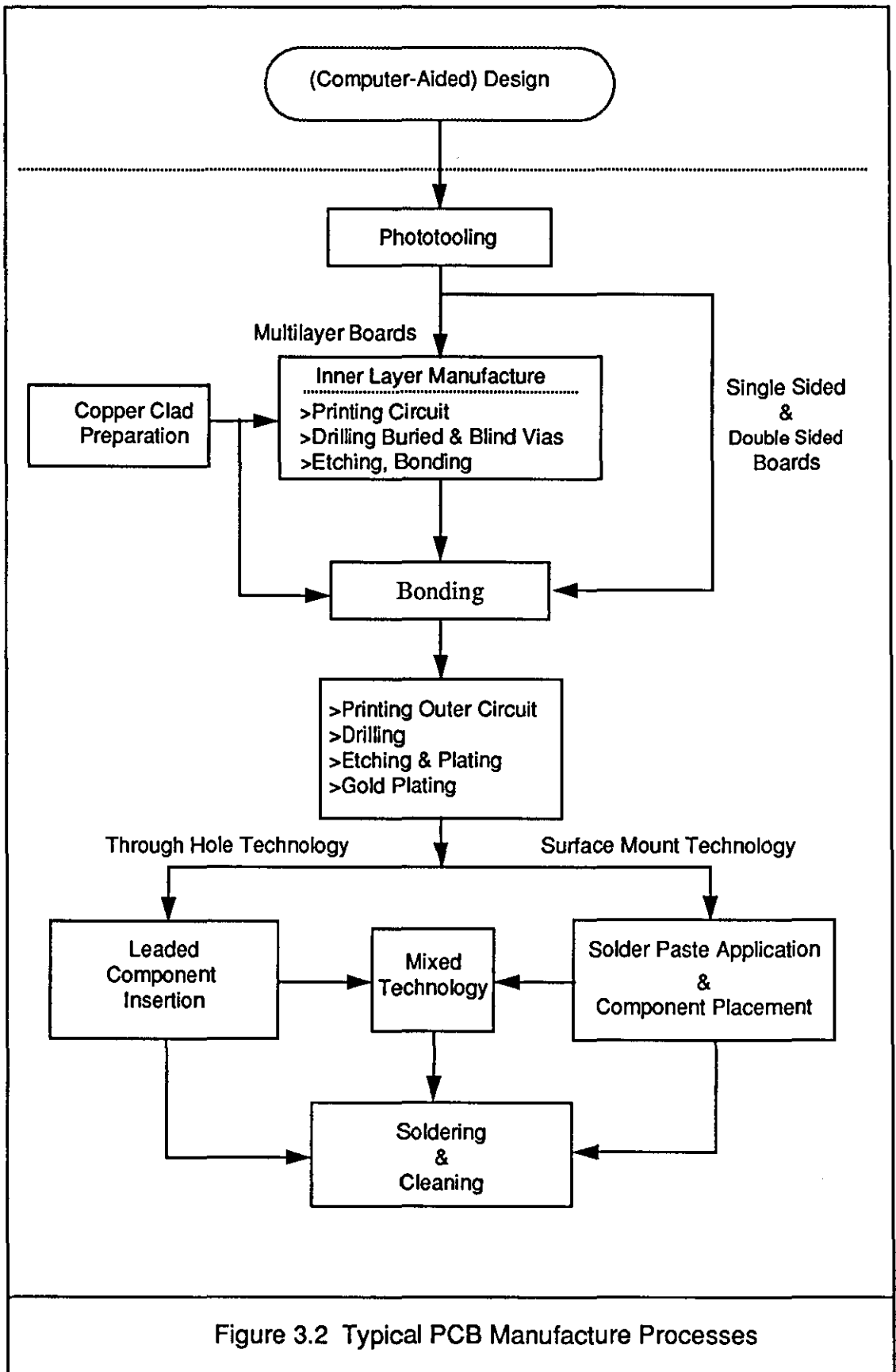


Figure 3.2 Typical PCB Manufacture Processes

PCB product manufacture, loosely based on [Kiko 1984] [Farrimond 1988] [ICL-Kidsgrove-1] [P-CAD 1988b] [Bauer and Alt 1990] [Hidde 1991].

3.2.3 Monitoring

Facilities and processes employed to monitor manufacturing activities have basically two broad classes of functionality, namely,

- 1) Machine-oriented real-time sensing and information feedback.
- 2) Management-oriented product inspection/test and information feedback.

Class 1) functionality is mainly for machine “performance monitoring” and thus is mostly applied at the device level for the purpose of real-time, on-line feedback control [Du Feu 1988]. This level of monitoring is necessary for enhancing and/or optimising machine performance, improving machine accuracy, precision, and yields. Sensors and transducers of various sophistication [Bailey 1989], often coupled with computer-based data processing capabilities, are used for data acquisition [Babb 1987]; various closed loop feedback control methods [Franklin et al 1986] are being adopted by machine builders.

Class 2) functionality is mainly for the provision of information support which can be utilised for the purpose of “quality monitoring/control” and process control; this becoming increasingly vital to the survival and success of any manufacturing industry [Challis 1989] [Stanton 1989]. Its importance in PCB manufacturing has become widely appreciated as well [Eckes 1990]. The application of this type of monitoring processes/devices are becoming imperative in safeguarding product quality and promoting yields. Examples can be found like automatic optical inspection (AOI), automatic testing equipment (ATE), infrared thermography equipment (ITE) [Dresser 1990], statistical process control (SPC) [Chen 1990] [Cowie 1990], human visual inspection, etc.

Whilst the realisation of the first of these functional classes is very much dependent on individual machine builders, the exploitation of the second one is indeed at the mercy of machine users, or, to be more exact, of the system integrators. In other words, the realisation of a level of “quality monitoring/control” will largely rely on computer integrated manufacturing (CIM) to provide appropriate information support, feedback control mechanisms and the infrastructure for a total quality management (TQM) [Ranky 1991].

3.3 Automatic Optical Inspection and Its Role in PCB Manufacturing

Since its introduction in early 1980s [Dolberg and Kovarsky 1989], AOI has proven imperative to the PCB manufacture industry, especially as far as product quality is concerned. An increasing trend in favour of employing AOI in PCB industry is the result of both technological development within the electronics manufacturing industry itself and of external marketing pressures; which have been described as “technological push and market-pull” [Munro and Noori 1988] [Edwards 1990]. The availability and wide use of new technologies like surface mount, fine line [Gurian 1990] and multilayer construction have led to board geometry and circuit feature size continuously being miniaturized. This in turn makes human visual inspection more and more difficult, ineffective and unreliable [Landman 1988]. On the other hand, with increasing functionality being built on boards of ever shrinking size, the cost of scrap of finished board (or even of a completed single layer of a board) has risen dramatically, thus justifying the investment in machine-based automatic inspection systems [Rieley 1990].

In response to the demands for precise inspection equipment in contemporary electronics industry, AOI systems have been developed. In fact AOI has now become an important enabling technology for the successful introduction of new PCB manufacture technologies such as SMT and fine line; surely, this has been reflected in the sheer amount of literature concerning AOI applications as reviewed

in Chapter 2; for example [Ejiri 1973] [Jarvis 1980] [Hara et al 1983] [Mandeville 1985] [Darwish and Jain 1988] [Hara et al 1988] [Ye and Danielsson 1988] [Benhabib et al 1990] [Lloyd 1990] [Doyle 1990] [Sprague et al 1991].

Appropriate applications of AOI systems at earlier stages of manufacture promise to make it possible to detect faulty layers before they are laminated with good ones, thereby avoiding volume production of scrap or intensive repair work and much reducing unwanted adding of monetary values to faulty PCBs [Eldan 1990]. Better still, since the inspection results are in the form of computer data files, they can be easily transferred and processed for other uses. For example, where available, information feedback from these AOI systems could be utilised more intelligently to assist process and product quality control, e.g. for the purpose of statistical process control (SPC) [Gilutz 1990] and statistical quality control (SQC) [Zwern 1990] [Lozano 1990]. This offers opportunity to monitor process trends, detect and correct process deviations before it is too late to do so.

Examples of AOI applications can be easily found in the area of PCB product inspection [Chin 1988] [ICL-Kidsgrave-1] [ICL-Kidsgrave-2] [Lloyd 1990]. A check-list of common AOI applications is given as follows, although the author does not claim this list to be exhaustive.

- a) phototooling inspection,
- b) innerlayer inspection,
- c) bare board inspection (BBI),
- d) solder paste application inspection,
- e) component placement/assembly inspection,
- f) solder joint inspection

3.3.1 A Comparison between AOI and ATE

Traditional Automated Test Equipment (ATE) has for some time found wide application in the field of electrical testing of PCB product, and has been considered as the preferred final step in the PCB fabrication process [King 1990]. For years, ATE has been satisfactorily applied in the electrical testing of bare boards. Usually these tests are accomplished by processing the electrical signals collected by ATE probes, arranged in a “bed-of-nails” [Mawby 1989] [Prince 1989] configuration. In another word, ATEs can be considered to “feel” the object under test by “touching and analysing”, thereby locating faults.

As such, ATE normally only produces “pass/fail” results from testing; the quality of test results will thus depend on the selection of proper pass-fail threshold of the measured resistance [Mawby 1989]. Therefore, it can detect genuine shorts and opens according to a preset resistance threshold. This implies that it can reveal whether an expected electrical connection is present, but it will not likely show how good this connection is.

As contrasted with ATE, AOI systems “recognise” objects (i.e. circuit features) by “seeing and analysing” and thus (within constraints) are potentially able to quantitatively reflect how good or bad the connection is [King 1990], allowing for the verification of the integrity of the circuit interconnections [EP Report 1989]. Moreover, it can also reveal such information as whether certain pre-specified conductor widths or spacing requirements have been violated; obviously such measurement tasks cannot be tackled by ATE.

Since AOI systems utilise a non-contact sensing methods, i.e. use optical sensors, it does no harm at all to the object under inspection, nor will it modify the operational characteristics of the object whilst it is tested. This is a very appealing advantage of AOI over ATE; making AOI systems capable of offering important advantages in modern PCB manufacturing industry, especially where new technologies such as SMT, fine line and multilayer technology are being applied

[Kaplan 1990]. The miniaturized, densely populated PCB layers can be too vulnerable to withstand test/inspection processes involving direct physical contact. The increase in board density (in measure of functionality per area board) also presents great difficulty and leads to increased cost in the building of test probes and the “bed-of-nails” toolings [Buckley 1990b], which are integral parts of an ATE.

Nonetheless, as quality assurance techniques, both AOI and ATE have played important roles in PCB manufacture. The author believes that each of them has particular importance to PCB industry in its own right, and that any of them is not likely to be totally replaceable by another. For example, the functionality test performed by ATE is unlikely achievable by AOI systems. The roles of ATE and AOI, and the applications of these two technologies in PCB industry would be complementary rather than mutually-replaceable.

The dual goals of these quality assurance technologies can be summarised as

- 1) to prevent defective products from leaving the factory, and
- 2) to prevent defective products from being produced for example by means of optimum process control, taking advantage of the feedback information from monitoring equipment.

Obviously, the second goal is much more attractive as it implies substantial further reduction in unit product cost by reducing (or, to its extreme, eliminating) scrap. However, this goal is also much more difficult to achieve than the first.

3.4 Present Generation AOI

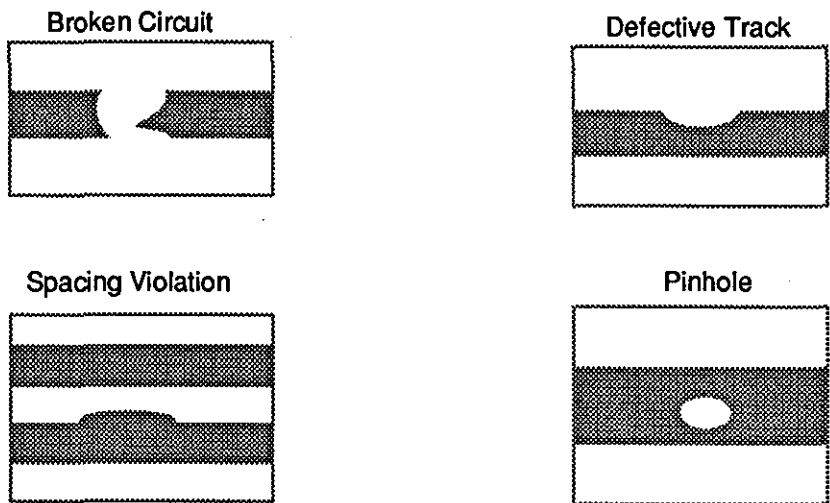
AOI system has been around for almost one decade. During this period of time, great developments have been made in improving their performance [Landman 1988] [Eldan 1990]. An example of prominent development is that of solid

state electronic cameras which gained rapid advances in production around early 1980s. By the middle of 1980s, solid state cameras became readily available on the market, leading to uses in laboratory research and industry applications [Braggins 1989], with “charge coupled devices” (CCD) based cameras being representatives of the class. Major advantages of CCD cameras over traditional thermionic cameras have been frequently quoted as stemming from their reduced size and power consumption, essentially indefinite lifetime, and greater resistance to physical shock [Purll 1985].

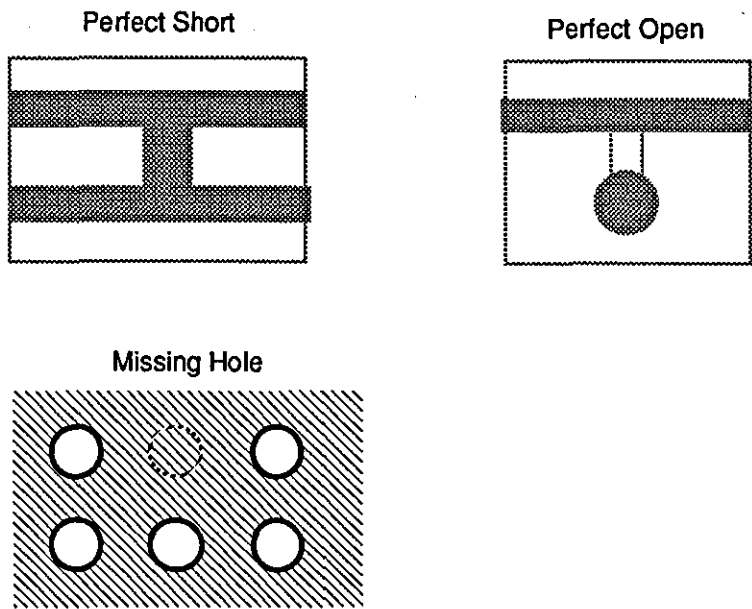
On the other hand, being one branch of computer vision, AOI systems have also benefited in many ways from parallel developments in the domains of computer vision (in terms of image processing hardware and software), pattern recognition and artificial intelligence. General and special purpose, functionally powerful machine vision products (e.g. off-the-shelf boards, turnkey systems, software packages, etc.) based on PCs and workstations have since become available from a variety of sources [Willis 1989] [Pfeiffer 1990] [Hospod 1990] [Howard 1990] [Mueller and Verrecchia 1990] [Braggins 1990b].

The many approaches to image processing and understanding commonly adopted in designing an AOI system have been reviewed in Chapter 2, and classified as falling into one of the following two categories, namely 1) the reference based approach and 2) the non-reference based approach.

Generally, non-reference based techniques allow the detection of such defects as broken circuits, annular ring, track shrinkage, spurious metals, pinholes, and pad size violations, as shown in Figure 3.3.a., whereas reference based methods allow the detection of typical defects such as missing circuits and perfect shorts [Powell and Carignan 1989], as shown in Figure 3.3.b.



(a) Design Rule Violation Examples



(b) Example Defects Detected by IR Approach

Figure 3.3 Example Defects Found Using DR and IR Approaches
[Powell and Carignan 1989]

3.4.1 Limitations of Present Generation AOI systems

As present generation AOI systems are designed to operate in a stand-alone manner, and with little accessibility to relevant design information resident inside other CAD/CAM stations, AOI systems have to rely on knowledge of an actual known perfect board (often referred to as the “master board”, or the “golden board”) to learn the board design and to extract the design rules or reference images. Thus two problems are presented to the PCB manufacturer, which they have yet to solve, namely 1) where to find the perfect master boards and 2) how to teach the AOI system the required properties of the master boards. Worse still, in the real world of PCB manufacture, this perfect master board hardly exists. Even with the most strict quality control, it is unlikely that it will be possible to guarantee that such a board is really defect free. By implication, there is a risk associated with the use of so called perfect boards. For example, if defects exist in the master board and without being detected, then all similar defects present in the PCB products will escape.

The dependency of inspection on a master board has long represented a major problem area of present generation AOI systems. Not only because real defect-free boards seldom exist, but also as the associated teaching of the AOI system is time-consuming and open to human errors introduced during the teaching process. As a result, the setting up of the AOI system can present major difficulties. Generally speaking the process of AOI system setup involves the steps listed as follows [Dolberg and Kovarsky 1989],

- 1) Defining areas to be inspected and eliminating irrelevant areas such as text (nomenclature), non-functional patterns, etc.; and
- 2) Establishing inspection parameters and criteria e.g. extraction of design rules and/or generation of reference images.

As dictated by market requirements, PCB manufacturers are required to be able to produce many small runs of boards, consisting of many types in a short turn-around [Bauer and Alt 1990]. This implies that, on one hand, quick system setup becomes more important than ever, and on the other hand, less human involvement is demanded so as to eliminate human error as completely as possible. The conventional methods of achieving system setup can then be challenged.

3.5 CAD Reference for AOI

Whilst PCB manufacturers are seeking the perfect master board, such defect free “board” often resides within CAD/CAM workstations, i.e. within CAD/CAM files describing the perfect product. This is the data representing the real intentions of PCB designer and the product demanded by the market and customers; such theoretical models of PCBs are by definition zero-defective. Thus the use of CAD data to support AOI of PCBs seems to offer a right way forward [Powell and Carignan 1989] [Esposito 1989] [Dolberg and Kovarsky 1989] [Rittichier 1989] [Doyle 1990]. Shown in Figure 3.4 is a comparison between the conventional approach (master board dependent approach) and the CAD information support approach to PCB inspection.

While this idea of CAD data reference can be traced back to 1985 [Wright 1990], the actual introduction of CAD data for inspection started in 1988 [Powell and Carignan 1989] (this incidentally co-inciding the start of this research project). Because of the inherent defect-free nature of the CAD data, this approach promises better solutions to PCB inspection problems than those offered by the conventional master board dependent methods.

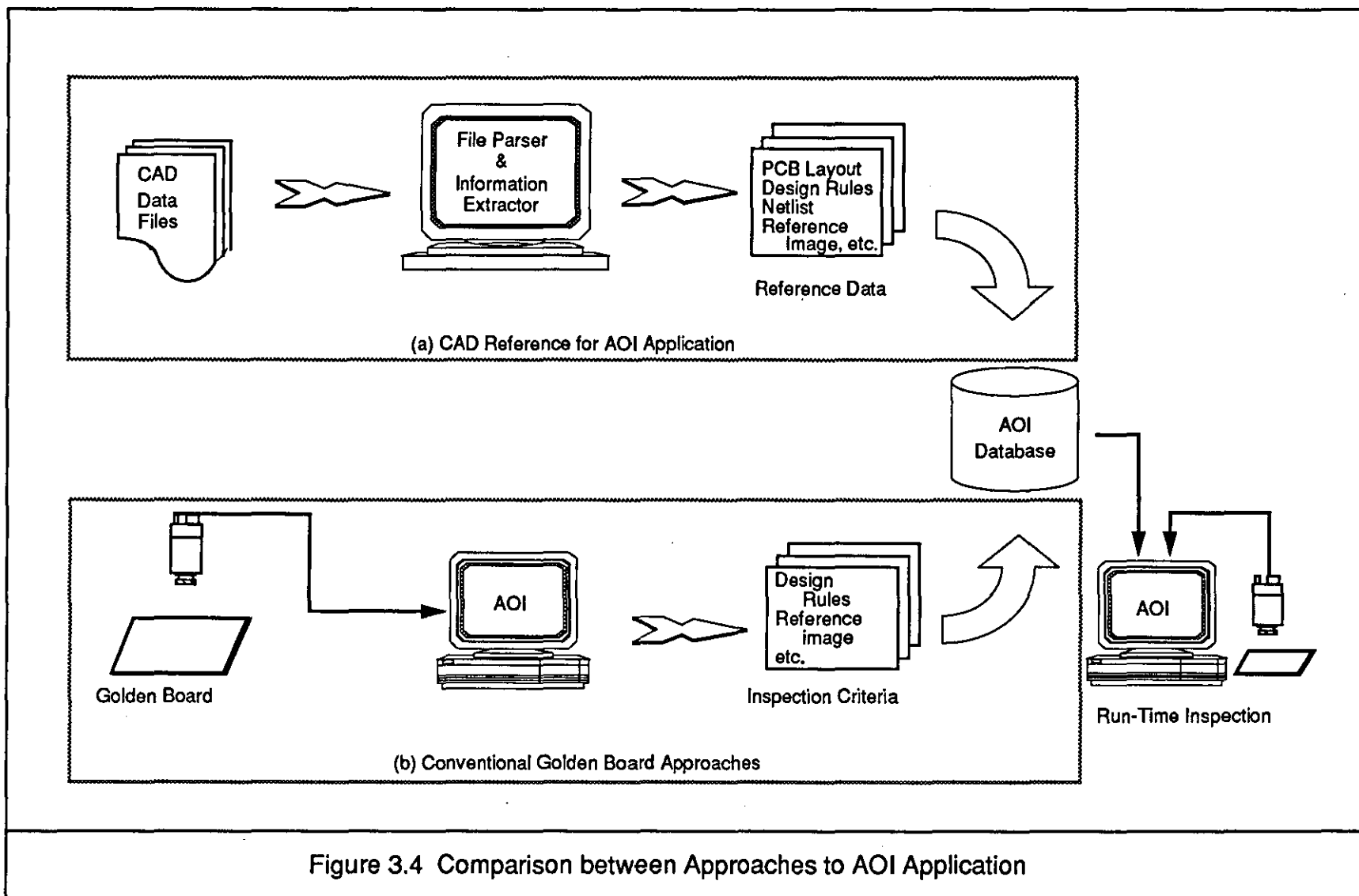


Figure 3.4 Comparison between Approaches to AOI Application

3.5.1 Benefits of CAD Data Reference

Thus using CAD reference data for AOI applications is a method of re-utilising the design information during PCB product inspection processes. With this approach, all products subject to inspection are compared with the design information; differences will be flagged as (potential) defects. Several immediate benefits arise from this approach:

- 1) Since the CAD reference data is by nature defect free, there should be no risks whatsoever regarding the quality of the reference data. In other words, no such hidden errors (undetected flaws) as might be found in a master board exist in the CAD generated reference data.
- 2) “Re-utilization of the design information” means that the original product specifications need to be entered (or created) only once at the stage of computer aided product design; therefore more effective and efficient use of the existing data can be made by multiple applications.
- 3) By establishing links with CAD systems, AOI systems will have direct access to the design data. The reference images, design rules and even netlists (representing electrical connectivity maps) can be generated and/or extracted from the CAD database automatically either on-line or off-line. This can greatly improve the performance of AOI system in terms of system setup, inspection reliability and system flexibility and reconfigurability [Dolberg 1989] to accommodate product changeover.
- 4) The ability to achieve effective reference to the CAD data will help maintain the AOI database well updated and more comprehensive.

3.5.2 Problems Associated with CAD Reference

To make effective use of the CAD information, one needs to be aware of the inherent problems and possible difficulties associated with this approach.

Firstly, proprietary data format and communication protocols are still prevalently adopted by individual CAD/CAM and AOI system vendors. The absence of standards in both communication and information representation have greatly hampered the effective information flow and restricted information sharing amongst all relevant devices. As a result, it has been difficult for AOI systems to make reference to appropriate CAD/CAM information. This problem has to be tackled before effective CAD reference can be made for AOI applications.

Secondly, current AOI systems themselves are not as yet readily designed with an open architecture to facilitate multilateral communication with other systems. Nor have they been designed with a suitable internal architecture and equipped with appropriate software for the purpose of making effective use of imported CAD information. The introduction of CAD information into the inspection processes of AOI equipment demands that new and suitable methodologies be adopted in the design of such AOI systems.

Thirdly, it is unlikely that real world products can totally match the CAD models or specifications. This is due to the fact that all manufacturing processes generate distortions to the work-in-progress (WIP). These process-induced distortions (PIDs) are all contributing towards producing defective products, yet not all these distortions will warrant the product being labelled defective. However, these distortions do make the product deviate from its specifications. If the AOI system is not designed with due consideration for accommodating such minor PIDs, it will be unusable as it will generate false alarms at high rate. In view of this situation, it is important the AOI systems which make reference to CAD information at the same time take into account other attributes of process variables.

3.6 Information Support for AOI

It is suggested by the author and by others [Powell and Carignan 1989] [Wright 1990], that comparing the board under inspection with CAD information could lead to significant improvements in PCB product inspection. However, simply connecting CAD stations to AOI systems, or even making the CAD data totally accessible to AOI applications, is not sufficient to take full advantage of the CAD reference data. Nor can the full potential of AOI applications be attained by taking such a simplistic approach. More information is also required to support the AOI applications, which can place detected faults in context.

For example, process information which describes allowable tolerances (i.e. acceptable PIDs) must be incorporated into the process of product inspection to avoid large numbers of false alarms. If information about all processes prior to inspection can be obtained and made available to AOI systems, the false alarm rate could be expected at its lowest level. With trends towards the computerization of more processes (such as in the drill shop, assembly line, etc.), increasingly there will be possibilities of modelling these processes and providing suitable information feedback or feed-forward to relevant processes such as AOI system, at least from the view point of their statistical performance characteristics such as that of acceptable distortions made to the work-in-progress.

Other examples of such information support can also be found, such as information about board material and about illumination equipment, which could be useful to the process of image segmentation; information about board transportation and positioning apparatus, which could help locate board or sections of the board where certain board features are expected; information about the surface features of solder joints generated by certain soldering processes, which could be useful for the purpose of solder joint inspection.

Therefore, information support for AOI is the key to the realisation of the fully automatic inspection of PCB products (Figure 3.5). Such a level of information support can only be achieved in a computer integrated manufacturing environment and only then can the full potentials of AOI applications be gained.

3.6.1 Specifications of a New Generation AOI system

An advanced AOI system with the built-in capability of two way communication and information sharing with other systems should have the following features (obviously the first two features are standard features of any AOI systems, be they stand alone or integrated with other systems; they are included here for completeness),

- An ability to interface with physical equipment used to collect raw image data from a real scene. Typically this equipment would be a CCD line or area camera for present AOI. Also a properly selected illumination method is essential.
- A comprehensive range of image processing, analysis and understanding software and/or hardware that permit the inspection of a variety of PCB features with fast and accurate performance as well as friendly user interface.
- An ability to interface with CAD/CAM workstations to input and/or make reference to design information (design rules, netlists, etc.) and to process information (e.g. to gain knowledge of process variables); that is, being able to operate in an integrated environment consisting of a wide range of PCB product realization facilities, e.g. by including "interfacing" algorithms in AOI software routine library.
- An ability to maximise the benefits of being information sup-

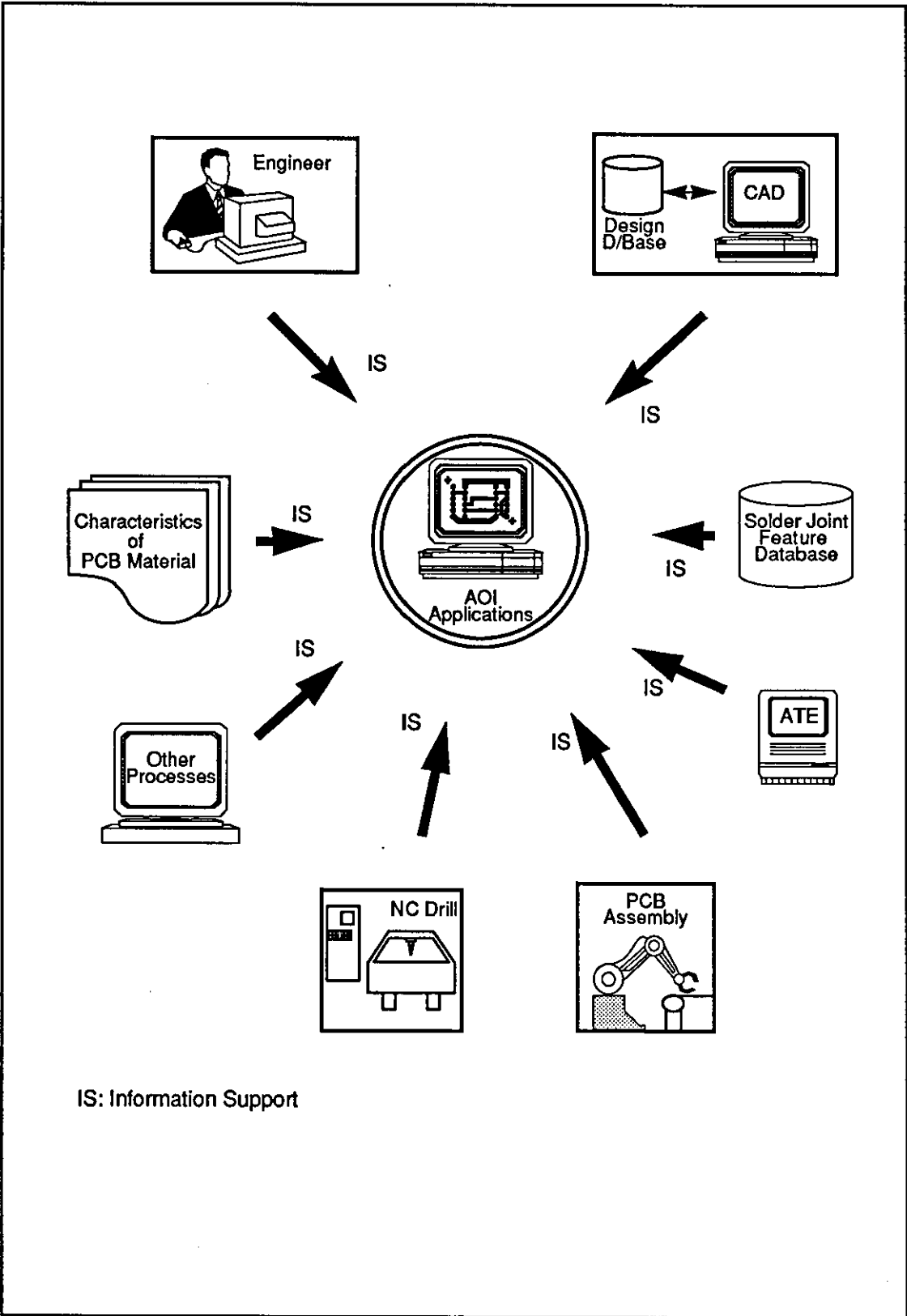


Figure 3.5 Information Support for AOI Applications

ported, e.g. by means of image processing and interpretation algorithms that can make best use of available information.

- Provision of an effective human interface and an ability to make use of operator input/modified parameters during inspection processes, e.g by including graphical user interface (GUI) facility, etc.
- An ability to report the inspection result in formats which not only assist engineering understanding and product rework, but also to facilitate information reference by other systems (e.g. process control system), and to update product models for later process utilisation, e.g. by including a set of application specific routines which can be used present the inspection result in various forms to cater for differing applications.
- Can support standard data format and communication protocols and fit within emerging open architecture frameworks to facilitate information sharing and integration, e.g. by adopting standard communication protocol, neutral (ideally standard) data structures and information representational formats (e.g. in representing inspection information).

3.7 Summary

This chapter has discussed a particular application area of industrial machine vision systems, namely, the application of AOI systems in the PCB manufacturing process. Comparison between AOI and ATE has been given to help understand the role of AOI systems in PCB manufacturing, as well as the complementary nature of these two technologies.

Also briefly discussed are the approaches adopted and inherent limitations of current generation AOI systems which highlight the need for AOI systems to make use of PCB product models and relevant manufacturing process information so as to achieve information supported AOI of PCB products; therefore stressing the requirement for integrated applications of industrial machine vision systems in the PCB manufacturing industry. This discussion provides a context for the author's research study into the flexible integration of machine vision systems within the realm of computer integrated manufacture.

The problems associated with adopting a CAD reference approach have been discussed, leading to an identification of the following questions which have to be answered to allow solutions to be evolved.

- 1) Given the reality that heterogeneous systems coexist and proprietary data formats proliferate, how can CAD PCB modelling information be made accessible to the AOI systems.
- 2) Assuming that CAD reference data is readily available to the AOI system, how should it, and in what way can it, make use of this information.
- 3) In view of the fact that manufacturing processes will lead to product distortions, how can information supported AOI system handle minor deviations and separate these from fault conditions which need to be detected.

However, the author believes that solutions to the above mentioned problems can be of vital importance to PCB manufacturers. Figure 3.5, which is a subset of the more general information supported PCB manufacture, offers a schematic framework for this research study.

Chapter 4

The Integration of Machine Vision and Robotic Systems

4.1 Introduction

As discussed in the previous chapter, there exists a need for the integrated application of AOI (automatic optical inspection) systems in the PCB (printed circuit board) manufacturing industry. However, the practice of integrating AOI systems in PCB manufacture is a complicated one; having special requirements relating to the complex nature of machine vision as an example of automated equipment, and necessitating many special studies and investigations through hand-on practices. Thus in order to gain knowledge about and an understanding of the problems involved, a study was conducted of methods of integrating the activities of machine vision systems with those of robotic systems. This study is presented here which is one of the two system integration problems addressed by the author. Particularly, this study aimed

- 1) To advance the author's knowledge in regard to machine vision and robotics,
- 2) To categorise existing and develop new approaches to establishing integration between such manufacturing entities, and
- 3) To establish pre-cursor guidelines for integrating and using vision guided robot in PCB product realisation.

The two proprietary systems involved in this study are of a heterogeneous nature. They are from different vendors, designed for totally different applications, and programmed and run under dissimilar computer operating environments. Hence the integration problems faced are representative of those commonly found in manufacturing industries.

4.2 Description of the Facilities Used

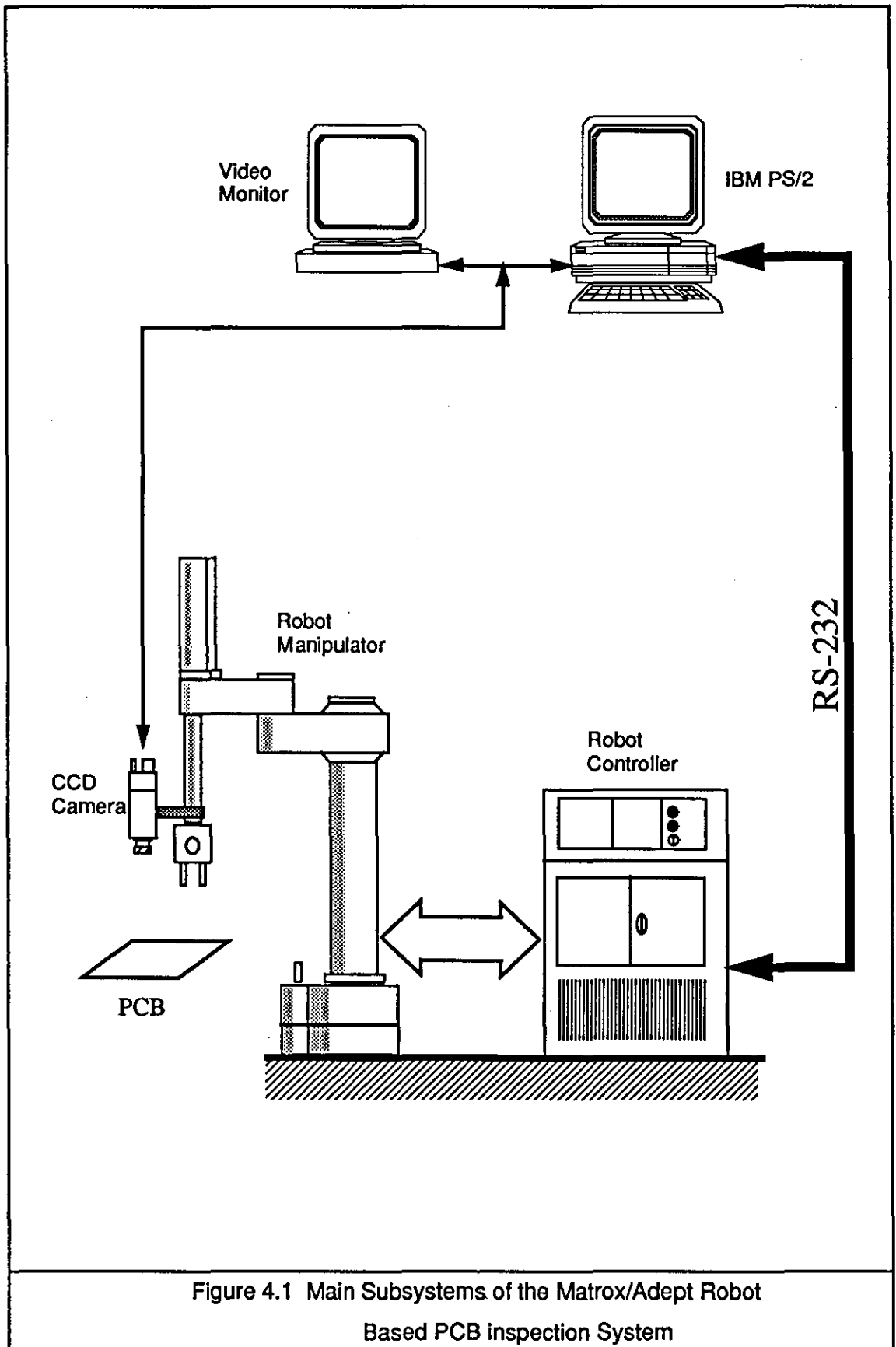
A Matrox machine vision system [Matrox 1988] and an AdeptOne robot manipulator system [Adept 1985] formed the essential two components (which will be referred to as “manufacturing devices”) of the integrated PCB inspection system. Figure 4.1 illustrates this first phase integration scheme, while the following sections briefly introduce its main features.

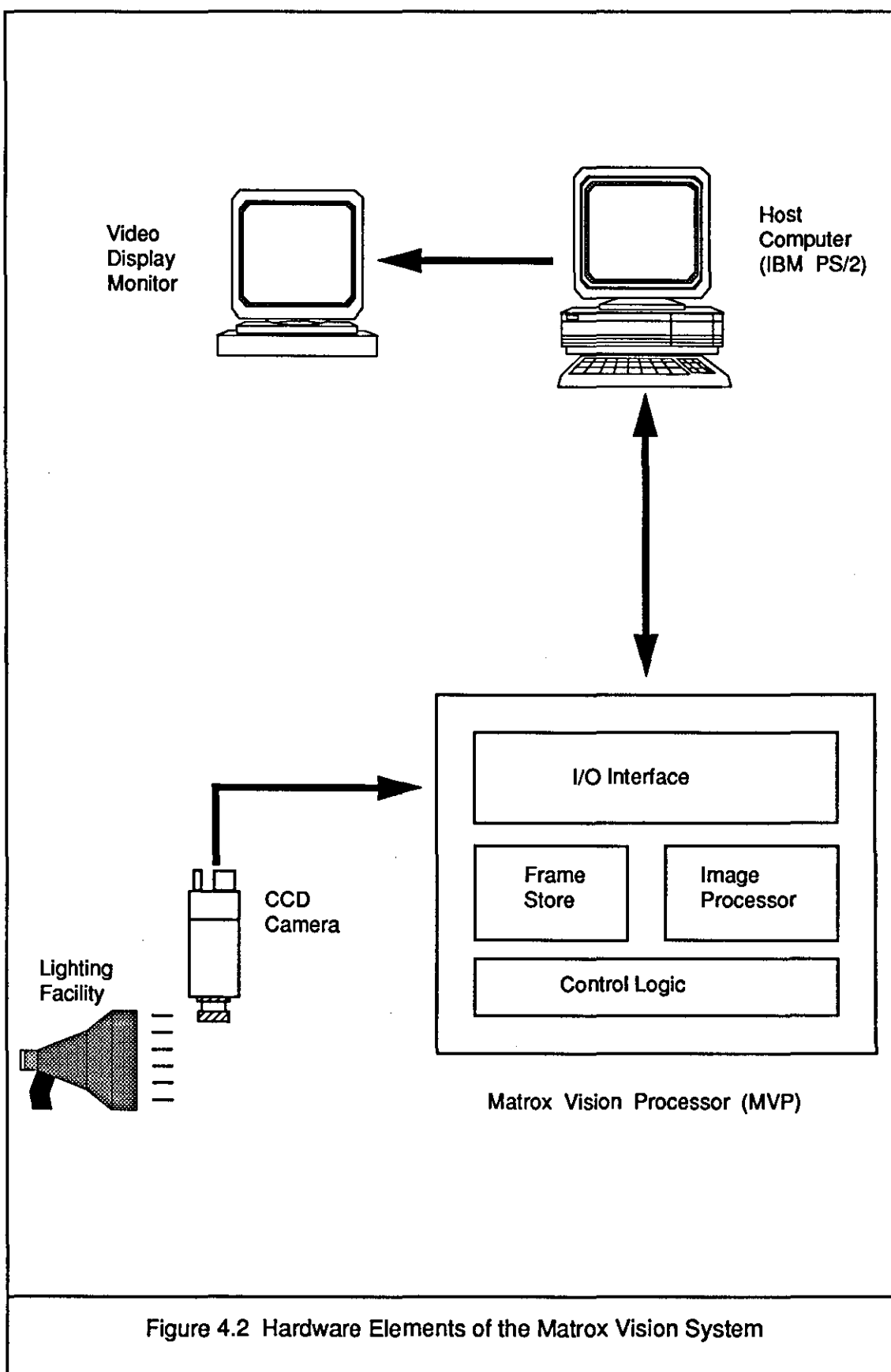
4.2.1 The Matrox Machine Vision System (MVS)

This is the machine vision system used throughout the author’s research project. It comprises five hardware sections as follows:

- 1) A Matrox vision processor (MVP-AT),
- 2) An IBM PS/2 host computer,
- 3) A CCD solid state camera,
- 4) A monochrome analog monitor for the display of grey scale images, and
- 5) An illumination system.

The arrangement of the hardware components used and the links between them are illustrated in Figure 4.2.





The MVP-AT vision processor is a board level product supplied by the Matrox Electronics Systems Ltd. It comprises three circuit boards which use the internal PC-bus of any IBM compatible personal computer. In this project the host machine used is an IBM PS/2 personal computer system which runs the image processing software but also bridges the gap between this MVP board product and its users, be they human operators or any other computerised equipment.

The image processing software supplied by Matrox is a library of low level image processing routines written in the C language. The functional capabilities of the routines can be classified as follows

- a) Configuration and control functions (e.g. to perform initialization, operating mode selection, video monitor display format selection, zooming in/out, etc.),
- b) Image I/O and display control routines (e.g. snapshot, storing images to/from disks, etc.),
- c) Routines which achieve image processing via neighbourhood operators (e.g. Sobel, Kirsch, Prewitt, Laplacian edge detectors, image sharpening and averaging, etc.), and
- d) Statistical analysis of processed images (e.g. intensity histogram and profiles, maximum/minimum intensity values, etc.)

The library routines can be linked to user application programs written in the Microsoft C code [Matrox 1988]. The combined 'target' software can then be run on the host computer, under the Microsoft C environment.

4.2.2 The AdeptOne Robot Manipulator System

The AdeptOne robot system essentially comprises two main hardware elements, namely the robot manipulator and its control system. It is designed mainly

for assembly applications of light components, with a maximum payload reaching six kilograms [Adept 1985], and has been used relatively extensively in industry for electrical component placement and assembly. This particular robot manipulator used has four degrees of freedom, to which a fifth can be added as an option. It has four joints arranged in a SCARA (Selective Compliance Assembly Robot Arm) configuration [Groover et al 1986b] (see Figure 4.1).

The control system for the manipulator comprises three 68000-based computers. VAL-II (the upgraded Victor's Assembly Language) [Gruver et al 1984] is the high level, Basic-like language used by the system for the programming and control of the robot movement. As such, the information representations used within the robot system must conform to the syntax and semantics of the VAL-II language, using a proprietary data format, albeit that VAL and VAL-II syntax semantics are widely used.

4.3 Vision-Robot Communication

4.3.1 Overview

Whilst the use of the C language is natural within the Matrox vision system, in the Adept robot system the use of VAL-II is a prerequisite. Therefore, any interaction between these two systems (involving communication between them) requires the use of some form of language 'translator'. This translator should fulfil the following three main tasks, namely 1) establish physical link for electronic data interchange, 2) provide control of message flow over the physical link, and 3) format messages (i.e. meaningful information transfer) which can be understood by both manufacturing devices.

4.3.2 Specific Constraints

Since the two systems (robot and vision) to be integrated are of a heterogeneous nature (i.e. supplied by different companies, operating under different environments, programmed using different languages, etc.), the characteristics of each party need to be examined as the first step of the integration study. Some of these characteristics could be the specific constraints governing the choice and establishment of communication interface between the vision and robot entities used in the project.

The control system of the AdeptOne used in this research study has five general purpose serial ports each of which can be configured by the user to establish RS-232 based data communication with other systems. However, apart from this low level data link, there were no suitable higher level communication protocols available with the system (either of a standard or indeed proprietary nature) which would enable the required integration processes.

Similarly, the Matrox vision system, as supplied in a stand-alone form, has no external communication capability beyond the general purpose serial and parallel ports (data links) supplied as part of the host computer. Any communications with other devices have to go through the host PS/2 computer. Thus

- Direct (and potentially higher speed) back plane communication between the Matrox vision processor and external devices could not be established without the unknown proprietary knowledge of the vision processor, and
- Once again no suitable high level messaging communication protocol was made available by the equipment vendor.

4.3.3 Description of the Approach Adopted

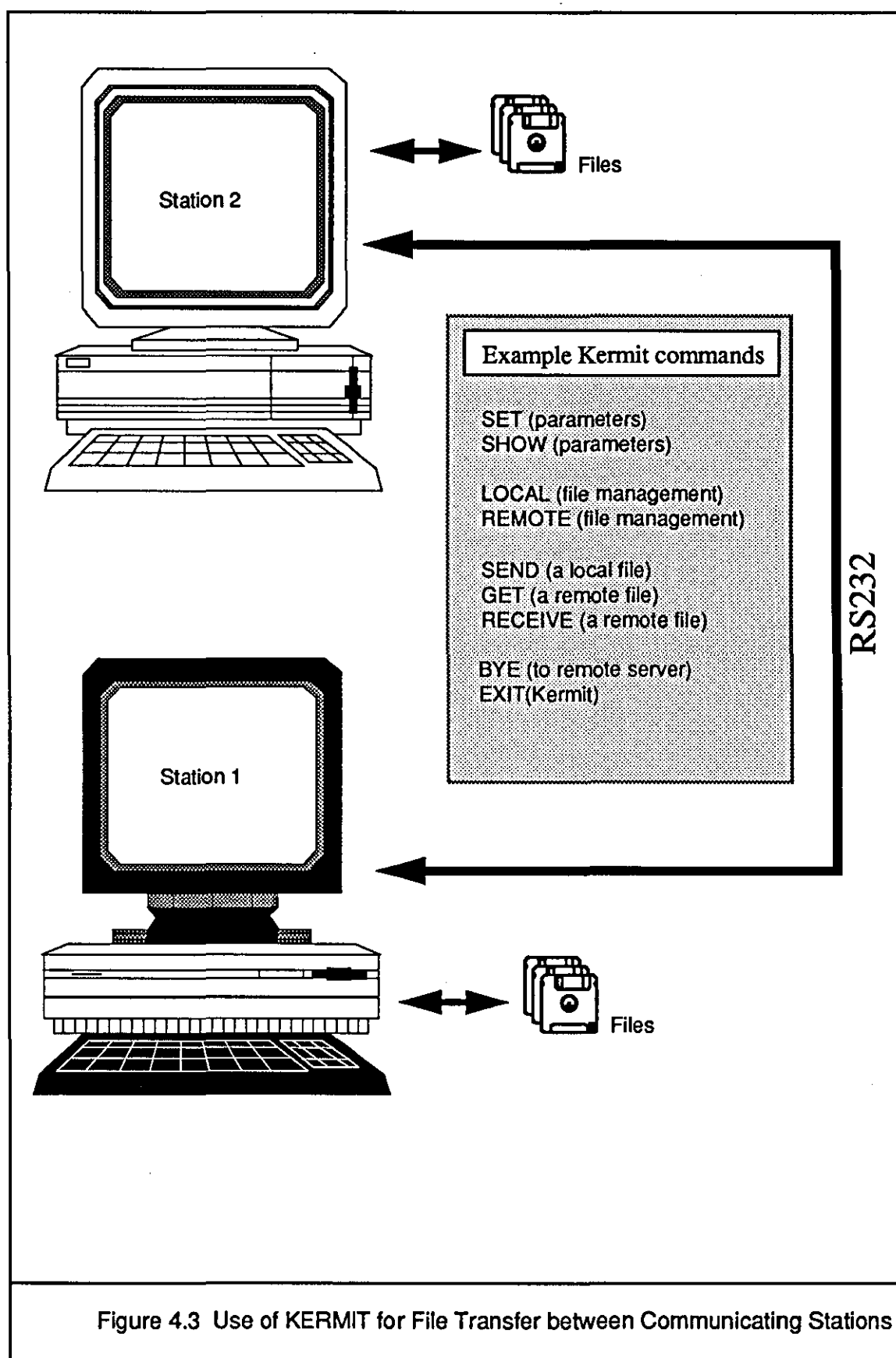
Recognising the constraints outlined in the last section, the author tried to seek assistance from some existing communication tools. Initially, the MS-DOS KERMIT protocol [Cruz and Damens 1985] was considered as a likely candidate for the transfer of files between these two communicating systems. This is because the KERMIT was the only available tool to the author at that time. Figure 4.3 depicts the use of this approach where KERMIT itself only provides a file transfer facility for data rather than a messaging system.

However, later investigation revealed that although the MS-DOS KERMIT has been widely implemented and used on PCs, its use with Adept machines has only recently been initiated by its vendors and did not (as expected) become a fully vendor supported option for any of the AdeptOne control systems available to the author.

It was not appropriate for the author to spend significant time implementing KERMIT communication protocol on Adept machines, or indeed to wait for it to be implemented by the vendor. Instead, another approach was adopted, namely to specify and implement a set of communications protocol to meet the specific demands of interaction between these two systems. Although the protocols so created are not sophisticated, nor conform to any international standard, they provide sufficient functionality to fulfil the communication tasks involved in this integration activity. Furthermore, their realisation provided an important learning opportunity for the author, providing a basis of knowledge on which more general observation could be made.

An assessment of the interaction requirements determined that the communication protocol should perform the following basic tasks:

- 1) To parameterise physical characteristics of the interface between the communicating manufacturing devices. Example characteris-



tics are physical port address, baud rate, bit pattern, parity, etc.

- 2) To achieve message control between the devices. For example, this protocol is required to generate handshake signals which initiate or conclude tasks,
- 3) To encode/decode messages according to a defined convention. For example this protocol facilitates the formatting of data to form messages.
- 4) To interpret the received information.

Down to the physical level of the communications, the messages carrying the required information is transferred via the RS-232 serial interface, as dictated by the conditions of the systems involved. (see Figure 4.1)

4.3.4 The Information Requirements of the Two Systems

The definition of the required information to be exchanged between these two systems is essentially application dependent. In this case, the author examined the information requirements of the following two major application areas of such an integrated system,

- 1) Inspection of bare printed circuit boards (bare-PCB).
- 2) Vision-guided robot for PCB assembly.

The information requirements determined as a result of this study are described in the following sections.

A. PCB Inspection

For the automated visual inspection of bare-PCB, the inspecting camera can be mounted on the robot arm (or indeed an alternative approach might be to

manipulate the board), and thus it can be moved around to inspect the whole PC board. This 'mobility' of the camera is necessary if higher resolution inspection is on demand and/or in cases where the board is over-sized.

In such an inspection scenario, it is required that movement of the robot manipulator (and thus of the camera) be controlled by the Matrox machine vision system (or more generally, by the automatic visual inspection system) and that the actual position of the robot end effector be fed back to the Matrox vision system. The spatial relationship between the camera and the robot arm should be calibrated beforehand. With the calibration data and the location information of the robot end effector, the vision system will be able to determine the spatial positions and orientation of the camera, and hence more accurately to locate the important features in the captured images.

In this application of the integrated vision-robot couple, the minimum information interchange required can be outlined as follows,

- a) Camera-robot calibration data, normally resident in the robot control system.
- b) Robot movement control commands, normally initiated from the MVS.
- c) Feedback from the robot to the vision system of actual position and orientation of the robot end-effector.
- d) Status of robot and its control system.
- e) Status of the vision system.

B. Vision Guidance for Robot

The flexible assembly of components into aggregated subcomponents or final products has been one of the major growing application areas of industrial

robots [Mangin 1988]. By equipping robot manipulators with vision sensing facilities in the application of robots in assembly, greater operational flexibility can result. Thus change can be accommodated leading to opportunities for smaller batch working and improved quality of products [Heginbotham et al 1983].

While this study of vision-robot integration has focused on its application to PCB inspection, it is clear from the use of other robot/vision coupled systems that the same concepts can apply in PCB assembly (for instance, component placement, insertion and onsertion) and indeed in more general flexible assembly applications. In fact, with this notion in mind, examination has been carried out on the information requirements of PCB assembly applications.

In PCB assembly, one of the most common uses of machine vision is to locate the actual footprint of components during placement operations. Following this, the disparity between the actual value and its nominal one (which may be derived from geometric model of the PCB) is calculated. This calculation can allow the required offset to be evaluated and fed to the robot so that it can correct its destination position. In such an example, the information feedback from the vision system to the AdeptOne robot forms the main stream of the information flow. The corresponding exchange of information can be categorised as following,

From AdeptOne robot to the vision system:

- a) Camera-robot calibration data.
- b) Current position of the robot arm.
- c) Status of the robot and its control system.

From the vision system to AdeptOne robot:

- d) Offset of measured locations from their nominal values.

- e) Status of the MVS.

4.3.5 Information Representation Format (IRF)

Interaction between the vision system and the AdeptOne robot has thus been realized through using a standard RS-232 interface over which data packets are transferred according to higher level customised interaction rules (or protocols). This serves to fulfil the first two tasks specified for the previously mentioned translator, that is to enable and to control the transmission of information in the form of messages (see section 4.3.1). The third task is concerned with the formatting of messages (i.e. the rules by which information is represented) and the way of interpreting the exchanged information by an end system.

Here again a specific approach was adopted to enable customised operation but also to highlight the underlying generic principle. Having taken into account the characteristics of both the vision and robot controller system, the author chose appropriate representation formats and the rules of interpreting the received information. This was achieved by first specifying the required information (as done in section 4.3.4) and then engineering specific software to run at each end. Obviously, agreement on information formatting and interpretation must be established to allow application processes to access the information at either end at a later time.

Shown in Figure 4.4 are examples of the information representations chosen, together with the format of corresponding messages. It should be pointed out, however, that only for the application of PCB inspection where “interaction services” of this type are fully created and used.

With reference to the information requirement specified in section 4.3.4(A) for PCB inspection, and to information representations introduced above in this section, Figure 4.5 illustrates some of the likely interactions (and corresponding message formats) between the robot and the vision system. Communication software

General Format:

_START	COMMAND	DATA	_END
--------	---------	------	------

Example Commands:

Commands	Meaning
_COMMS	Initiates communication links
_VRCALIB	Gets vision-robot calibration data
_STATUS	Enquires about robot's current status
_MOVE	Drives robot to a new position defined using absolute coordinate frame
_SHIFT	Drives robot by the offset, i.e. relative position
_WHERE	Requests current position of the robot
_HERE	Sends position data to vision system

Example information representations:

Commands	Data Set
_COMMS	port, baud rate, bit pattern, parity...
_VRCALIB	Xc, Yc, Zc, yc, pc, rc
_STATUS	calibrated/non-calibrated/ execution /...
_MOVE	X, Y, Z, y, p, r
_SHIFT	Xoff, Yoff, Zoff, yoff, poff, roff
_WHERE	void
_HERE	X, Y, Z, y, p, r

Figure 4.4 Examples of Implemented Commands and Data Set

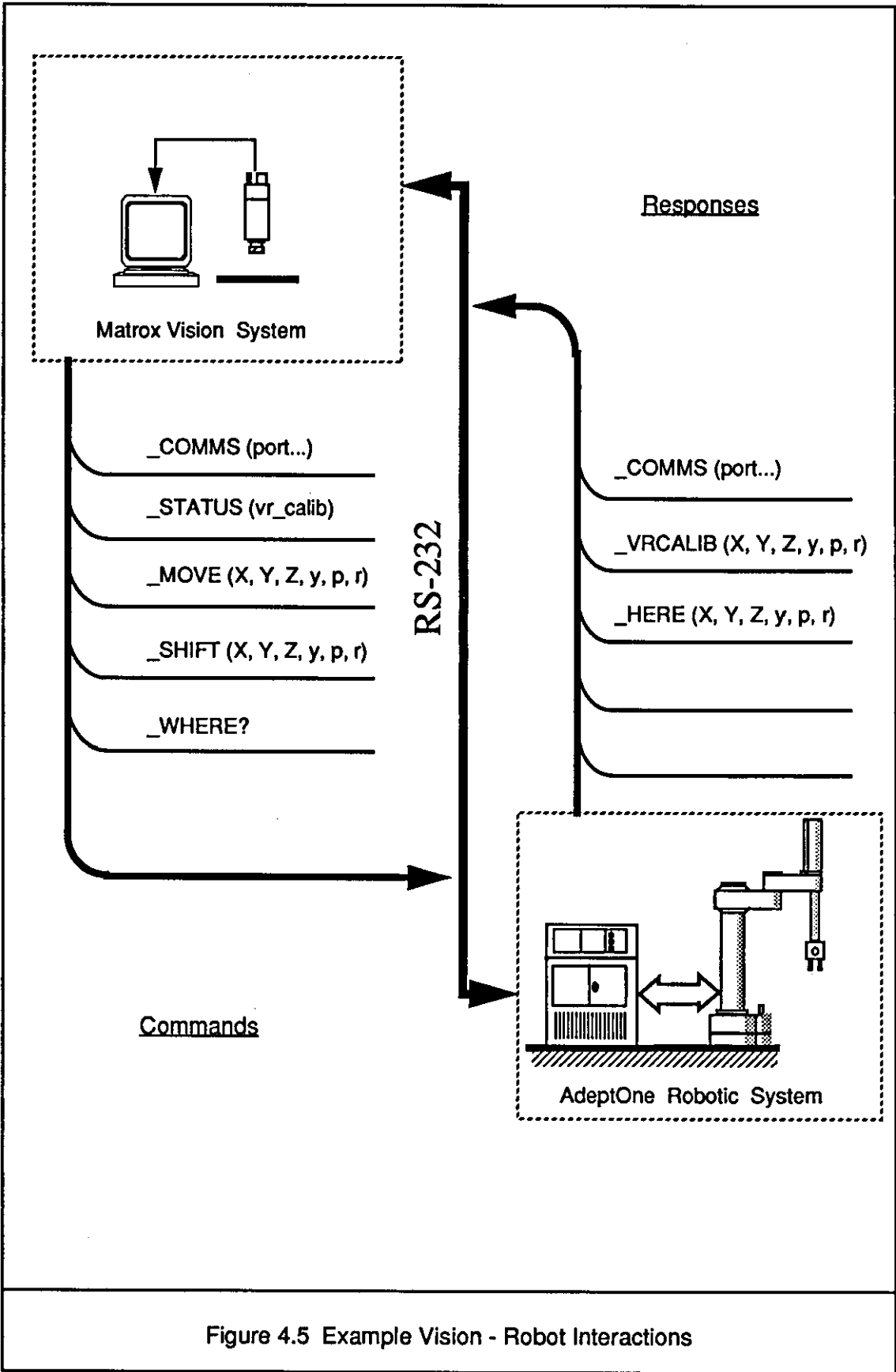


Figure 4.5 Example Vision - Robot Interactions

was designed and implemented successfully for this integrated vision/robot system to achieve interactions a) through e) as specified in section 4.3.4(A). Descriptions of the software used at either end can be found in section 8.5 of Chapter 8.

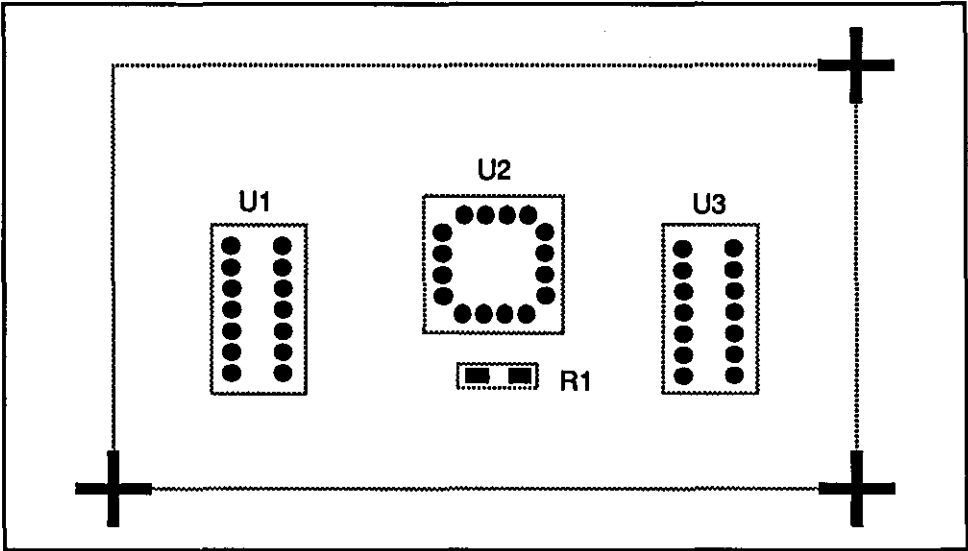
4.4 An Example Application of the Integrated System

To demonstrate typical interactions and information exchange between the two communicating systems, the next few paragraphs will describe an example application of the integrated vision-robot system. This is clearly a subtask of the more generalised task 'PCB inspection'. The description is also intended to demonstrate how interaction between the robot and vision systems can be achieved and how the inspection task can be accomplished more flexibly by using an integrated robot/vision couple.

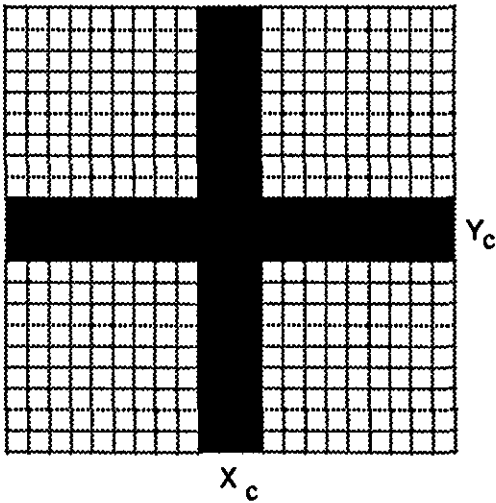
The requirement here is to locate the fiducial marks (usually two or three marks) on a bare printed circuit board (Figure 4.6), so that any mis-registration in this artwork can be corrected for and thus allow representation of the information entities describing the board to be expressed relative to a new board frame of reference. Commonly, mis-registration errors of this type result from any positioning errors caused by PCB transportation, handling and/or fixture equipment. Relaxing demands on high-precision positioning tools and fixtures can lead to significant improvements in flexibility, reduced cost and improved cycle times and is just one of the benefits promised by the application of machine vision in robot assembly systems.

In this application, the following conditions and/or constraints are assumed,

- a) The camera-to-robot relationship (spatial) has been calibrated, the mounting frame is maintained at a fixed position relative to the robot arm between calibration operations, and the calibration data



(a) An Example PCB Layer with Three Fiducial Marks



(b) Image of a Fiducial Mark

Fiducial Mark	Centroid
Mark #1	(X_{c1}, Y_{c1})
Mark #2	(X_{c2}, Y_{c2})
Mark #3	(X_{c3}, Y_{c3})

(c) Located Fiducial Marks

Figure 4.6 Inspecting and Locating PCB Fiducial Marks Using a Vision System

is stored in the robot control system.

- b) Satisfactory illumination conditions have been setup, and the Z-component (height) of the robot location (the latitude of the imaging plane) is such that the artwork is in focus.
- c) The PCB is positioned with coarse precision, and the nominal location of those fiducial marks are known to the vision system (with respect to a certain frame of reference).
- d) Image processing algorithms are available which have been designed to identify and locate the fiducial marks on the PCB.

Before the MVS can be put into use, the vertical and horizontal scale factors have to be calculated. This procedure is usually referred to as calibrating the vision system, or CCD camera. In this research activity this is done using a routine called **VISCAL** generated by the author. (A more detailed discussion about this routine is given in chapter 8). The *a priori* knowledge about the nominal positions of the fiducial marks can be established either by teaching the vision system or through downloading the information from a CAD/CAM system (this second possibility also being considered further in later sections of this thesis).

Having knowledge of these nominal positions, the vision system then can remotely control movement of the robot so as to drive the camera to desired locations at which images of the fiducial marks can be captured for subsequent analysis. This is accomplished through sending control commands via the vision-robot communication services. The actual locations reached by the robot are fed back to the vision system via the same communication facility. The status of the robot and its control system can also be established via the issue of appropriate commands by the vision system. After the vision system has identified and located the fiducial marks, the “actual” board frame of reference can then be used in future image processing tasks.

4.5 Discussions

Given any pair of dissimilar computerised manufacturing devices, dedicated programs can be designed to enable information exchange between them. Under certain circumstances, this type of approach to “integration” can be very effective. Since software is specifically designed and the communication facility is dedicated, resultant information exchange should be accomplished at higher data rate than might be the case using a general purpose interaction (or communication) service.

However, the ‘flexibility’ of this resultant ‘hard-integrated’ system is likely to be very limited, resulting from the use of non-standard (custom designed) protocol and the need for the communicating devices to have intimate knowledge of each other. When system requirements change, the existing software and hardware interface may no longer be applicable. For example a major system change may occur when one of the communicating devices is to be replaced by a more advanced yet incompatible one; here the integration services could no longer be at all appropriate and may have to be re-engineered at significant cost both financially and in increased lead time in system availability.

Many other types of system change can lead to similar problems. As an inherent theme of this research study relates to methods of building integrated systems, which inevitably will demonstrate changing requirements, the need for “soft” (or flexible) integration will be considered in various sections of this thesis.

The drawbacks of a dedicated approach to integration can become prominent when system expansion is taken into account. This can be better demonstrated by considering more complex product realization systems where many homogeneous/heterogeneous devices co-exist. If every communicating device is to be linked using the above-mentioned pair-wise integration, the number of dedicated ‘translators’ could soon reach an unmanageable level.

For example, within a group of n devices, if every device is to communicate with all other devices, and if the communication is based on the pair-wise integration approach, then $n(n-1)/2$ translators will be required. To emphasise further the scale of this problem, Table 4.1 lists the number of translators required for different numbers of devices.

Nonetheless, the successful integration of these two systems is valuable in its own right. Although the approach is dedicated, the problems revealed by this practice are of a general nature. In addition, the knowledge gained by this integration of vision-robot couple was utilised in the author's later research activities. It also represents an integral component of a wider prototype scheme demonstrating potential advances in PCB realisation as described elsewhere in this thesis.

Table 4.1 Number of Translators Needed

Number of Devices	Number of Translators
2	1
3	3
5	10
10	45
15	105
20	190
30	435
n	$n(n-1)/2$

Referring back to the ISO/OSI reference model reviewed in Chapter 2 (see Figure 2.5), we can see that the communication protocols designed here serve to perform the functions specified for layers 1, 2, and layers 6, 7 in the reference model. This is illustrated in the following:

Layer 7 (application): A specially defined set of application services (e.g. “_MOVE”, “_WHERE”, “_STATUS”, etc.) to enable the interaction between the vision system and the robot.

Layer 6 (presentation): A set of customised rules governing the representation of information that enables meaningful message transfers between the two systems, i.e. using mutually agreed representation formats, as illustrated in Figure 4.4.

Layer 2 (data link): A set of handshaking signals to control and manage the establishment, maintenance and termination of the data link between the two systems.

Layer 1 (physical): Physical link is realized using standard RS232 with parameters such as serial port number, bit pattern, parity check, baud rate, etc. being configurable by the user at initiating the communication links.

For example, when a specific application such as the example given in section 4.4 needs to communicate with the robot, it uses the services supplied by the “application layer” to send over the command message to the robot; these being the control of the movement and position of the robot, or enquiries about the status or current position of the robot arm. When a command such as “_MOVE” is issued, the command message is formatted according the rules specified by the “representation layer”, the communications link is established and controlled by the handshaking signals defined by the “data link layer” and the data stream is transferred over the physical link of RS232 with user programmed bit pattern, baud rate, etc. Note that, in this scenario, the robot system acts passively, i.e. makes a movement only upon receiving a command message from the vision system, otherwise it remains where it is.

4.6 Summary

An integrated vision-robot system has been introduced in this chapter. The approach to integration adopted here is a highly dedicated one in the sense that the communication protocols (including rules governing the formatting of the transferred information) are specifically designed for the two systems being integrated. As discussed in the last section, the protocols serve to accomplish the functions specified for layers 1,2, 6 and 7 of the ISO/OSI reference model. An example application of the integrated system is given in this chapter. Above all, the integration practice introduced here has helped the author to understand inherent problems when integrating machine vision system with robot manipulation.

Chapter 5

Using CAD Information to Support AOI Operations

5.1 Introduction

Many requirements, which determine the successful and widespread application of AOI systems at various stages of PCB production processes [Zwern 1990], are essentially related to the demands of producing today's high density PCBs at low cost and high quality with short product lead time. However, in reality, present generation turnkey AOI systems have not been so designed and marketed. Thus their potential has not been fully exploited. This is due to the following facts (as discussed in Chapter 2 and Chapter 3):

- 1) Contemporary AOI systems are typically designed to operate in a stand-alone manner, often with a closed-architecture. This has restricted both their ability to access information resident at other "islands of computerisation" and their ability to produce AOI-generated information for use by other sub-systems. This in turn has greatly restricted the realisation of AOI based process characterisation and control [LeBeau 1991].
- 2) Contemporary AOI systems rely on the use of master board method to achieve product inspection, thus the quality of performance suffers from the fact that real defect-free master boards are

not normally available in the real world of PCB manufacturing [Dolberg and Kovarsky 1989].

On appreciating the limitations of present AOI systems for a few years, now users in electronic industries have recognised the need to reference CAD/CAM information to support AOI applications [Powell and Carignan 1989] [Rittichier 1989] [Lloyd 1990].

Thus, this chapter presents a study of issues concerning the use of CAD/CAM information to support the automatic optical inspection of PCBs. This study was realised in a practical sense by achieving the extraction of useful design information from a proprietary CAD system, and the re-utilisation of such information within the Matrox machine vision system previously described. Subsequently, this study formed the basis for generalisation and practical extension into a proof of concept PCB product model driven AOI system.

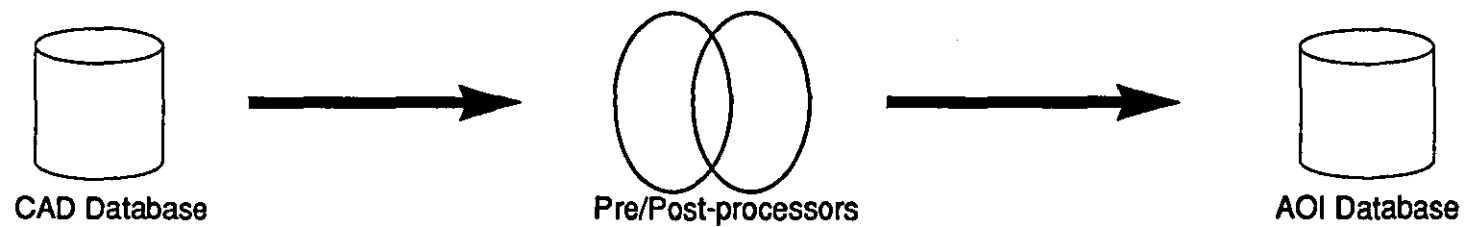
5.2 Systems/Tools Utilized

To facilitate this segment of the study, the following systems/tools are used, namely, a P-CAD (Personal CAD) system [P-CAD 1989a], a Matrox machine vision system [Matrox 1988], and a SUN 3/60 workstation (Figure 5.1). The general functions of the elements of Figure 5.1 are described as follows:

The P-CAD system is used to design and define the schematic, artwork, layout, component and manufacturing representations relating to a circuit board. The output file of the P-CAD database is processed (i.e. is parsed and interpreted) within the SUN workstation. Also within the SUN workstation the information extracted from the P-CAD system is re-formatted into a suitable format for direct use by the vision system so as to support its performing of a given inspection task.



a) Hardware Arrangement



b) Functional Illustration

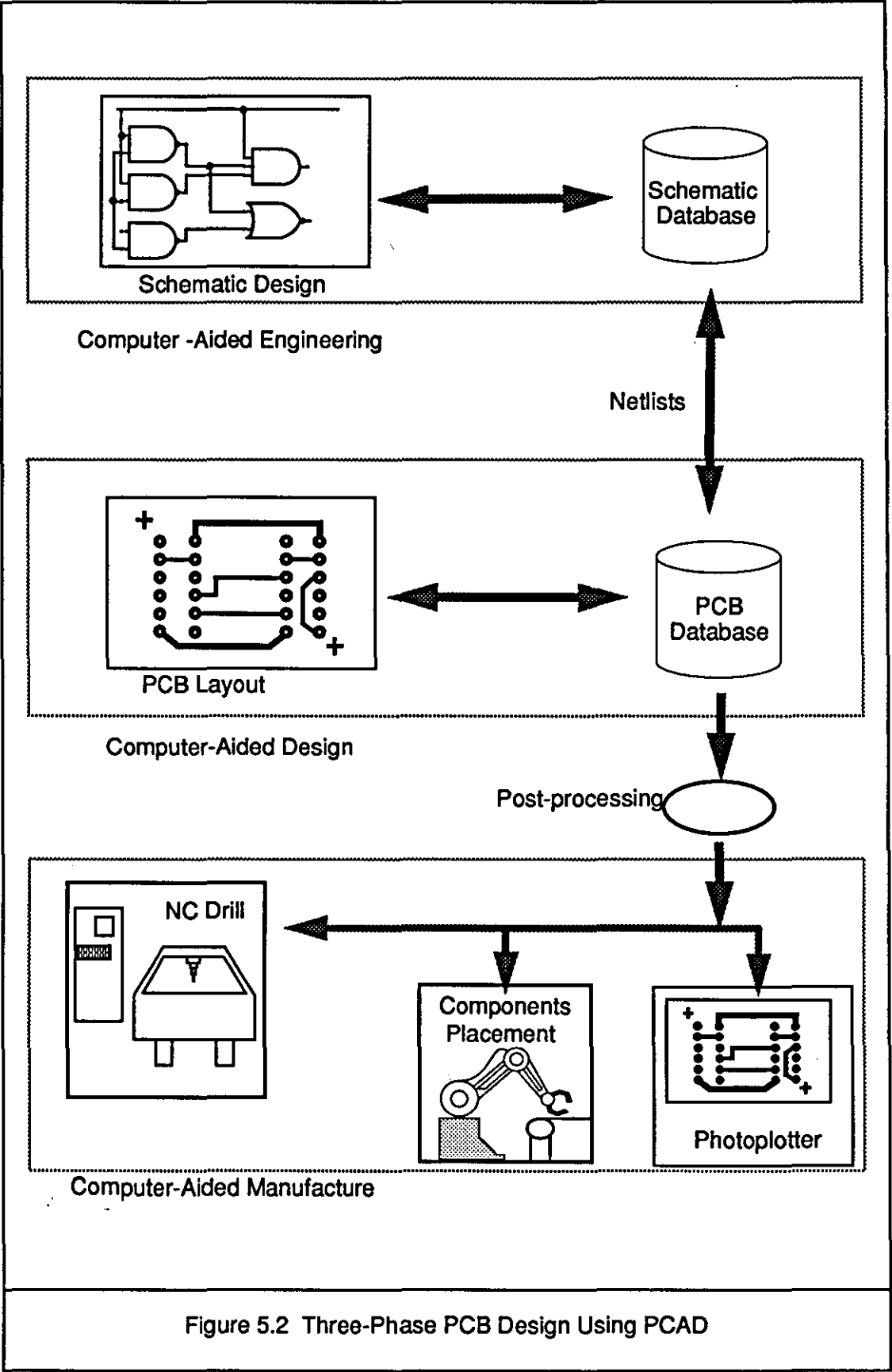
Figure 5.1 Linking CAD to AOI

The following subsections give some more detailed descriptions of each of the elemental systems and tools used.

5.2.1 The P-CAD System

The P-CAD system utilised is a computer-aided-design system supplied by Personal CAD Systems, Inc. It is in fact a CAD software package (which can be installed on any IBM PC-AT, PS/2, or PC compatible) developed for the purpose of computer-aided PCB product design. This system allows users to perform a three-level design of PCB products, as depicted in Figure 5.2

- 1) Conceptual design of circuits (known also as schematic design). This allows design engineers to enter the original product specifications and to represent the original concepts or ideas by means of schematic symbols. This is achieved using a graphic editor. A schematic netlist is generated here and delivered to next stage of design. Thus it ties together the conceptual design and the physical design of a PCB.
- 2) Physical design of a PCB layout. This includes the design of the geometry of a board and all necessary board features such as the fiducial marks, conductive tracks, various via holes and footprints (positions of components on a PCB). Facilities are also provided to assist the designer in component auto-placement (the distribution of component footprints within the geometry) and auto-routing (necessary electrical signal, power, ground lines). Various libraries (e.g. describing physical components, schematic symbols, etc.) are available to support layout activities.
- 3) Generation of manufacturing information for processes such as NC drill, photoplotter, and automatic component placement equip-



ment (e.g. ACI/ACO), through accessing the PCB database and post-processing the stored design information.

The ASCII output of the PCAD system is in a proprietary P-CAD Database Interchange Format (referred to as PDIF) [P-CAD 1989a]. This PDIF file contains information about board geometry (i.e. outline), PCB layout, electrical connectivity maps (netlists), components types and placement locations, and manufacturing information (e.g. for drilling, photoplotting, component placement, etc.). The structure of the PDIF file is formally defined and loosely based on the evolving industry-standard EDIF (Electronics Data Interchange Format [Hillawi and Bennett 1986]).

5.2.2 The Matrox Machine Vision System

The Matrox machine vision system is basically the same vision system as that introduced in chapter 4; here being referred to as an AOI system. The focus of research activity in this study was therefore to design novel vision algorithms for the AOI systems, which make use of the available CAD information in achieving execution of PCB inspection tasks.

Once again the vision system is used in conjunction (i.e. via an integration couple) with the AdeptOne robotic system to achieve PCB inspection. Thus the mobility of the camera can be maintained for the inspection of large circuit board. Furthermore the author believes that the inclusion of a mobility capability provides means by which a further generalisation of the experimental work is enabled, as increasingly it is likely that future vision system will need to move within their industrial work-place.

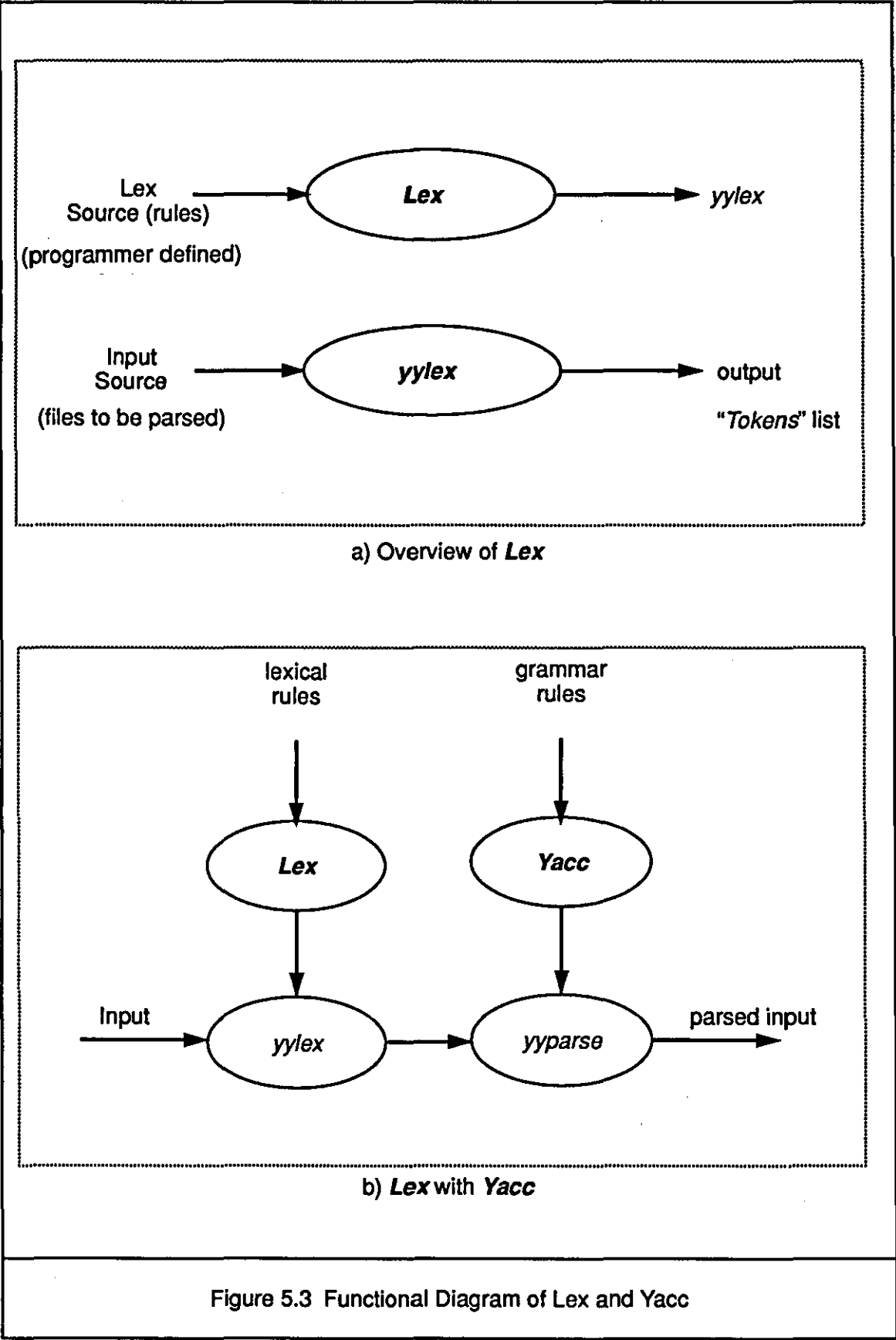
5.2.3 SUN Workstation and the LEX/YACC Tools

As previously mentioned, the PDIF files are in a proprietary format specifically designed for the purpose of information interchange between P-CAD databases (schematic database and PCB layout database). The information contained in such PDIF files is created mainly to serve the internal purposes of the P-CAD system; therefore some of the information contained in the PDIF files will not be relevant to the purpose of PCB inspection. Although some of the information is useful, it is not directly accessible to the devices of the AOI system. Thus the PDIF files have to be processed before their information content can be re-utilized by AOI applications. In this study this processing of the PDIF file is performed by the software created by the author and run on the SUN workstation.

Here, the UNIX programming tools LEX (A Lexical Analyser Generator) [Sun 1986a] and YACC (Yet Another Compiler-Compiler) [Sun 1986b], which are embedded tools of the SUN workstation, were utilised to create a software information generator which is run on the SUN workstation and performs the following operations:

- a) Taking as input and parses the PDIF file,
- b) Extracting the required information from the input PDIF file, and
- c) Re-formatting the extracted data into an output file in a format suitable for the AOI system.

Hereafter in this thesis, the term “software information generator” (referred to as “Software InfoGen”) will be used to represent the forging specified file parser, information extractor and reformattor. A functional diagram of the LEX and YACC is given in Figure 5.3, which illustrates the general case of using UNIX tools for the creation of a software information generator. Some details about the specific Software InfoGen created by the author using the LEX/YACC tools will be presented later in section 5.4 of this chapter, following a discussion in next section

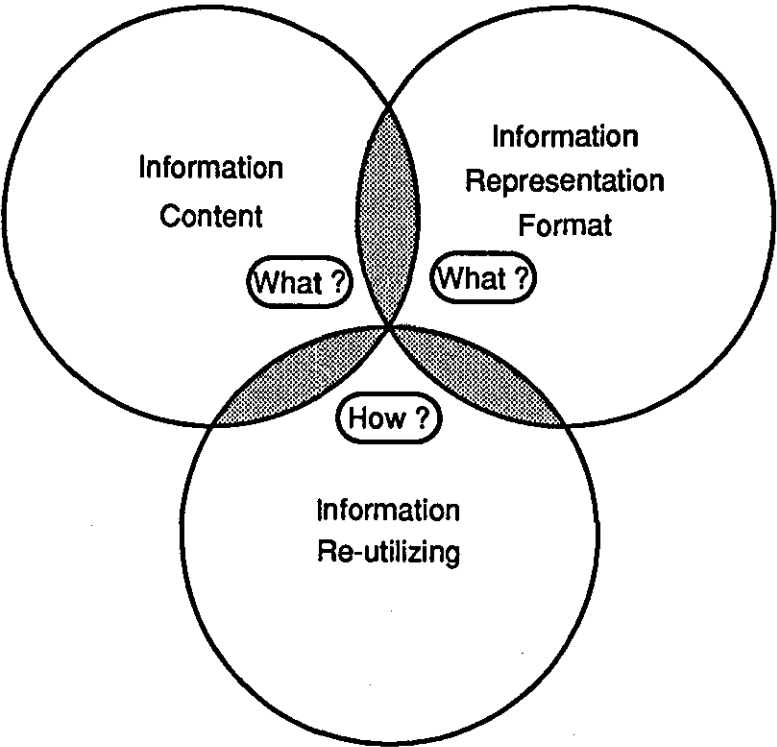



which addresses the more general issues concerning the re-utilization of design information by AOI systems.

5.3 General Issues of CAD Information Support for AOI Applications

While a commercial CAD system may differ from a turnkey AOI station in many ways (e.g. elementary hardware, basic user interface, usual data storage, proprietary data format, etc.), in regarding to the difficulties faced by the author the major disparity between them lies in their differing information requirements and the use of different representations of internal information. Thus, in this section, the concentration is on identifying features of the information (which is to be extracted from a CAD system) required to support AOI systems. Specifically, these features are considered in the context of defining and producing readily accessible and useable information for the AOI system. To this end, the author has identified three general questions which require answers (as depicted by Figure 5.4), viz:

- 1) What classes of information need to be extracted from a CAD system database or a CAD output file? This is concerned with the “information content” required by the AOI system.
- 2) How should the extracted Information be presented to the AOI system so that effective use can be made of the reference information? This is concerned with identifying an “information representation format (IRF)” suitable for a specific AOI system.
- 3) How should the reference information be interpreted and utilized by the vision system? This is concerned with “re-utilization of the CAD reference information” by the AOI system in its execution of PCB inspection task.



 Question Domain

Key:


 Intersection represents inter-dependency

Figure 5.4 Three Questions Concerning CAD Reference for AOI

These three questions need to be answered before appropriate CAD driven information supported AOI applications can be realised. Obviously these questions would be simplified if there exists a neutral format that is adopted by both the CAD system and the AOI system involved. Sadly this is not yet the case; therefore in the absence of a neutral (and ideally standard) format, and before the advent and wide adoption of such a format by the systems involved, any solutions to the problem of providing CAD information support to AOI applications will inevitably involve efforts of linking the relevant non-conformant systems, interpreting proprietary CAD data format, extracting and re-formatting the demanded information of which use can be made by the AOI system. This process is illustrated in Figure 5.5.

A further examination of these three questions revealed that (a) specific answers to the first and the second questions are related to the establishment of the overall information requirements of the specific AOI system involved and thus to the design of the information generator which provides the needed information, and (b) the answer to the third question is linked to the internal architecture of the AOI system and thus related to structural and operational characteristics of its software algorithms and/or hardware configuration. However, these questions are also mutually dependent (represented in figure 5.4 as intersections of two relevant question domains) in the sense that the answer to any one question will have effect on or be affected by the rest. For example, the information content is dependent on the AOI application as well as the overall approaches adopted by the AOI system; the selection of an information representation format will, to some extent, depend on the information content (so that all the content can be easily accommodated) as well as the AOI software algorithms (so that fast access to and precise interpretation of the information can be attained); the actual design and implementation of AOI algorithms for re-utilization of CAD information will in turn depend on the information content and the associated data structure and representation format used. The tightly-coupled nature of these questions has indeed made the task of re-using CAD information to support AOI applications more complicated.

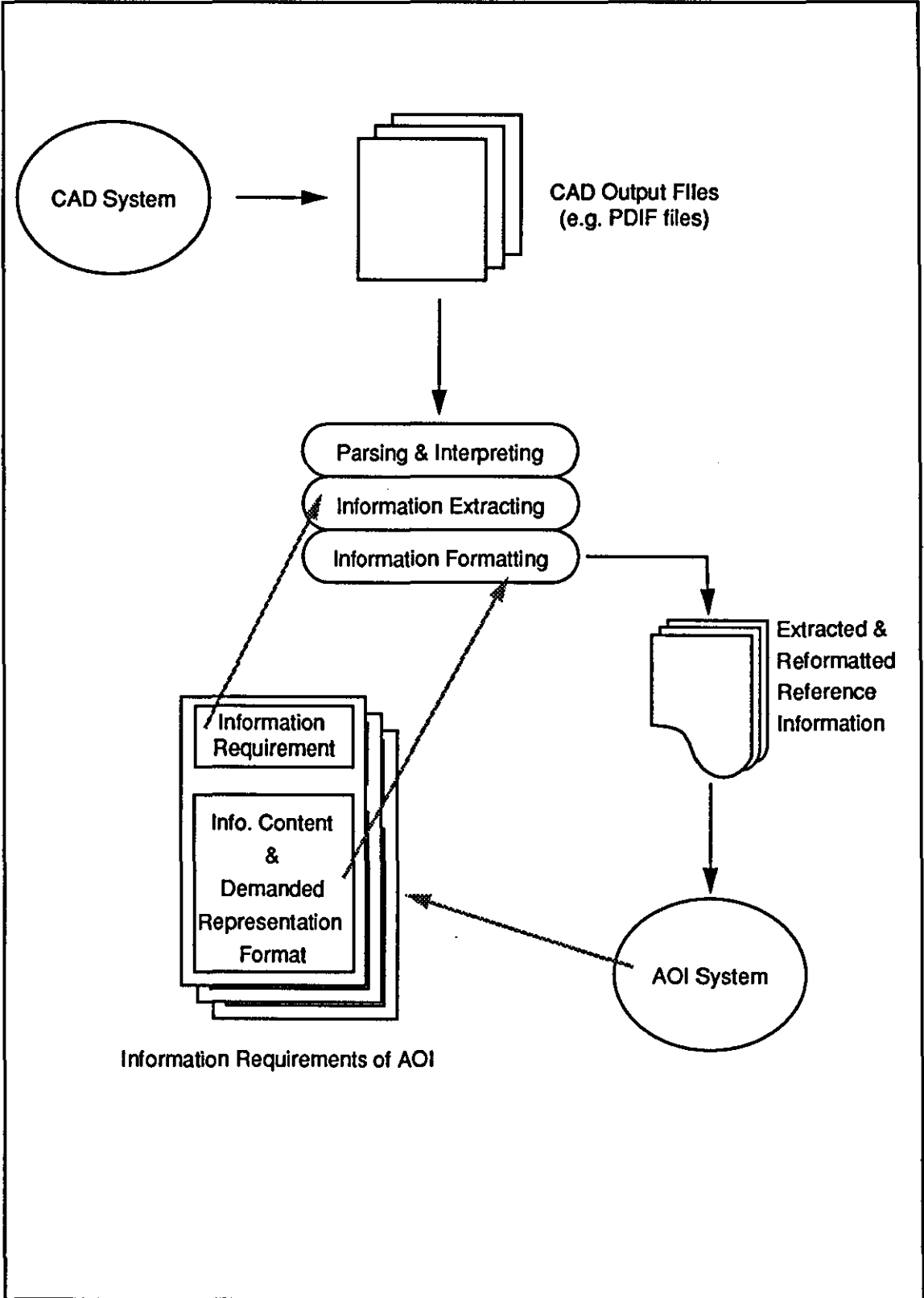


Figure 5.5 Providing CAD Reference for AOI Applications

5.4 Designing of a Software Information Generator Using LEX/YACC

As has been described in subsection 5.2.3, the major operations identified for the Software InfoGen are parsing the input, extracting information and formatting the output. Thus in order to design the Software InfoGen using the LEX/YACC tools, one needs to have an insight into the structure of the input files (here the PDIF files) and its conventions used in representing information (obviously, a knowledge of LEX/YACC is assumed). This implies that intimate familiarity with the PDIF structure is essential in designing the Software InfoGen. Furthermore, answers to the first two questions outlined in section 5.3 (i.e. questions relating to the particular information content and representation format) must be found so as to assist in designing the Software InfoGen which can extract and format the needed information from the input files.

In this study of information supported AOI used in achieving bare board inspection, the author chose to extract the following information items from the CAD system (i.e. from its PDIF output files).

- a) The Overall board geometry.
- b) The Nominal position, size and shape of board elements to be inspected. (e.g. fiducial marks, pads, tracks, etc.).
- c) The design netlist representing the electrical connectivity maps of the circuit board.
- d) Design rules such as specifying minimum feature size and spacing requirement.

An example of the P-CAD system output (PDIF file) is given in Appendix B.1. Figure 5.6 shows the PDIF keywords and the hierarchy of sections and subsections of the PDIF file structure.

The LEX source code generated by the author is given in Figure 5.7, which can be “compiled” by LEX to generate another software code “*yylex*” that performs lexical analysis on the input PDIF files (see Figure 5.3a) by recognizing the regular expressions (i.e. PDIF keywords listed in Figure 5.6) and returns “*tokens*” for later grammar analysis.

The YACC source code generated by the author is given in Appendix B.2, which can be compiled by YACC to generate “*yyparse*” that performs grammar matches on the “*tokens*” returned by “*yylex*” (see Figure 5.3.b). Figure 5.8 illustrates examples of the information formats used and implemented in the proof of concept CAD information supported AOI system. Examples of the information extracted and formatted using the Software InfoGen described in this section are given in Figure 5.9.

5.5 The Utilization of the CAD Information

The output file from the Software InfoGen described in last section is in a format suitable for the specific AOI system employed in this study, and its extension to bare board inspection tasks. This information is required to be accessed by the vision system. Thus special routines need to be devised, implemented and run on the vision system so that when executed they can read in the CAD information and make use of this reference information at appropriate stages of image processing, analysis and interpretation (decision making) thereby assisting in the accomplishment of inspection tasks.

As a result of carrying out the design processes described in this chapter, the author has recommended that the CAD information be utilised at three phases of image processing, namely

- 1) Using nominal positions of board elements as navigation information when capturing images of expected board elements or for

COMPONENT	Org	Cv
	Ty	Pv
ENVIRONMENT	Smd	Cv
PDFview	Jmp	SUBCOMP
Program	EX	COMP_DEF
DBtype	At	PIN_DEF
DBvrev		P
DBtime	DETAIL	Pt
DBunit	ANNOTATE	Lq
DBgrid	Arc	ploc
Lyrstr	C	PKG
Lyrphid	Fr	Rdl
Ssymtbl	L	Pnl
Apr	R	Pid
PCLR	T	Sd
Polyap	Poly	SPKG
PSIZ	Polyap	Sna
	Ol	Sp
USER	Pv	Apn
VIEW	Cv	PIC
Mode	Pv	Arc
Vw	Cv	C
Lv	NET_DEF	Fr
Gs	N	L
RCTL	DG	R
	W	T
DISPLAY	V	Poly
	Arc	Polyap
SIMBOL	Poly	Ol
PIN_DEF	Polyap	Pv
P	Ol	Cv
Pt	Pv	Pv
Lq	Cv	Cv
Ploc	Nn	ATR
PKG	ATR	IN
Rdl	IN	Ty
Pnl	Ns	Smd
Pid	Rats	Jmp
Sd	Un	EX
SPKG	PAD_STACK	At
Sna	Pad	I
Sp	PAD_DEF	CN
Apn	ATR	IPT
PIC	IN	ASG
Arc	Org	Rd
C	Ty	Pn
Fr	Smd	ATR
L	PIC	IN
R	Arc	Pl
T	C	Sc
Poly	Fl	Ro
Ployap	Fr	Mr
Ol	L	Ps
Pv	R	Pa
Cv	T	Nl
Pv	Poly	Un
Cv	Polyap	Iat
ATR	Ol	EX
IN	Pv	At

Figure 5.6 PDIF Keywords and File Hierarchy (From [PCAD 1989c])

CARD	[+A-Z0-9.'/*[?A-Z][A-Z0-9.'/*	PAD_DEF	return(PAD_DEF);
%%		PAD_STACK	return(PAD_STACK);
\	return(LCURL);	PDIFvrev	return(PDIFVREV);
\	return(RCURL);	PIC	return(PICTOK);
\	return(LSQR);	Pid	return(PID);
\	return(RSQR);	PIN_DEF	return(PIN_DEF);
\	return(NEG);	PKG	return(PKGTOK);
%%	comment();	Pl	return(PL);
"	{string();return(STRING);};	Ploc	return(PLOC);
\	return(yytext[0]);	Pn	return(PN);
[0-9]+	{yyval = atoi(yytext);	Pnl	return(PNL);
	return(DIGIT);};	Poly	return(POLY);
ANNOTATE	return(ANNOTOK);	Program	return(PROGRAM);
Apn	return(APN);	Ps	return(PS);
Apr	return(APR);	Pt	return(PT);
Arc	return(ARC);	Pv	return(PV);
ASG	return(ASGTOK);	R	return(RECTANGL);
At	return(AT);	Rats	return(RATS);
ATR	return(ATRTOK);	Rd	return(RD);
C	return(CIRCLE);	Rdl	return(RDL);
CN	return(CNTOK);	Ro	return(RO);
COMP_DEF	return(COMP_DEF);	Sc	return(SC);
COMPONENT	return(COMP);	Sd	return(SD);
Cv	return(CV);	Smd	return(SMD);
DBtype	return(DBTYPE);	Sna	return(SNA);
DBvrev	return(DBVREV);	Sp	return(SP);
DBtime	return(DBTIME);	SPKG	return(SPKGTOK);
DBunit	return(DBUNIT);	Ssymtbl	return(SSYMTBL);
DBgrid	return(DBGRID);	SUBCOMP	return(SUBTOK);
DETAIL	return(DETALTOK);	SYMBOL	return(SYMBLTOK);
DG	return(DGTOK);	T	return(TEXT);
DISPLAY	return(DISPTOK);	Tj	return(TJ);
ENVIRONMENT	return(ENVIRTOK);	Tm	return(TM);
EX	return(EXTOK);	Tr	return(TR);
Fl	return(FL);	Ts	return(TS);
Fr	return(FR);	Ty	return(TY);
Gs	return(GS);	Un	return(UN);
I	return(INSTANCE);	USER	return(USERTOK);
Iat	return(IAT);	V	return(VIA);
IN	return(INTOK);	VIEW	return(VIEWTOK);
IPT	return(IPTTOK);	Vw	return(VW);
Jmp	return(JMP);	W	return(WIRE);
L	return(LINE);	Wd	return(WD);
Lq	return(LQ);	[\n]	;
LS	return(LS);	{CARD}	{card();return(IDCARD);}
Lv	return(LV);	%%	
Ly	return(LY);	#include <string.h>	
Lyrstr	return(LYRSTR);		
Lyrphid	return(LYRPHID);		
Mode	return(MODE);		
Mr	return(MR);		
N	return(NET);		
NET_DEF	return(NET_DEF);		
Nl	return(NL);		
Ol	return(OL);		
Org	return(ORG);		
P	return(PIN);		
Pa	return(PA);		
Pad	return(PAD);		

comment() { char c; do { c = getchar(); } while(c != '\n'); }
string() { int i=0; strcpy(buffer, str); str[0]=" "; do{ i++; str[i] = getchar(); } while(str[i] != ""); str[i+1] = '\0'; }
card() { int i=0; strcpy(idbuf, idcard); idcard[0]=" "; strcpy(&(idcard[1]), yytext); while (idcard[i++]); idcard[i-1]=" "; idcard[i]= '\0'; }

Figure 5.7 The LEX Source Code Used for Parsing PDF Files

Format for Representing FIDUCIAL MARKs

FDMK	Location (x,y)	X_Dimension	Y_Dimension	Line_Width
------	----------------	-------------	-------------	------------

Format for Representing PADs

PAD	Location (x, y)	Shape_id	Dimension
-----	-----------------	----------	-----------

R – Round S – Square	Radius or side_length
-------------------------	-----------------------

Format for Representing TRACKs

TRACK	Track_Width	(x1, y1)	(x2, y2)	(x3, y3)	...	-1
-------	-------------	----------	----------	----------	-----	----

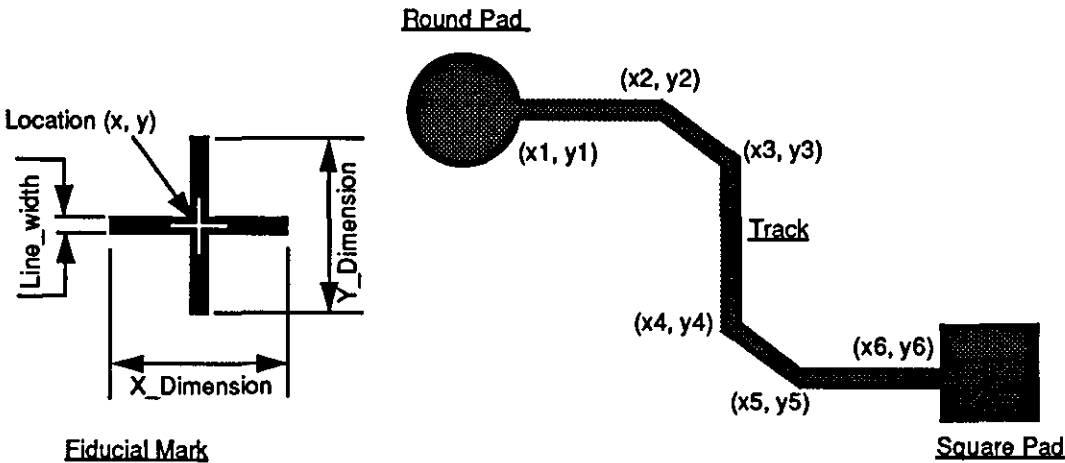


Figure 5.8 Examples of Information Representation Formats

FDMK No.	Location	X_Dimension	Y_Dimension	Line_Width
FDMK01	0 0	100	100	13
FDMK02	2300 1300	100	100	13

(A) Information about FIDUCIAL MARKs

Pad No.	Location (x, y)	Shape_id	Dimension
P001	0 1000	S	60
P002	0 900	R	30
P003	0 800	R	30
P004	0 700	R	30
P005	0 600	R	30
P006	0 500	R	30
P007	0 400	R	30
P008	300 400	R	30
P009	300 500	R	30
P010	300 600	R	30
P011	300 700	R	30
P012	300 800	R	30
P013	300 900	R	30
P014	300 1000	R	30
P015	600 1000	S	60
P016	600 900	R	30
...			

(B) Partial Information about PADs

Track NO.	Track_Width	(x, y) (x, y) (x, y) . . . -1
T001	13	300 600 500 600 500 650 700 650 700 900 -1
T002	13	1250 500 1050 500 -1
T003	13	200 700 0 700 -1
T004	13	1050 500 1050 450 200 450 200 700 200 1000 0 1000 -1
T005	40	0 900 -100 900 -100 1150 300 1150 -1
...		

(C) Partial Information about TRACKs

Figure 5.9 Examples of Information Extracted

locating image sections containing certain board elements for subsequent processing and analysing. The only assumption here is that the PCB positioning apparatus should operate with sufficient precision so that the expected board elements can be captured or located within a predefined processing window.

- 2) Using geometric information and design rules (e.g. minimum feature dimensions and/or spacing requirements) to assist in identifying objects (e.g. pads, vias, tracks, etc.), and as comparison criteria for making decisions regarding the quality of the identified board entity.
- 3) Using board geometry, electrical netlists, global PCB layout information and design rules as theoretical inspection criteria for determining the quality of the whole board under inspection.

In this research activity, a minimum set of software routines has been designed and implemented on the Matrox machine vision system which performs the bare board inspection tasks. With these routines, appropriate reference to PCB CAD information can be made. However, it should be pointed out that during the study introduced here only the first two phases of information utilization listed above have been implemented and tested. Opportunities for implementing the third phase is discussed later in chapter 8 of this thesis.

5.6 Discussions

Contemporary stand-alone AOI systems will be limited in their application if they do not meet current and emerging requirements of the PCB industry to facilitate integrated operations. With emerging CIM methods and open architecture, increasingly often computerised equipment will need to inter-operate with other manufacturing devices and subsystems in such a way that product lead-time, quality

and functionality levels are in advance in the competition of a given market. Thus future AOI systems will need to share information with other devices and subsystems (such as CAD/CAM stations), where such a capability will become compulsory ^{rather} than optional. The use of CAD information to support AOI is just one step toward the realization of fully information supported AOI applications in the computer integrated manufacture of PCB products.

In practice, care must be taken when 're-utilising' CAD-generated information. As discussed in chapter 3, the nature of the CAD data is that it describes only real defect-free products, whereas the real manufacturing processes almost always introduce distortions to the product under processing. Therefore, when utilize CAD information for AOI applications, allowable tolerances on process variables must also be taken into due consideration.

As a result of this study and the work reported in this chapter, a proprietary link between two heterogeneous CIM subsystems (namely an AOI system and a CAD system) has been established, providing a means by which CAD information can be used to support AOI of PCBs. Since this link is based on the 'pair-wise-integration' approach, it naturally inherits the drawbacks inherent in that approach (see chapter 4). Furthermore, since the information requirements and the desired information representation format are all AOI system/application dependent, changes of either application (e.g. as a result of product changeover or change of class of inspection task) or AOI system (i.e. different AOI system or even use of a different image processing algorithm) will naturally imply the need to re-engineer the Software InfoGen. This is not an ideal situation and one which is difficult for users in PCB manufacturing industry to be able to support.

However, through using a standard, formal conceptual model describing the required information content and a neutral format for representing the extracted CAD information, a level of independence between the overall information requirements (content and format) and the AOI internal structure can be achieved. In this

way, the three mutually dependent questions, as outlined in section 5.3, can be decoupled to a certain extent. This is very important in relaxing the burden of engineering the Software InfoGen; therefore a more flexible integration of the CAD and AOI systems can be attained. This issue will be further elaborated on in the following chapters by introducing and implementing a flexible integration scheme where PCB product models are employed to hold the demanded information for various processes/subsystems (including AOI) involved in the life cycle of PCB products.

Although, in this study, the approach to utilizing CAD information to support AOI is simply one of extracting and re-formatting the CAD information for use by AOI applications, it also offered an opportunity for researching into future requirements of both AOI and CAD systems so that they can support interaction and information interchange. Many of the problems encountered, relating to the integration of CAD and AOI are of a generic nature. Such generic problems require solutions to the following questions, which are generalised versions of the questions raised in section 5.3 and depicted by Figure 5.4, these being:

- 1) What Information content is required by the specific application.
- 2) What Information representation format is suitable, this being dependent on individual characteristics of the systems employed.
- 3) How will the information be re-utilized within the system to be information-supported.

5.7 Summary

Examples of the re-use of CAD information to support AOI application is presented and an approach to post-processing and re-utilizing design information in PCB bare board inspection is discussed. Specific topics regarding the underlying requirements for, and the proper representation format of CAD information have

been discussed in greater detail. From the author's point of view, apart from simply making CAD information available to AOI systems, effort should indeed be given to evolving an open and flexible integration method so that full information support for PCB manufacture and for AOI applications can be achieved, and thus the full potential of AOI applications can be readily reaped. This implies not only that a flexible integration strategy is of paramount importance but also that an open architecture and novel algorithms have to be adopted and implemented in developing a new generation of information supported AOI systems. New ways of viewing the CAD information from the viewpoint of AOI must also be appreciated.

To summarise, the author believes that the following two major issues will have to be tackled before the flexible integration of AOI systems and other CAD/CAM workstations can be widely achieved, namely,

- 1) A proper approach or methodology for achieving system and information integration is required, which should result in a highly configurable, open and integrated system environment.
- 2) A suitable, preferably open, architecture and a range of carefully designed algorithms for the AOI system is required. This would facilitate the exchange of information between AOI and other CAD/CAM stations and facilitate the utilization of relevant information fragments by the AOI system in its execution of automatic inspection tasks.

Chapter 6

A Product Model Based Approach to Integrating Vision Systems in the CIM of PCB

6.1 Introduction

Much has been discussed about the importance of the applications of AOI (automatic optical inspection) systems in the PCB manufacturing industry [Landman 1988] [King 1990] [Lozano 1990], and so too about the importance of making reference to CAD/CAM information when executing an AOI task [Powell and Carignan 1989] [Wright 1990] [Dolberg and Kovarsky 1989] [Rittichier 1989]. Yet in reality almost invariably AOI systems are still employed in a stand-alone manner, which has greatly limited opportunities for information sharing between AOI systems and other PCB product realization processes/subsystems and in turn has generally limited the possibility of making more timely and informed decisions. In order to integrate AOI systems with many other heterogeneous systems, one needs to take into account the inherent disparity between these systems, and then chose the proper approach accordingly. Ideally an integrated system should possess the following advantageous characteristics:

- 1) Being flexible in the face of changes in system's configuration (i.e. removal of a old system, addition of new systems, changeover of products, adoption of new standards in production lines and/

or products, etc.), and

- 2) Facilitating the sharing of the various information fragments resident within the systems being integrated.

In the particular case of AOI application, such an ideal system should facilitate the utilisation of CAD/CAM information by AOI systems as well as the utilization of AOI generated information by other processes, with flexibly configured linkages established between AOI systems and the various design and information support systems involved.

6.2 Notion Relating to a Product Model Based Approach

6.2.1 Use of Terminology

Although there exists a collection of commonly used terms in the area of product modelling, in a manufacturing context the use of such terms is not necessarily consistent. Thus there exists a need for a definition of the terms used by the author in this thesis.

Model

The term *model*, in the context of this thesis, is defined as “some representation of an object”. In other words, a model represents, describes or defines an object with a certain degree of granularity (or resolution) and completeness.

Product Model:

A product model is a model of a product, e.g. a printed circuit board. The PCB product models referred to in this thesis is considered as explicit descriptions of a PCB product, providing information concerning certain aspects of a PCB product.

The PCB product models referred to can be classified as being one of two types: according to their intended “*use*”; these two types being (i) *conceptual product models* or (ii) *physical product models*.

Conceptual Product Model:

In this thesis, the term “*conceptual product model*” is defined as “a conceptual description of a product, often in terms of entity-relationships, entity attributes and descriptive rules, which serves to give a qualitative definition of the product concerned.”

“EDIF conceptual models of PCBs” [EDIF-PCB 1990] are typical examples of this type. In this model a PCB is defined in terms of its constituent elements and their associated attributes, relationships and descriptive rules. However no exact or absolute data values are included to define for example the actual board size, component type, track width, and so on.

Physical product Model:

The term “*physical product model*” is used in this thesis to refer to “a description of a product which includes data that defines the desired values of certain product attributes (e.g. size, shape, etc.); which serving to give an unambiguous quantitative definition of the particular product concerned (e.g. for product realising processes).”

For instance, proprietary CAD output files such as PDIF data files generated by the P-CAD system are typical examples of this type of physical product models. In such data files, detailed information is provided to define the expected (or “theoretical”) PCB product in terms of attributes such as board size, number of layers, pad shape and dimension, track width, and so on. In this thesis, such a data file will be considered to be a “*physical model of the PCB product*”.

Each type of product model is further distinguished according to the “*scope of its*

use"; i.e. whether the information content of the model is to be used at various stages of product realisation, or confined to a more restrictive sub-set of these activities and processes.

Global product Model:

The term "*global product model*" is used in this thesis to denote "a model whose content is intended to be used to form a broad description of the product concerned, so that its content can be used to provide information which can support the various processes involved in realising that product.

Such a *global product model* could be structured and implemented in various ways. One particularly important point to make here is that it can be centralized in one location or distributed among a range of related subsystems or processes. Also product model data can be stored on various storage media, and data access mechanisms can be installed using either standard or proprietary methods.

If the "description" given in a global model is in terms of general rules, then such a global model will be considered to be a global *conceptual model*; otherwise, if the "description" is given in the form of definitive data values, then it will be considered to be a global *physical model*.

Local product Model:

The term "*local product model*" will refer in this thesis to "a model whose content is used to form a description (possibly at a high level of detail) of only a certain portion or fragment of the product concerned, where this content will provide information support for one (or more) specific local process(es) concerned with particular realisation aspects of the product". In other words, the "*scope of use*" of the information content of a local product model is limited to a local (possibly only one class of) process.

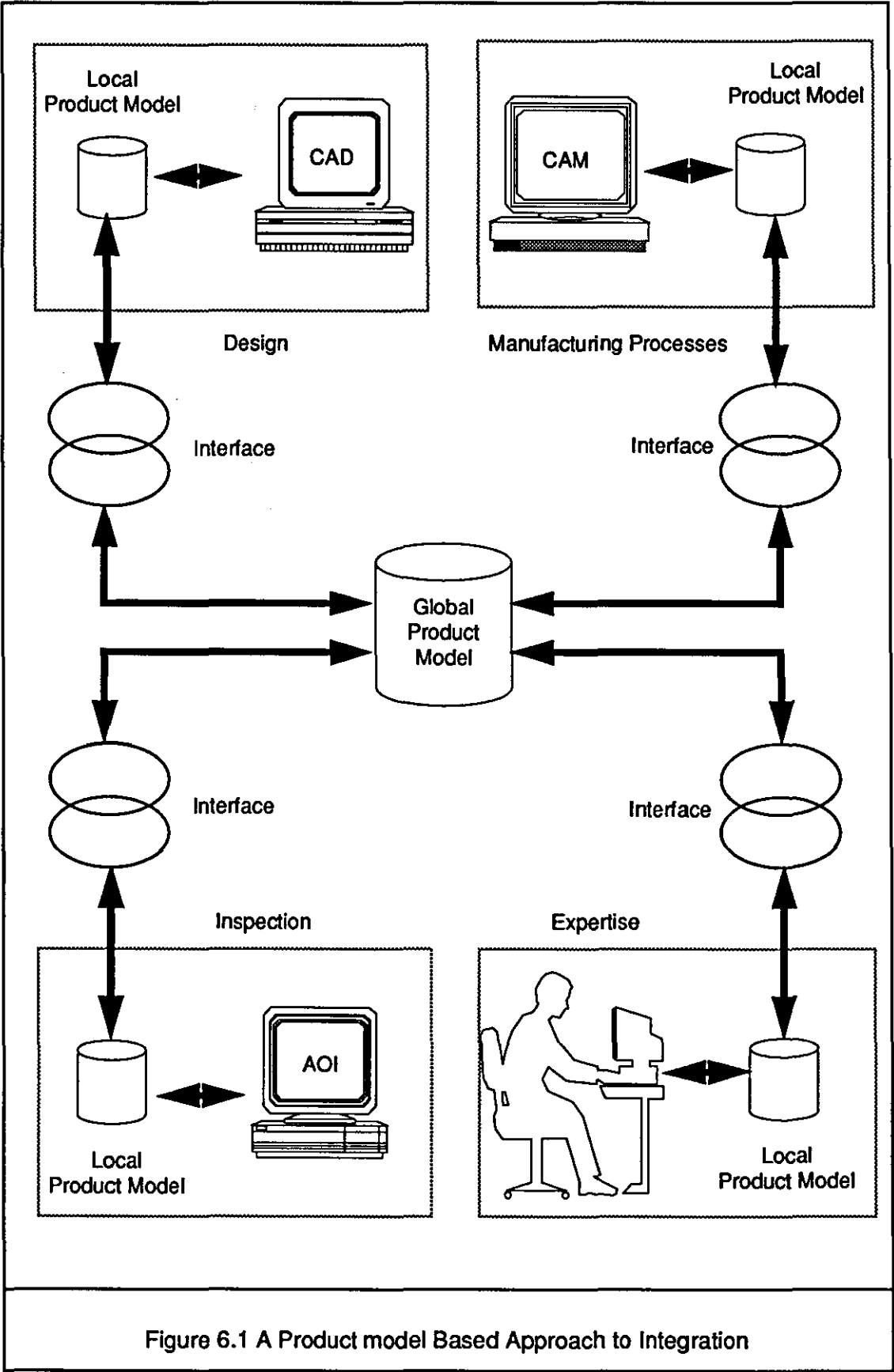
Again if the "description" given in a local model is in terms of general

rules, such a local model will be considered to be a local *conceptual model*; otherwise, if the “description” is given in the form of definitive data values, then it will be considered to be a local *physical model*.

6.2.2 A Proposed Product Model Based Approach

Based on experiences drawn from the integration studies presented in last two chapters, this section presents a product model based approach to integrating AOI systems within the CIM of PCB products (CIM/PCB). As depicted in Figure 6.1, the use of PCB product models is viewed as a means of providing information support to relevant sub-systems. In particular, PCB product models are employed to hold sufficient valid information to support various product realization processes; these including product design, manufacturing, test, inspection, etc. The various classes of computerised systems utilised such as machine vision inspection system (e.g. an AOI), NC (numerical control) drilling machines, ACI/ACO (automatic component insertion/onsertion), automated test equipment (ATE), etc. are thus viewed as local users of the product models. The main source of information content of the product model may well be a CAD system (or systems), whereas other sources may include engineer/operator input/modification, inspection and testing processes (such as AOI and ATE), manufacturing process statistics, etc.

The design information generated in a CAD system can be considered to represent a theoretical model of the PCB product as it will embody specifications of an ideal product: although as mentioned above additional information may need to be added to that model, or indeed model parameters may need to be changed, during subsequent product realisation. In reality it is difficult to guarantee that all products will be manufactured to conform exactly with the theoretical model, yet as previously described they may still be acceptable as products. Thus all the information generation or modification processes which change the model created during design can be viewed as updating and/or calibrating the original theoretical model so as to



make it more accurately represent a real world product (thus the information stored in the product models can become more meaningful and useful).

However, in order to maintain the “purity” of the model and in order to keep the model from being corrupted by illegal information sources or indeed by accidental errors of some systems, mechanism should be provided to safeguard the model so that its integrity can be maintained. For example, access control like “read only” may be imposed on the original model, “validation check” over the incoming information should be employed, and so on.

With this approach, the participant entities (subsystems, software tools, etc.) are broadly classified into three groups, namely

- A) *Product Models Group (PMG)*, which is a group of product models employed to describe/define a product from various points of view (i.e. conceptual, physical) and/or with various capacity or scope of use (i.e. global, local). See section 6.3.
- B) *User Group(UG)*, which includes all the processes involved in the PCB product life cycle which interact with the product model to either request information support or provide information input or feedback. In this sense, CAD systems are considered to belong to the user group.
- C) *Pre- and Post-processors Group (PPG)*, which is a group of software tools (pre-processors and post-processors) performing necessary data interpretation, information extraction and representation operations. See section 6.4.

The relationships between these three groups is illustrated in Figure 6.2.

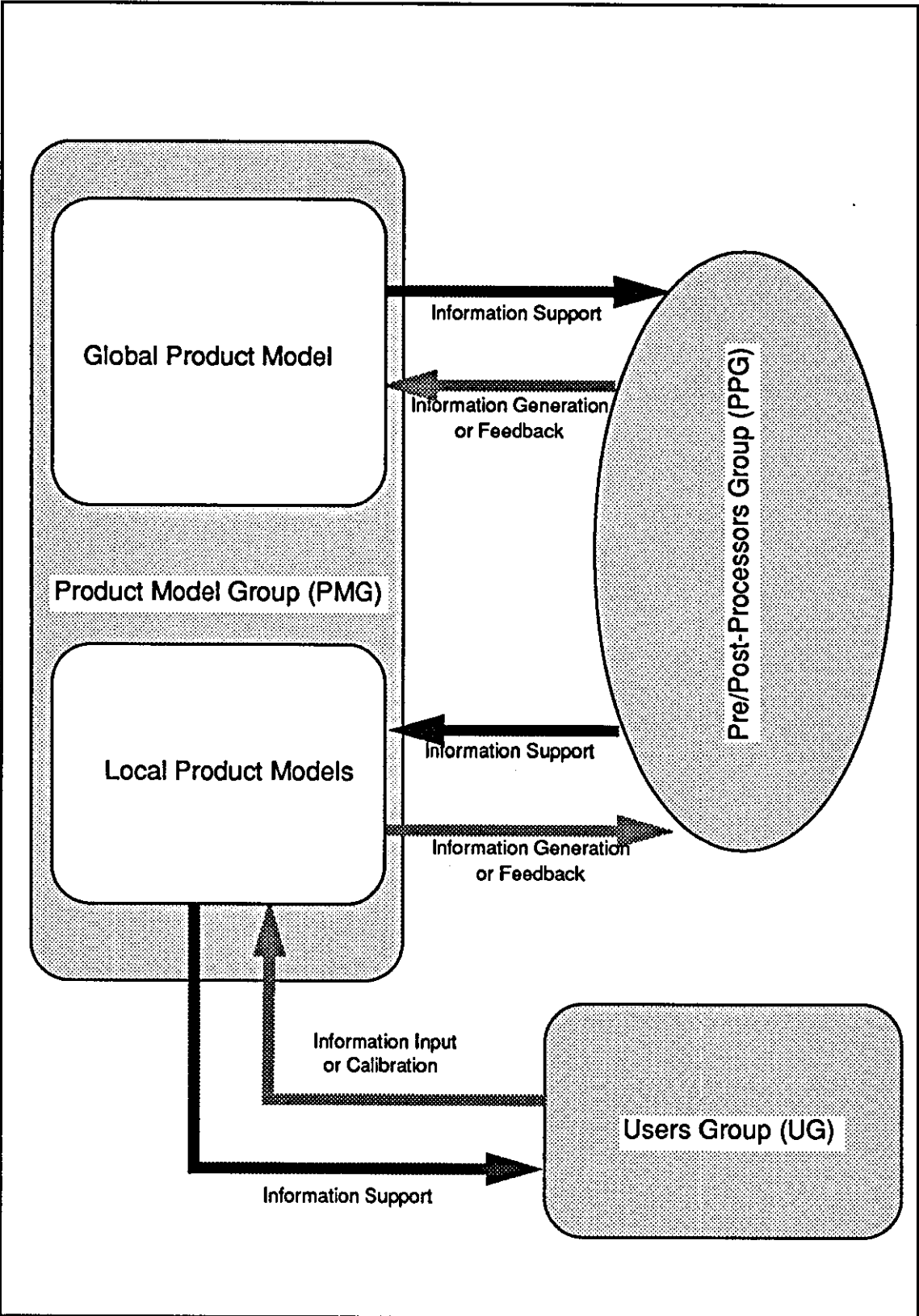


Figure 6.2 Relationships between PMG, PPG and UG

6.3 Some Issues Relating to Product Model

The use of terms in this thesis, especially in this chapter, relating to product modelling is based on the definitions given in subsection 6.2.1 and constraints imposed thereof.

As outlined earlier, the product models are expected to hold sufficient information about a product so that as required the model can provide appropriate information support to all processes involved in the PCB product life cycle. Usually this information support can be realised through post-processing the information stored in the global product model (see Figure 6.2) so as to cater for the specific information requirement of a particular application. It is worth stating clearly that different processes will normally have different information requirements, and this will dictate differences in the associated post-processing software, though one cannot rule out the possibility of two different applications sharing the same information requirements. In other words, the information requests will be essentially application dependent and the integrated system must be capable of catering for all types of application dependent information requests.

To demonstrate the differing information requirements, let us examine three different applications, namely,

- An NC drilling operation,
- The automatic component placement, and
- AOI of inner layers of a board.

Obviously, for NC drilling operations, the machine needs to know where on the board to drill holes with what sizes (i.e. the information about hole location and size).

For applications like automatic component placement, the equipment needs to know where to pick up what component (i.e. information about component

feeding position) and where to place the grabbed component at what orientation (i.e. the information about footprints of certain components on the board). In addition, information may well be required as to the properties of the components themselves (e.g. packaging type, handling method, specific grippers required, etc.) and various forms of testing or inspection operations (e.g. operations performed by a vision equipment on a SMT assembly machine).

For applications of automatic inner-layer inspection, the AOI system requires information (including inspection criteria) about all the entities to be inspected on that layer (i.e. the information about the nominal location, minimum size, designed shape, applicable manufacturing tolerance, etc. of the entity).

To help understand the role of the product models in this proposed approach, the following paragraphs present an expanded discussion of the classification and definition of PCB product model presented in sub-section 6.2.1).

6.3.1 Two Types of PCB Product Models

As defined in subsection 6.2.1, according to their use (or purpose), product models can be broadly classified into two types, namely, *conceptual product model (CPM)* and *physical product model (PPM)*. This is illustrated in Figure 6.3.

A) The Conceptual Product Model

The use of a *conceptual product model* (hereafter referred to as CPM) is to conceptually describe a product. This conceptual description is usually in the form of a set of rules or entity-relationship diagrams [Chen 1977], or more recently using the data modelling language EXPRESS [Express 1989]. Often a CPM is used to describe the building elements of a specific product (or part of the product), together with the relationships among them. However, it does not give any quantita-



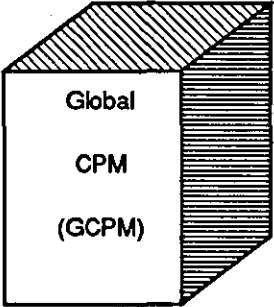
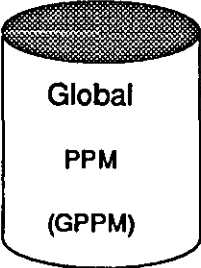






Scope \ Type	Conceptual Product Model (CPM) 	Physical Product Model (PPM) 
Global	 Global CPM (GCPM)	 Global PPM (GPPM)
Local	  Local CPM (LCPM)  LCPM	  Local PPM (LPPM)  LPPM

Figure 6.3 Types and Views of Product Model

tive definition as to physical features of the product such as the exact board geometry or track width.

Furthermore, a CPM does not specify the various possible ways in which the product model could be physically implemented, nor will it specify the possible methods to access the stored data. In fact, the detailed data defining a product may be held in the form of a set of flat files, or it could be stored in a database system, or it can take the form of hard-copied documents. Moreover, even in the form of flat files, the data could still be represented in many different formats and stored on various media.

Nevertheless, it is necessary and important to have this type of *conceptual product model*. Firstly, the model describes the product from a conceptual point of view; therefore, it gives a clear concept of what the product is to be and what constituents are to be expected to make up the final product. Secondly, it defines all items of the information content to be expected in a physical product model; therefore it can be used as “pointers” to information content.

B) The Physical Product Model

The use of a *physical product model* (hereafter referred to as PPM) is to define physical properties of a product in terms of quantitative data values, providing all necessary and detailed information about a product (or a portion of a product). In other words, it contains detailed specifications of all building elements of the product.

This definitive specification can be anything necessary to achieve the realization of the product, such as the specific technology and/or processes utilised to build the board (e.g. surface mount technology, fine line technology, multi-layer technology, and the number of layers making up the final PCB, etc.), or the precise dimensions of a specific circuit feature such as the shape and size of a pad or the

width of certain conductive tracks. Furthermore, as mentioned earlier, a PPM can be stored in various forms like flat files or database (centralised or distributed); it can be stored in computer memory, hard/floppy diskette, magnetic tapes, etc.; and it also needs to contain information about how the data can be accessed. It is obvious that without this physical definition, the product will remain in concept.

6.3.2 Two Views of the Product Models

According to the “scope”, or the “completeness” of their information content, both the CPM and the PPM can be further distinguished as “*global*” or “*local*” (see Figure 6.3).

A) Global Conceptual and Physical Product Models

A *global conceptual product model* (hereafter referred to as GCPM) is itself a *conceptual product model*, describing all necessary building elements of which the final product is to be made up and the relationships between all these building elements, as well as providing some product-realisation-process related information. The information content of such a GCPM is therefore concerned with the entire product.

A *global physical product model* (hereafter referred to as GPPM) is itself a *physical product mode*, defining physical properties of the entire product in terms of actual data values, as well as specifying actual processes utilised in achieving the realisation of the defined product. The information content of a GPPM can be accessed and processed to support all processes and systems involved in the various life cycle stages of a PCB product.

Although the physical model can be implemented in various different ways, it is highly desirable that a neutral (and ideally standard) data structure and information format be adopted in representing the information contained in a GPPM.

By doing so, firstly, the physical product model will have less dependency on any specific system vendor. Secondly, efforts involved in writing the pre- and post-processors can be reduced. This is because post-processors will now share the same (i.e. the chosen neutral) data format as input, whereas the pre-processors will have the same (i.e. the chosen neutral) data format as output (refer to Figure 2.11 for an illustration of the concept of pre- and post-processor and the neutral format). In other words, to a certain extent, the pre- and post-processors could be *partially* standardised. Thirdly, the author believes, the advantage of adopting standard data representation format will encourage system vendors to introduce standards into their product (e.g. hardware systems, software packages, etc.), which will eventually make easier the practice of systems integration.

B) Local Conceptual and Physical Product Models

A *local conceptual product model* (hereafter referred to as LCPM) is itself a *conceptual product model*, containing only partial information (which is application-oriented) about a product. It conceptually defines that part of the product which is of interest to a specific application process (e.g. drilling machines, component placement equipment, inspection stations, etc.). The use of such a local conceptual product model (LCPM) is to define conceptually the information requirements of a particular application. For example, in the case of bare board inspection, the LCPM will describe all the information items relevant to the inspection task, and the relationships between them. Such a conceptual model will also serve to guide the collection of all fragments of information required to support this specific case of application.

A *local physical product model* (hereafter referred to as LPPM) is itself a *physical product model*, containing partial information that defines physical properties of a fragment of the product. It can be considered as being a projection of the GPPM onto a specific application in the sense that it is a view of the entire GPPM

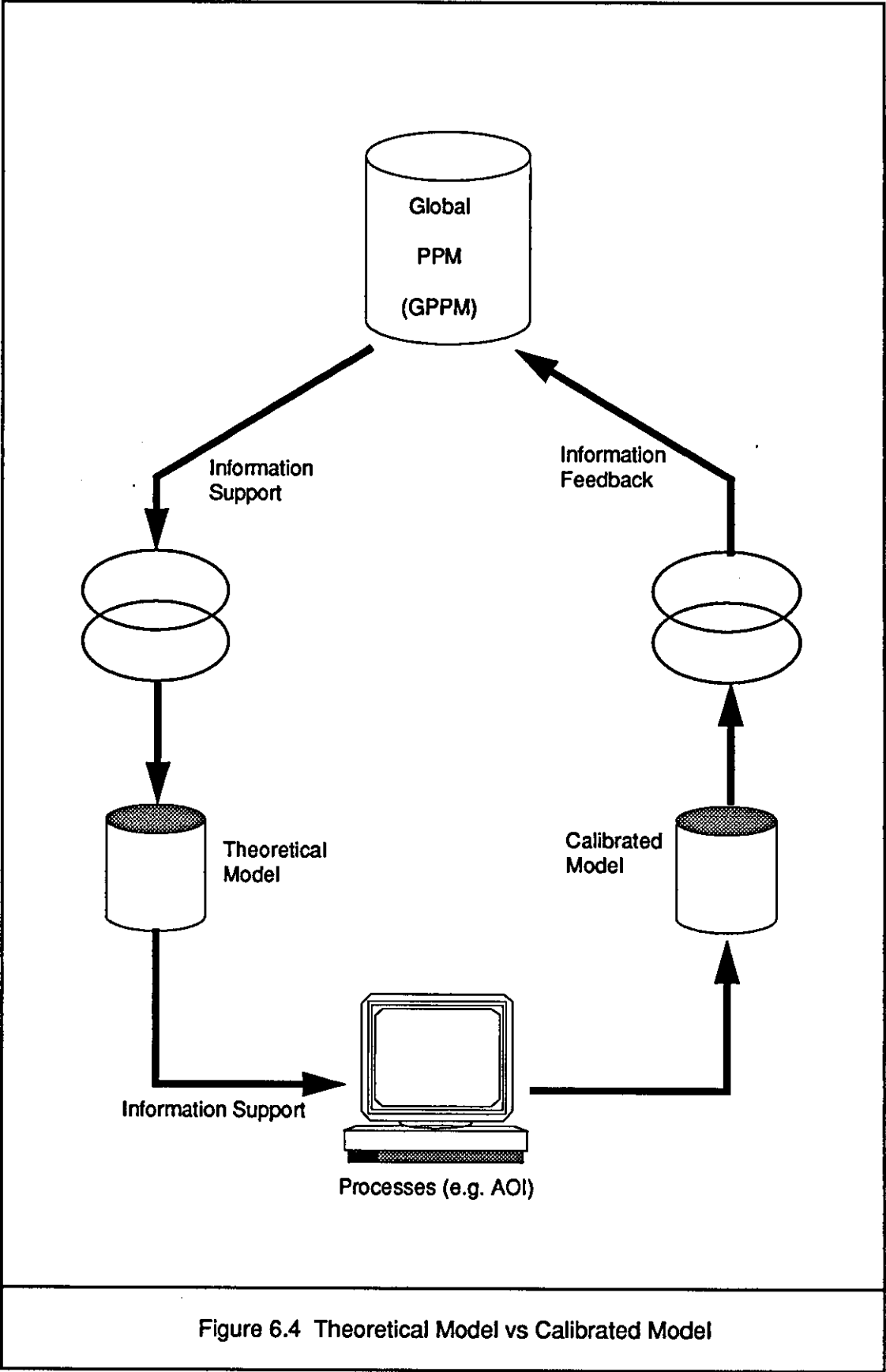
taken by that specific application concerned. The information content of such a LPPM is incomplete and is only concerned with that part of the product which is of interest to the application under discussion. In other words, its information content is dependent on the information requirement of the specific application and thus is application-dependent. There will be a number of different local views of the same GPPM so that a variety of application-dependent information requirements can be catered for. For example, an LPPM used for bare board inspection will contain information about the construction specifications and inspection criteria for all board elements subject to inspection. The information will precisely define the geometrical characteristics (shape & dimensions) of relevant board elements such as pads, vias, tracks, etc., as well as the demanded electrical interconnections among them.

The LPPMs are generated by PPG software through reformatting the information extracted from the GPPM, and it can be physically implemented as files of tables, etc., depending on the requirement of the local application concerned. It is also recommended here that some standard data structures and representational format be adopted so that the change of a local system will not necessarily imply the need of rewriting the existing software which generates the LPPM.

6.3.3 Theoretical Model vs Calibrated Model

So far, discussions relating to the characteristics of local product models has been focused on its theoretical form, i.e. the "*theoretical local model*" generated from the global product model. The counterpart of this theoretical model is the "*calibrated model*".

As stated earlier, an LPPM contains information required to define certain fragments of the product as it is "*expected*". On the contrary, a calibrated model contains information which describes actual properties of the product concerned. This concept is illustrated in Figure 6.4. For example, a theoretical model used by an NC drill would specify where and with what size the holes are to be drilled on a



layer, whereas a calibrated model will contain information about the actual positions and size of the holes drilled, providing information as to whether these holes have been drilled correctly (actual information relating to the calibrated model may well be provided by AOI systems). Clearly this information concerning, for example, any offsets from the desired positions can be used within subsequent product realisation processes (such as component insertion operations).

Another similar example can be found in bare board inspection. Here the theoretical model specifies all the expected entities (their expected location, shape, size, etc.) on a bare-board, together with the electrical interconnection and spacing requirement. This information can be utilised by the AOI system in its execution of the inspection task (e.g. use of the information to form part of the inspection criteria). Upon finishing the inspection task, the output of the AOI system (i.e. the calibrated model) will contain information describing what the actual bare-board is like; for example, whether the expected board element is present at the right location with a demanded size, orientation, required spacing and connectivity, etc. Here the calibrated model will be an output of the inspection task.

6.4 Pre- and Post-processors

Pre- and post-processors are software tools which together establish links between the global product model and the various local product models whose information is consumed by the local processes or subsystems; therefore they establish links in terms of information sharing and support between individual systems. In other words, they provide a means of extracting information required from the global product model and of presenting the extracted information in a desirable structure and format to support local applications. Thus pre- and post-processors have the same functionality as that specified in Chapter 5 for a general software information generator (see subsection 5.2.3). The functionality is re-outlined here as follows.

- 1) parsing of the input information (which is represented either in a neutral data format or in a proprietary data format used by a specific system),
- 2) extracting desired information from the input, and
- 3) reformatting the extracted information into a desired format and presenting the formatted information either to a local application or to the global product model.

6.4.1 Pre-processors

Pre-processors are software used to transform the information required from a proprietary data structure and format as used by a specific local system (e.g. an AOI system, a CAD system, an NC drill station, an automatic component placement equipment, etc.) to a chosen data format used for representing the information stored in the global physical product model (GPPM).

6.4.2 Post-processors

This is the software used to provide local views of the GPPM for various local user applications. If a neutral or standard data structure and information format is chosen to represent the information contained in the GPPM, this data structure will be parsed by all the post-processors and desired information^{will} be extracted from the GPPM. Since, in the case of a neutral structure and format being used in GPPM, the input to all the post-processors will be essentially the identical, the parser will be the same to all these translators. Normally, each different application requests different information support that is to be represented in a different format; therefore each post-processor will require different routines to extract the relevant information from the general physical product model and to represent the extracted information in a desired format suitable for a given application. This point will be further reviewed in Chapter eight.

6.5 Summary

In this chapter, a scheme is proposed which aims to promote a product model based approach to flexibly integrating AOI systems within CIM of PCBs. With this approach, the use of product models is considered as a means of providing necessary information support to various processes and subsystems involved in PCB product realization.

Two types of product model have been distinguished according to their use, namely, the *conceptual product model* which conceptually defines a product by describing all the building elements making up the final product, and the *physical product model* which physically defines a product by providing detailed definitive information about its building elements.

The global view and local views of both the CPM and the PPM are further distinguished according to their scope of use and to their information content. The global product model holds the overall information potentially useful for all processes and subsystems, whereas various local product models hold information directly accessible and useable for a specific application. The provision of multiple local views of both CPM and PPM have been considered as a way of catering for varieties of application dependent information requirements.

With the identification of these characteristics of the product model, it is believed that the roles of various product models (all describing the same product from different point of views and/or at different levels of information completeness) in systems integration can be made clear. Pre- and post-processors have been discussed as basic software which can be used with these product models in integrating diverse systems and providing application dependent information support for local applications.

Chapter 7

Suggested Architectural Design of an AOI System

7.1 Introduction

The reader may recall from chapter 5 that major limitations of present generation AOI systems can be related to their inherent stand-alone nature, which has greatly restricted the manner in which AOI systems can share information with other manufacturing systems and processes. As a result, AOI systems often have to rely on a restricted supply of information concerning “known good boards” (or “master boards”) as a basis for establishing inspection criteria [Rittichier 1989]. This dependency on limited information can lead to inspection error and can result in it not being possible to accomplish certain classes of inspection task or to detect certain types of defect.

Thus the author has aimed to investigate the notion that AOI systems should operate as an integral part of CIM systems. To achieve this, AOI systems should be sharing global information resources as required and interact appropriately with other system building elements. Hence as a starting point the author has aimed to characterise information sharing and interaction requirements between AOI systems and other computerised equipment typically used in PCB manufacturing organisations, see Figure 7.1.

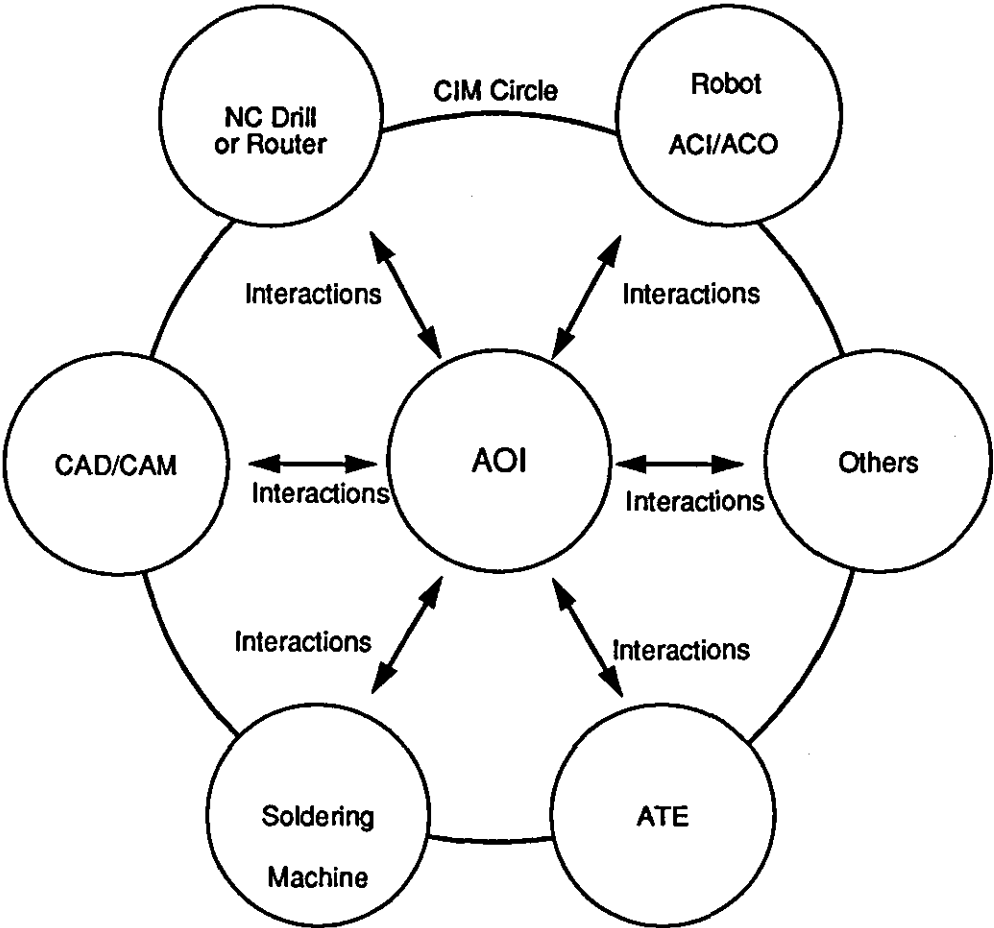


Figure 7.1 Interactions between AOI and Other CAD/CAM Stations

This chapter starts by reporting on an investigation into the characteristics of typical AOI applications in the arena of PCB inspection, which serves to identify the hierarchical nature of such AOI applications and helps to characterize typical interactions between AOI and other PCB product realization processes and systems. This investigation leads onto a proposed decomposition of a general AOI task. Based on this decomposition, a five-level reference architecture is suggested which can be used as a guideline to structure the design and implementation of AOI algorithms with a view to facilitating information sharing between AOI systems and many other classes of computerised equipment.

7.2 Characterizing Typical AOI Applications in PCB Manufacture

As discussed in chapter 3, AOI systems can be used at many stages of PCB product-realization processes. Mainly they are employed to accomplish various forms of product inspection, especially in situations where traditional human visual inspection has proved inadequate or unfeasible: particularly as new and advanced electronic devices, interconnection methods and PCB manufacturing technologies evolved to enable increased functionality to be realised on a smaller board area, leading eventually to it becoming impossible to employ human inspection methods [Gilutz 1988] [Landman 1988] [Lozano 1990]. To summarize, typical AOI applications are as follows:

- 1) Inspection of phototools (i.e. a film),
- 2) Inspection of individual layers,
- 3) Inspection of a finished bare board,
- 4) Inspection of component placement,
- 5) Inspection of solder paste application, and

6) Inspection of solder joints.

On considering image processing requirements, applications 1), 2) and 3) are essentially very similar (although of course appropriate imaging and illumination methods will need to be devised and adopted for each case). For these first three applications, two-dimensional images representing PCB layouts need to be analysed (if we can ignore for the moment the thickness of the copper clad used on the physical layers). Present in the image are those board elements of which the panel under inspection is made. The AOI system has to process the image so as to identify and locate features, carry out measurements, make decisions about the board elements being inspected, to combine the inspection results (as well as to examine the overall arrangement of these board elements within the 2-D space confined by the outline of the panel) and to establish the relationships among them (i.e. the electrical interconnection and the geometrical distribution among them). In this thesis, therefore, no distinction will be made between applications 1), 2) and 3); instead they will be treated as one (class of) application, and will be collectively referred to as "*Panel Inspection*".

Applications 4), 5), and 6) are each unique in their own right and will be dealt with individually.

Thus four basic classes of AOI applications were identified, namely *panel inspection (PI)*, *solder paste application inspection (SPAI)*, *component placement inspection (CPI)* and *solder joint inspection (SJI)*, as illustrated in Figure 7.2. It is important to point out that these four application classes were chosen by the author based on (i) an examination of the literature, (ii) discussions with personnel at ICL Kidsgrove and examination of certain PCB inspection related documents (specifications of inspection criteria), and (iii) conducting application studies at the university and extracting knowledge of the requirements imposed.

Hereafter in this thesis the word "*board*" will be used to refer to PCB-related things being inspected; for example, a master film, an inner/outer layer, a

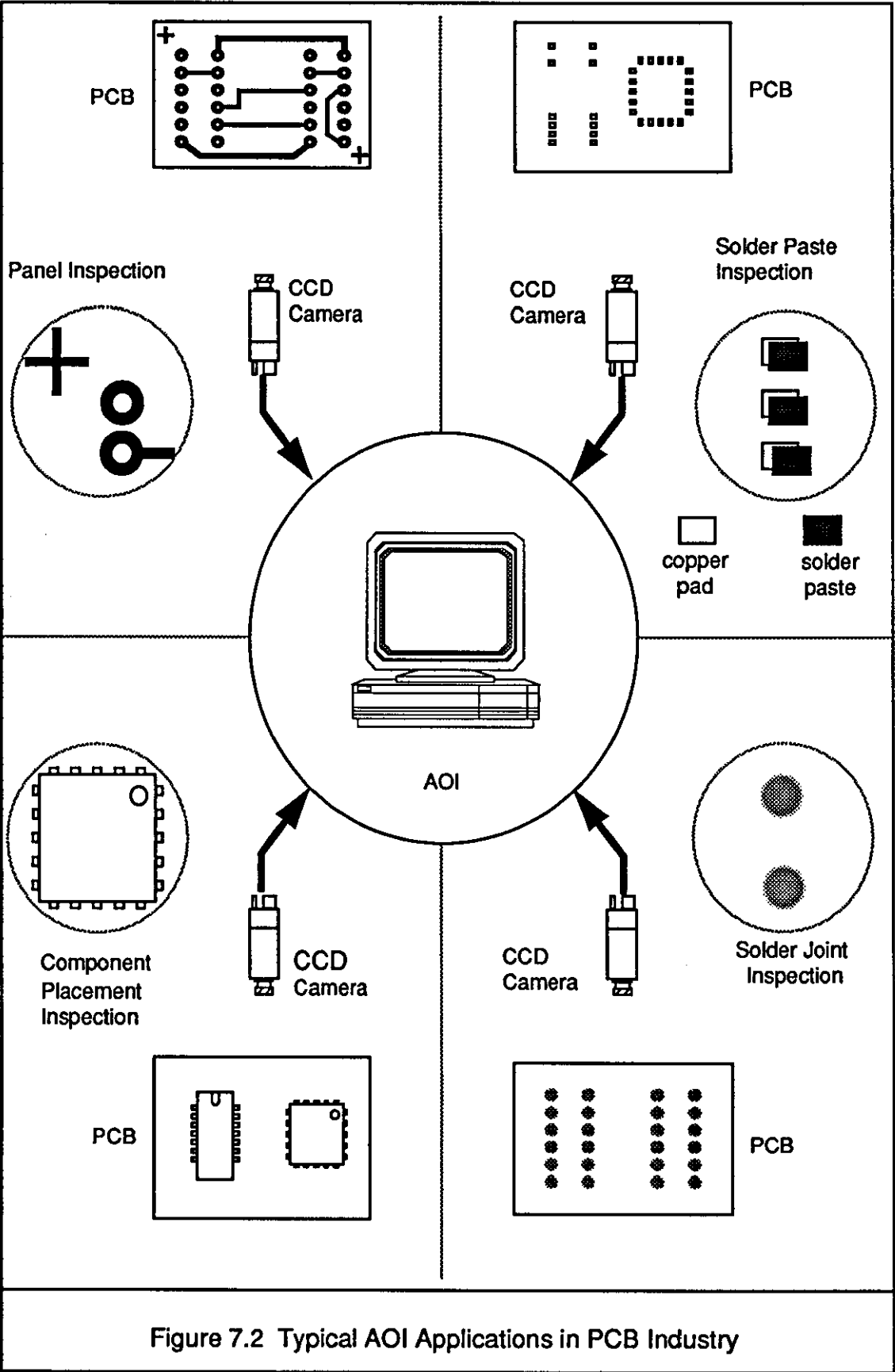


Figure 7.2 Typical AOI Applications in PCB Industry

drilled layer/board, a bare board, a populated board or a finished board. Likewise, the term “*board element*” (*BE*) will be used to refer to anything that is present on the “board” and that is subject to inspection; for example, tracks, pads, fiducial marks, various components, solder paste, solder joints, etc.

7.2.1 Further Examination of AOI Applications and Their Generic Characteristics

A. Panel Inspection (PI)

As previously discussed, *panel inspection* includes the inspection of both individual layers (i.e. inner layers, outer layers, or master artwork films) and finished bare boards. In most cases, the panels being inspected are inherently of a 2-dimensional nature and therefore 2-dimensional image processing techniques will normally suffice.

Individual objects (i.e. the BEs) present in the image would typically include soldering pads, inter-layer connection vias, conductive tracks, various printed components, etc. These BEs are usually inspected with reference to design specifications and allowable deviations from the specification. The relationships (e.g. the desired electrical interconnections) between individual BEs will also need to be verified; for example to check if a certain pad is connected to a certain conductive track. Furthermore, a global analysis of features of the board being inspected will be required in order to produce global board information that can be utilised, for example, as feedback information for the purpose of process control or adjustment. Thus the inspection process will naturally include the inspection of all the BEs and that which examines the relationships among them; therefore generally a task of panel inspection can be viewed as consisting of the following two subtasks, viz:

- 1) An inspection of all the BEs: This is concerned with locating,

identifying and measuring one or more features of each and every BE which is subject to inspection, as well as identifying its status of connectivity with others. CAD information about these BEs, where available, can be utilized in the appropriate stages of this subtask. For example, design rules such as minimum pad size and minimum track width can be applied to check against the quality of an individual BE.

- 2) Examinations and verifications of the electrical and/or geometrical relationships among the BEs, and a global analysis of the board being inspected, based on the result of subtask 1). This is to carry out examinations of the whole panel in order to verify for example the required interconnections between BEs and/or to produce feedback/feed-forward information which can be utilised by other processes. Likewise, CAD information can also be referenced in support of this subtask. For example netlists will help to check if correct interconnection among certain BEs is present on the board being inspected.

B. Solder Paste Application Inspection (SPAI)

In cases where SMT (surface mount technology) is used in PCB manufacture, solder paste is commonly applied by screen-printing on the finished bare-board before the process of SMC placement. The quality of the screen-printed solder paste often needs to be inspected, as defects in solder paste application could eventually result in defective solder joints (e.g. shorts, voids, bridges, etc.), and solder-related defects are often the most critical source of low product yields [Besl et al 1985] [Driels and Lee 1988] [Chen 1990]. Worse still, the detection and repair of such solder-related defects are really awkward, not only because these defects are not so conspicuous (some defects may only detectable through functional test), but

also because rework/repair can cause damage to the components. It is, therefore, highly desirable to inspect the solder paste before starting component placement processes [Mangin and McClelland 1987].

The image being processed during SPAI includes three dimensional objects (i.e. lumps of solder paste) and therefore 3-D image processing algorithms are required. Often alignment inspection and volume inspection are a requirement: as the volume of the solder paste and its alignment relative to the copper pads are important parameters affecting the quality of the final solder joint, especially as surface mounting and fine line technology are being increasingly applied in the PCB manufacturing industry. With a shrinking of component sizes and the increases in component lead count, the typical spacing between pads is getting narrower and narrower. Good solder joints can not be guaranteed without the right amount of paste being supplied in good alignment with the solder pads. This requirement for solder paste application again imposes a high demand for precise (or high resolution) inspection so that high quality solder paste application can be achieved.

In SPAI, the BEs are mainly those pads covered by solder paste, whereas vias and tracks which are normally not to be soldered should not be covered with any solder paste and therefore should not be present in the image; otherwise mistakenly applied solder paste should be reported (assuming special illumination methods and filtering techniques are employed to help obtain the image of only solder paste, if this is not true, special algorithms are required to distinguish solder paste from other materials so that succeeding image analysis focuses only on the solder paste under examination). Fiducial marks often need to be located so that the actual board frame of reference can be established. The task of SPAI can be considered as being made up of the following two subtasks, namely,

- 1) Inspection of individual BEs, i.e. pads (or component footprint patterns) covered with solder paste: Usually, a set of inspection criteria is applied for this subtask, which specify for example the

minimum volume of the paste and the allowable tolerance on misalignment between the actual location of the paste lump and the expected one. Other criteria with respect to the quality of the paste application can also be used, for example, the required surface and/or edge features of the paste [Prasad 1989].

Information generated at the stage of bare board inspection can be used here to provide information relating to the calibrated model of the board; this demonstrating one case of reusing AOI generated information by other processes (or AOI) at other stages (i.e. here for SPAI). Since part of the information generated by bare board inspection is related to the actual location of the footprints on which solder paste should be applied, it provides a relatively reliable reference for checking whether the paste has been applied in the correct location.

- 2) Global examination of the board being inspected: This is to perform a global analysis of the inspection results of the first level inspection (i.e. subtask1), which offers the opportunity for producing information feedback which can be used for the purpose of say solder paste application process control.

C. Component Placement Inspection (CPI)

Placement inspection (both pre- and post-soldering) is necessary in cases where there is less confidence in the reliability of the placement equipment, or where complex component such as complex SMCs (surface mount components) are used [Mangin and McClelland 1987]. On the one hand, pre-soldering helps to detect any misplaced components before they are physically soldered on the PCB; thus post-soldering rework can be minimised. On the other hand, post-soldering inspection is employed as the final step before functional test to make sure the presence of

certain components as required; this limits the range of possible variables affecting the electrical functionality and helps the ATEs to make decisions about the board being tested.

As the BEs (i.e. various components) to be inspected in CPI are themselves three dimensional, 3-D information is usually required if inspection of 3-D orientation of the components relative to the board is a requirement. The AOI system should be capable of locating and identifying all the components present in the scene, and at the same time to point out places where an expected component is absent (inspection of presence/absence). To distinguish a particular (type of) component out of a group of objects with similar or even identical shape and pin count, it seems that the only possible way to fulfil the task is to implement OCR (optical character recognition) algorithms, which will recognise the nomenclature (i.e. identity stamp) of the components and thus identify them. As the image to be processed naturally contains sub-images corresponding to individual components, the task of “component placement inspection” can thus be viewed as consisting of the following two subtasks, namely,

- 1) Inspection of individual BEs (i.e. components): Generally this is concerned with identifying and locating all components on the boards. However, different inspection criteria may be applied to different types of components. For example, inspection of high lead count, small pitch complex SMCs will demand the use of tighter alignment tolerance than that for the inspection of simple SMCs (e.g. chip capacitors).

It is also believed by the author, that tighter inspection criteria should be applied to pre-soldering placement inspection. Not only because rework and repair are easier before soldering, but also as thorough and precise pre-soldering inspection can be used to monitor and identify any undesirable trends in the placement and

assembly processes. Furthermore, a level of confidence over the placement of components (pre-soldering) will help identify process variations introduced by the soldering process according to the post-soldering inspection results. Thus, by clarifying the “responsibility for defect”, better use can be made of the AOI inspection result for the purpose of process control.

- 2) **Global examination of the board being inspected:** This involves combining fragments of information about sections of the board (i.e. local features of board), and performing overall analysis so that global information about the entire board can be obtained; for example, the number of (each type of) components placed on the board, classes and distribution of placement errors found, and so on. Such global information can be useful, for example, for the purpose of information feedback for the control and adjustment of the component placement process.

D. Solder Joint Inspection (SJI)

Of all the inspection tasks in PCB manufacture, automatic solder joint inspection is probably the most difficult one to tackle [Driels and Lee 1988]. A specially designed illumination system is often a prerequisite due to the reflective nature of the metal joint. For this purpose, various research projects have been conducted focusing, in the first place, on obtaining better images of the inspected solder joint; for example, by the use of structured lighting [Nakagawa 1982], tired illumination [Capson and Eng 1988] [Takagi et al 1991], and fluorescent-diffused illumination system [Driels and Lee 1988]. All these approaches have aimed to facilitate the inspection of solder joints formed during wave soldering processes (these typically being commonly used for through hole technology).

With the introduction of surface mounting technology, a further level of difficulty results [Pound 1988a]. Not only as the number of solder joints in an area can be vastly increased, but also as the acquisition of solder joint images becomes increasingly complex with increase in board density. In cases where leadless SMCs are used, the normally conspicuous joints are now located under the surface mounted components (SMCs); this making it virtually impossible to acquire solder joint images with normal optical approaches (i.e. using conventional illumination) and thus has necessitated the use of specially designed X-ray imaging systems [Pound 1988b] [Buckly 1990d].

Apart from the apparent differences exhibited in the image acquisition subsystems of a typical AOI system and an X-ray inspection system, once the image has been formed, the subsequent image processing and analysis methodologies involved are rather similar. After all, the image formed by passing X-rays through a PCB can be converted into a standard optical image either with an X-ray sensitive videcon camera, or, much more cheaply, with a fluoroscopic system which converts the X-ray signals into light intensity signals that can drive a standard optical videcon camera [Pound 1988b]. Thus it is reasonable to widen the concept of AOI systems to include inspection systems using X-ray imaging methods.

Due to the 3-D nature of the solder joints, subsequent image processing algorithms will require an ability to extract information concerning the 3-dimensional geometrical shape and size of the joints. Therefore, for the purpose of joint inspection, various features relating to the joint surface are frequently extracted and compared with those representing (or describing) good joints [Driels and Lee 1988] [Besl et al 1985] [Bartlett et al 1988] [Keller 1988] [Davy 1988]. Often, the selection of sets of good features has great prominence to the successful fulfilment of inspection requirements.

An immediate goal of inspection would naturally be to detect flaws such as insufficient or excess solder, poor wetting, missing lead. However, the trend is

towards integrated PCB manufacturing, and thus inspection systems are increasingly required not only to simply mark a joint as good or bad, but also to be able to classify the flaws and to identify probable causes of the defects; this information will be utilised for the purpose of process control or adjustment [Driels and Lee 1988]. Taking into account these two aspects of the goal of solder joint inspection, the task of SJI can be decomposed into two subtasks, namely,

- 1) **Inspection of individual BEs (i.e. solder joints, fiducial marks):** This represents the most fundamental part of SJI tasks; features are selected, extracted and compared with certain criteria, and decisions are reached concerning the quality of solder joints. (In fact, this represents almost all of the tasks associated with stand-alone inspection stations).
- 2) **Global examination of the board being inspected:** This is essentially related to the global analysis of the results reached at the BE-level inspection (i.e. the first subtask). This analysis may involve, for example, a classification of the defects, generation of histograms of the global distribution of solder joints (or of solder joint defects), generation of information about solder joint defects peculiar to a certain type of components of certain areas of the board, prediction of probable causes of certain class of defects, etc. This type of global information about the solder joints on a board is of great importance to the optimal adjustment and/or control of the processes prior to inspection, in particular, the soldering process [Corey 1990], which could eventually lead to the elimination of the need for 100% solder joint inspection [Mosca 1990]. However, such benefits may be achieved only in a computer integrated manufacturing environment; otherwise it is doubtful as to whether it is worth generating such information in the first place.

In practice, a library of inspection routines for standard electronic components can be set up; solder joints on PCBs using the components can be inspected with these routines [Pound 1988b]. Once this has been done, CAD information about the board design (component type, location, and orientation) can be downloaded to the inspection system, providing a guidance to the selection of appropriate solder joint inspection routines for a certain component. This in turn can speed up the process of system set up and operation.

7.3 AOI Task Decomposition

Generally speaking, the purpose of task-decomposition is to decompose a complicated AOI task into less complex and less interdependent subtasks, so that each subtask can be accomplished with less effort and less dependence on others, whereas the interactions and relationships among them can be examined upon the accomplishment of all or some of the subtasks. By doing this, a clearer picture can be obtained concerning the overall requirements of the general task, and specific requirements of individual subtasks. Here, “requirements” refer to the information and the necessary algorithms needed by each task and/or subtask.

As the central part of a PCB inspection task lies in the processing and analysing of images of the boards, the decomposition of an AOI task implies the decomposition of the process of image analysis. Furthermore, in order to decompose the overall task of image analysis, it is necessary in the first place to decompose the image, i.e. to represent the whole image in terms of a set of sub-images. Each sub-image can then be treated as a single simpler image and be analysed accordingly. Then the task of analysing the original image can be accomplished by analysing individual sub-images and examining the relationships among them.

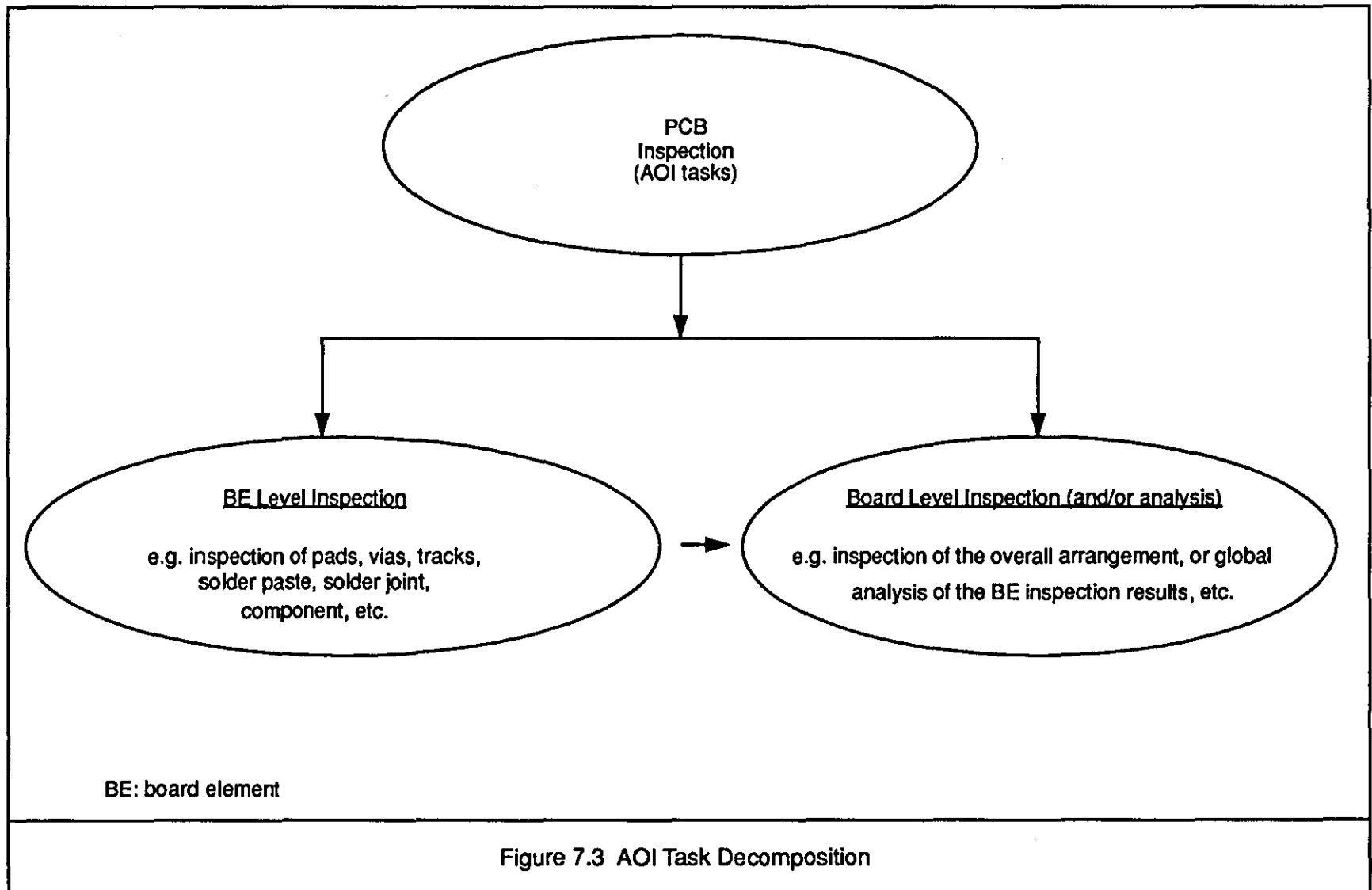
In the context of PCB inspection, the decomposition of the board image, i.e. the selection of the proper sub-images, can be done relatively easily. This is

mainly due to the fact that the board image contains natural sub-images (of the BEs) which can be extracted using certain segmentation techniques. Moreover, each of the sub-images is easily interpretable and physically meaningful, as each of them represents a certain BE of the board.

7.3.1 Decomposing General AOI Tasks

The hierarchical nature of PCB inspection processes is related to the physical construction of PCB products. Thus it is also related to PCB design and manufacture processes and mirrored in the information stored in and/or required by CAD/CAM systems. The author believes that complicated AOI tasks can be simplified, to a certain extent, by means of decomposition, which will facilitate the re-utilisation and sharing of available information by all systems. By summarizing the discussions presented in previous sections, we can see that, in the context of AOI applications in the arena of PCB inspection (PI, SPAI, CPI and SJI), an AOI task can be decomposed into two levels (Figure 7.3), namely,

- 1) *Board element (BE) inspection*, which is concerned with inspecting (i.e. locating, identifying and taking measurement) of all BEs which make up the final PCB product (e.g. a master film, a layer, a bare-board, a populated board, a soldered board, etc.)
- 2) *Board level inspection (and/or analysis)*, which is concerned with the examination and verification of the overall arrangement of these BEs and the relationships among them (i.e. the electrical connectivity, the relative geometrical distribution, etc.), and/or global analysis of features of the board being inspected, based on the inspection results reached during BE level inspection. To facilitate system integration, functions are increasingly required of this level inspection (and/or global analysis) that can provide global inspection information which can be utilized by succeeding proc-



esses and systems, or for the purpose of process control via information feedback.

Towards decomposing the AOI tasks, some general observations have been made by the author, which to a certain extent have formed the basis of the foregoing discussions; these being:

- a) A populated PCB (after component placement and soldering) can be viewed as being made of a bare board and a wide variety of pre-defined components. The definition of these components can be found in the Part Library (database) of most proprietary CAD systems.
- b) A complicated, typically multilayer, bare-board is made up of a set of manufactured (finished and/or semi-finished) individual layers (conducting layers and insulating layers) through various bonding processes.
- c) Each layer itself is made of a group of entities referred to as BEs which can be anything that is essential to the realization of the intended electronic functionality, e.g. printed components, footprints, pads, vias, tracks, fiducial marks, etc.

With this proposed decomposition, reference data from CAD/CAM stations, where available, can be utilized accordingly at various levels to assist in the execution of AOI tasks. For example, information about a particular type of BE can be extracted from the "information pool" and utilized at an appropriate stage of BE inspection, whereas the information about the overall board structure will be used for the overall board level inspection (or analysis). Likewise, information about certain BEs present on the board, or information about an entire board can be extracted and analysed, and be made available to and usable by other processes as a result of the task decomposition.

To the author's belief, the identification and recognition of the inherent hierarchy of both the physical construction of PCBs and their inspection processes are of great significance. Firstly, the hierarchical nature of the PCB physical construction is reflective of the process of PCB manufacturing (see section 3.2.2 and Figure 3.2); therefore information about individual constructs (layers and/or BEs) of a PCB can be of great importance to the control (or regulation) of the corresponding processes (e.g. used for both local and global statistical process control). This information should become readily available with the decomposition of a general PCB inspection task into several subtasks of inspecting constructs of a board.

Secondly, this decomposition of the inspection process is consistent with the way in which design information is represented in typical PCB CAD systems [Myers 1988], as typically a CAD database would store design information about a product at various levels. For example, there is information that describes the overall structure of the product (e.g. list of layers which make up the final product and the overall technology used to build the board), information that describes the structure of individual layers (e.g. list of BEs for a given layer and the relationships -- geometrical and electrical -- among them), and information that describes in detail each and every BE (e.g. pads, vias, tracks, fiducial marks, etc.). With such a decomposition of the inspection process and an understanding of the corresponding CAD information representation, it is reasonable therefore to expect a more flexible and efficient way of reusing CAD information for supporting AOI applications.

Thirdly, this approach to decomposing the inspection process is of help to the design and implementation, in a modular fashion, of AOI algorithms and such a decomposition is mirrored in the hierarchical reference architecture to be introduced in the following section.

7.4 Hierarchical Design of AOI Algorithms

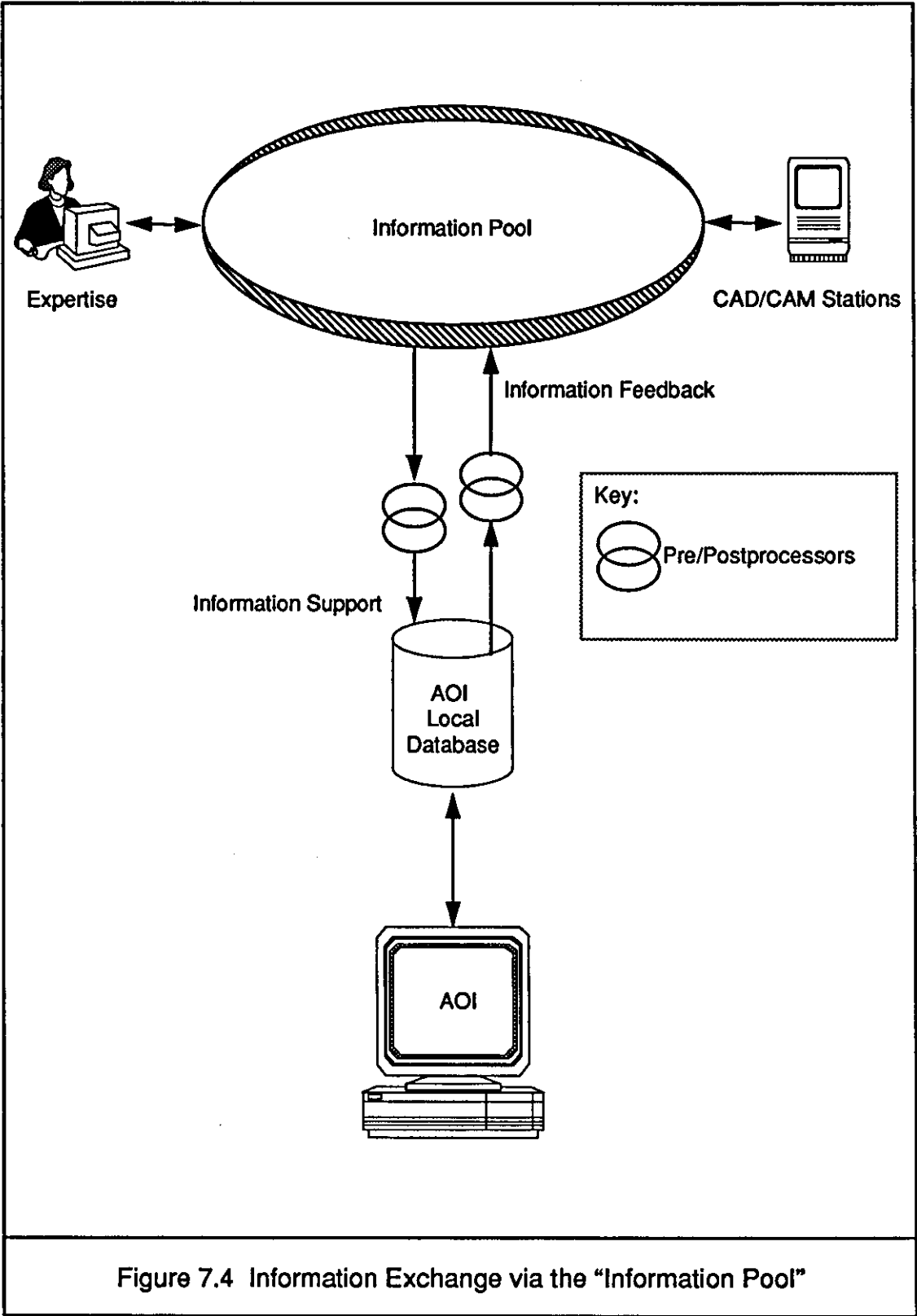
In view of the hierarchical nature of both the PCB inspection process and the available information about PCBs (as stored in CAD/CAM stations), a five-level reference architecture is constructed for the design and implementation of AOI algorithms. The basis for constructing such a reference architecture is the view held by the author that novel approaches to designing and implementing AOI algorithms have to be taken if AOI systems are (i) to be flexibly integrated within a broader CIM system, (ii) to benefit from achieving information supported manufacture of PCBs (ISM/PCB), and (iii) to produce useful information that can be utilised by other processes.

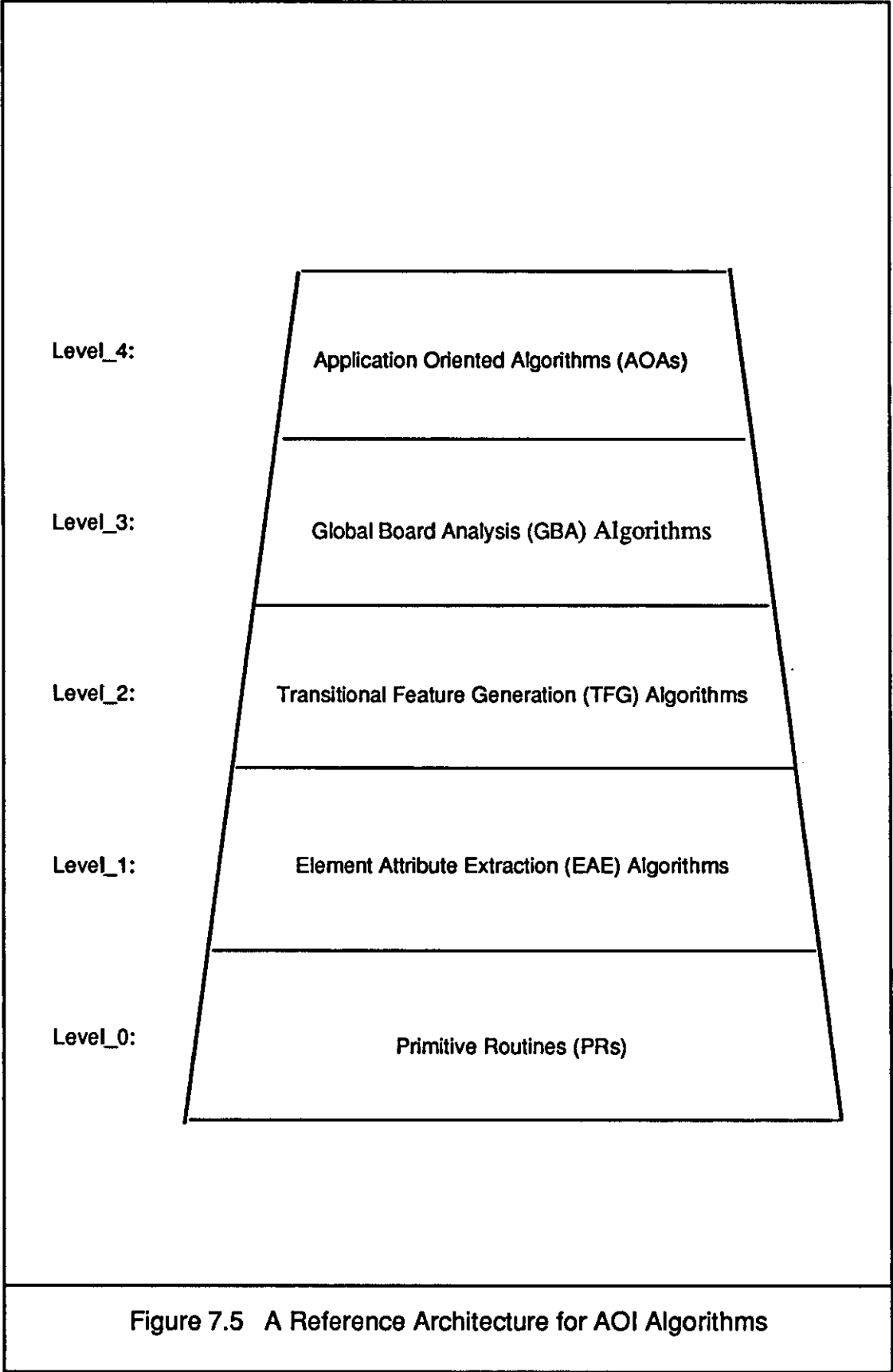
Thus the new approach demanded and adopted should not only facilitate the exchange of information between AOI systems and other computerised systems, but also facilitate the use of available design and/or manufacture information in assisting in the execution of an AOI task. Therefore, the author believes that it is desirable for the reference architecture to reflect the interior hierarchy of both the inspection process and the information used to describe the product.

It should be pointed out however that in such an ISM/PCB scenario, the AOI system is both a consumer and a producer of information. In other words, the AOI system gets information support from the “information pool” on one hand (i.e. use of information provided by other CAD/CAM stations), and generates and provides information support to other processes on the other hand (also via the “information pool”). See Figure 7.4.

7.4.1 The proposed Hierarchical Reference Architecture

The proposed five-level hierarchical reference architecture is illustrated in Figure 7.5. The definition of each “level” is based on the functionality of the algorithms within it; therefore, each level represents a distinct stage of image processing





involved in the execution of a given AOI task. For example, level_0 algorithms (i.e. the primitive routines) work at the bottom level and represent operations on images in the form of pixel data (i.e. in an array data structure), whereas level_4 algorithms (i.e. the application algorithms) generate overall application-oriented inspection features of the panel under inspection such as a report on the defects detected, or a report on the total numbers of a particular type of defects found.

A. Level_0: The Primitive Routines (PRs)

This is the bottom level of AOI algorithms, which carries out a range of operations on images in the form of pixel-matrixes and includes most of the fundamental routines provided with typical general purpose machine vision systems. Generally, the functionality of these level_0 algorithms is required to accomplish basic management and some necessary early processing of the image, that is to prepare the image for the next processing stage (i.e. feature extraction). In addition, the functionality will also be required to control and configure the vision system hardware and other related physical equipment such as a video monitor. Ideally, illumination control and camera setting should also be included at this level. The functions of level_0 algorithms are categorised and listed as follows (see Figure 7.6),

- a) Control and configuration: For example, to set up video monitor display format, select video signal (e.g. from a CCD camera) input channel, select frame buffers to be output to video monitor, set up processing window parameters, etc.
- b) Image I/O and manipulation: For example, to grab/delete an image, to copy image between frame buffers, to store/load an image to/from a disk file.
- c) Preprocessing of the raw image data: For example, to apply matrix-based pixel-to-pixel (PTP) transformation on the image

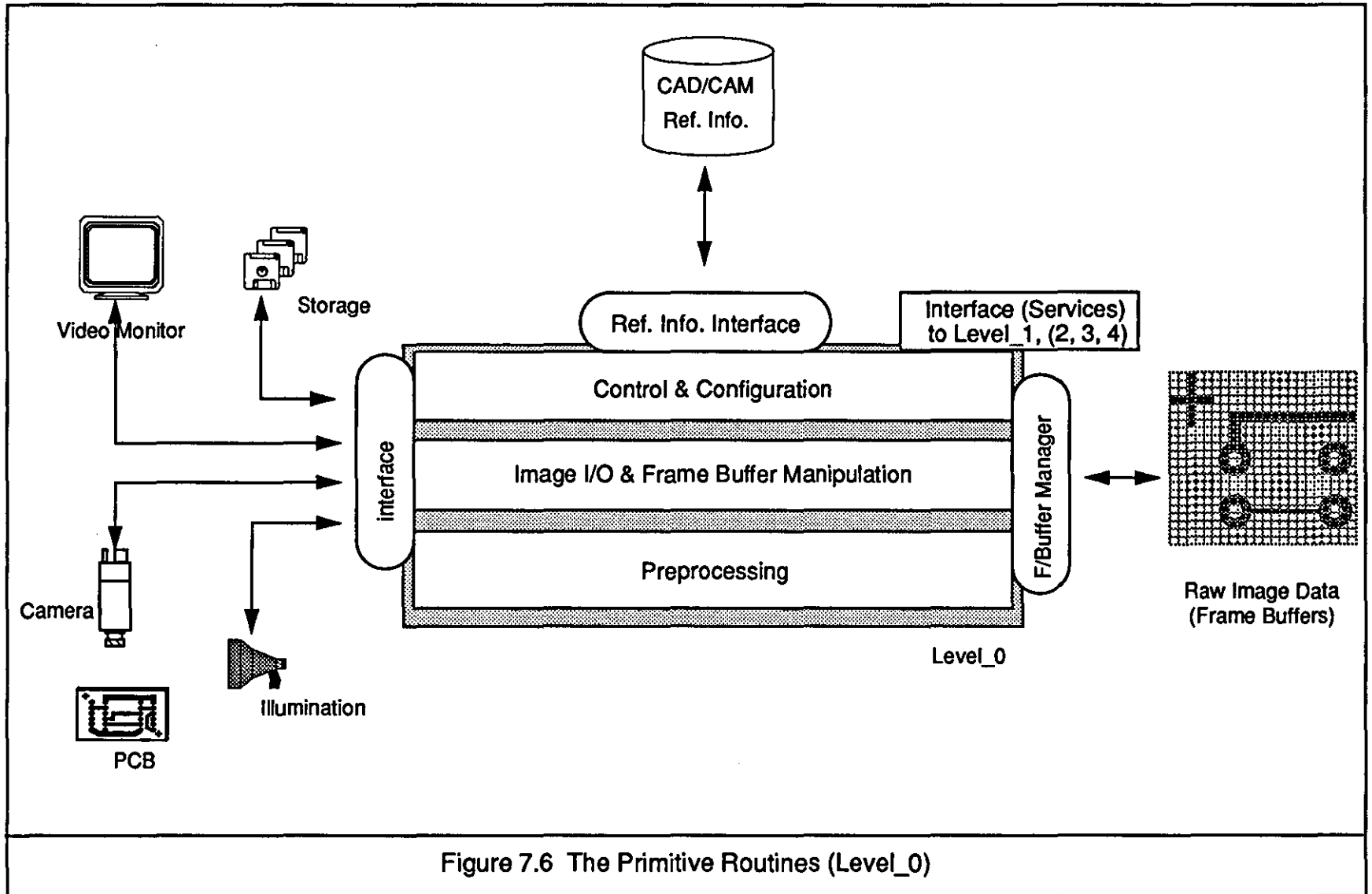


Figure 7.6 The Primitive Routines (Level_0)

data (e.g. edge operators, low/high pass filter, etc.).

It is necessary to point out that algorithms at this level generally deal with the *raw image data (RID)*, with PTP transformation as the typical characteristics. Images being processed at this level are normally confined to the form arrays of pixels, and so is the output data structure.

B. Level_1: Element Attribute Extraction (EAE) Algorithms

This is the second stage of image processing whereby operations are performed on the early-processed images so as to extract important local features about each and every object of interest present in the scene (typically BEs as defined in previous sections). CAD/CAM reference data can be used at this level to generate window parameters for locating various BEs out of the whole image, and to guide the selection of appropriate algorithms for the extraction of certain local features of the BEs. The idea is to make use of *a priori* knowledge available from CAD/CAM information about a particular BE so as to guide the extraction of appropriate local features which can be used in object identification at a later stage. The functions of level_1 algorithms can be categorised and listed as follows (Figure 7.7),

- a) Statistical operations: For example, calculation of an image histogram, image profiles along a given direction, minimum/maximum/mean intensity values, etc.
- b) Image segmentation: For example, image thresholding, template matching, edge detection, and image coding (i.e. chain codes, run-length codes, etc.).
- c) Local feature extraction: For example, calculation of features such as number of corner points detected, the distances between neighbouring corner points, area, centroid, perimeter, roundness, various moments, etc.

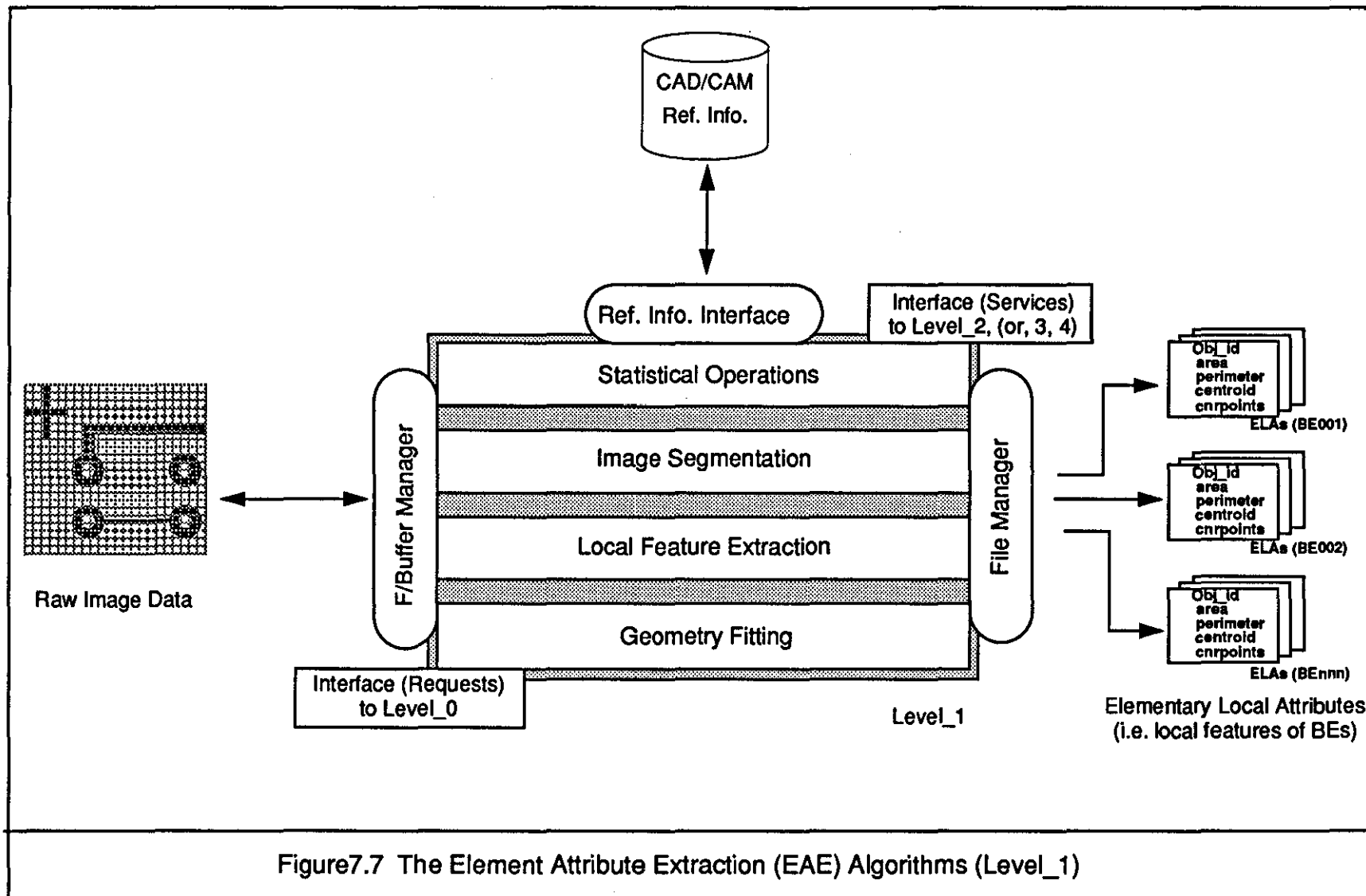


Figure7.7 The Element Attribute Extraction (EAE) Algorithms (Level_1)

- d) **Geometry fitting:** For example, line/arc/circle/polygon fitting, etc.

The local features are of great importance when locating and identifying individual BEs, and they are used mainly for the purpose of "*BE inspection*". As stated in the previous section, BE inspection is just one of the subtasks that is decomposed from a general AOI task (section 7.2). After being processed by the level_1 algorithms, the image is now represented by sets of local features of BEs. These local features will be termed as *elementary local attributes (ELAs)*. The selection of appropriate ELAs for a particular application is application dependent, and so are the EAE algorithms used. For example, in the case of panel inspection, features like area, perimeter, centroid, number of corners, etc. may be sufficient for locating and identifying simple BEs, whereas in the case of solder joint inspection, features relating to volumetric measurements and surface curvature are typically required [Besl et al 1985] [Driels and Lee 1988].

C. **Level_2: Transitional Feature Generation (TFG) Algorithms**

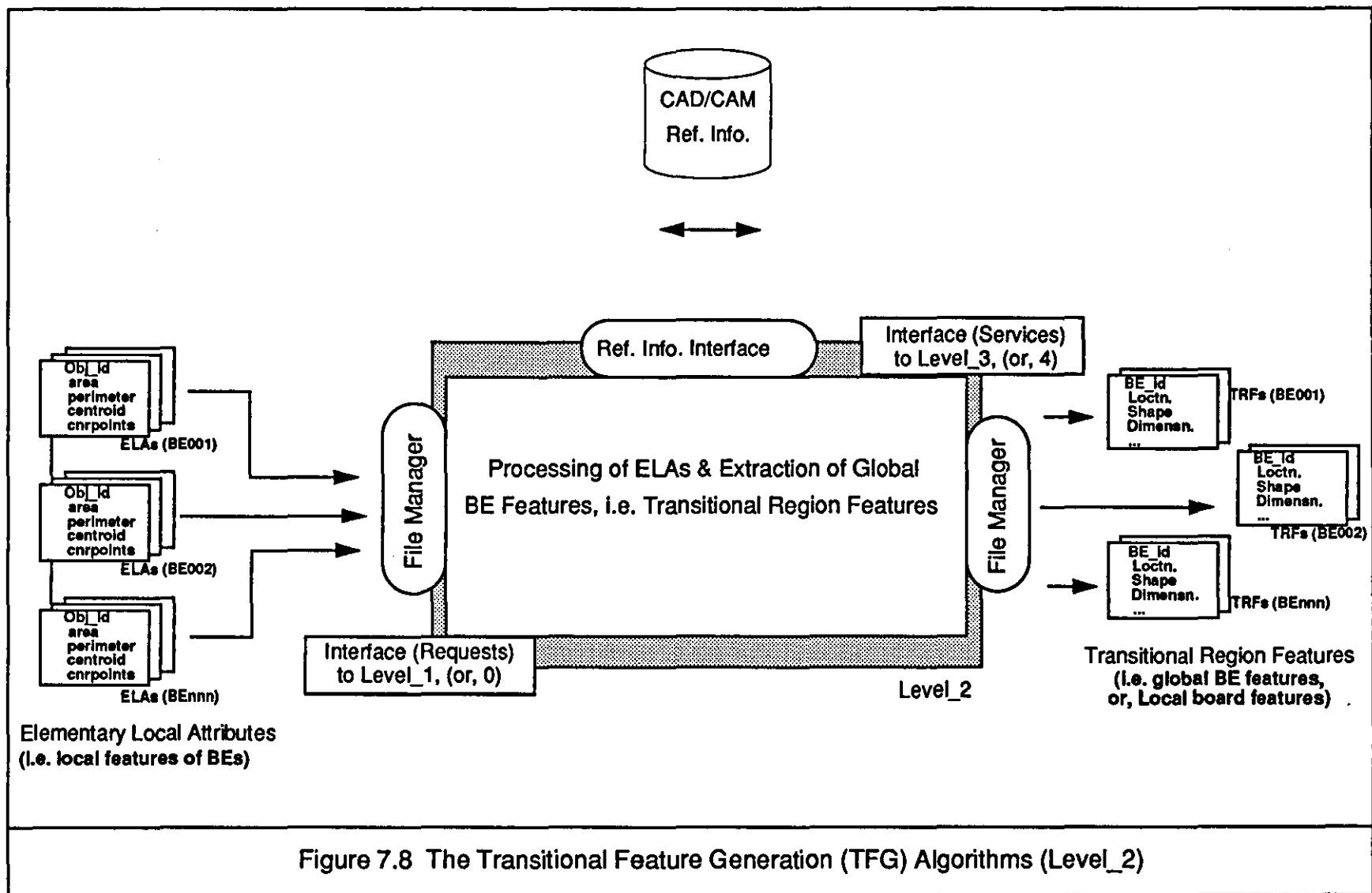
The function of the TFG algorithms is mainly to process the low level feature vectors of individual objects (i.e. the BEs) as returned from the EAEs and to generate more global features about the same object. In other words, TFG algorithms take as input a set of ELAs about a particular BE, perform various operations on the feature data and finally *generate* important global features of the same BE. Sometimes it may be still necessary to make reference to the lower level raw image data so that correct measurement can be made of the global BE features. Some of the frequently used global features of a BE include identity name, location, geometric shape descriptor, connectivity with other BEs (i.e to examine whether a particular BE an isolated entity on the board), etc. Again these features are application dependent.

Thus the functions of the TFG algorithms are related to the processing of the input ELAs and the generation of the output global features of individual BEs, and therefore can be categorised and listed as follows (Figure 7.8),

- a) Identification, localization and inspection of individual BEs,
- b) Connectivity check, and
- c) Generation of global features of the BE.

Following on from previous discussions about AOI task decomposition that completion of the subtasks of "*BE inspection*" is the basis for the execution of the subtask of overall "*board-level inspection*"; therefore, from the viewpoint of board level inspection, global features of individual BEs are at the same time local features of the board, as they characterise only certain local regions of the board. Thus in this sense we can consider the global features of individual BEs (or local features of the board) to be *Transitional Region Features (TRFs)*.

Algorithms should be provided which can reference CAD information and make use of such information in the process of BE inspection. Usually, the CAD reference information utilised at this level is concerned with inspection criteria for BEs, e.g. for checking whether the desired minimum dimensions of the element have been maintained. The reference data can also be used to specify certain expected attributes of the BE under inspection so that reference can be made when the actual attributes are being examined and verified; for example the number of corners and the distance between two neighbouring corner points, or the expected connectivity between this element and others. This type of information sometimes can be utilized to guide and simplify the process of BE identification. Reference to CAD information in this way will be further elaborated on in chapter 8.



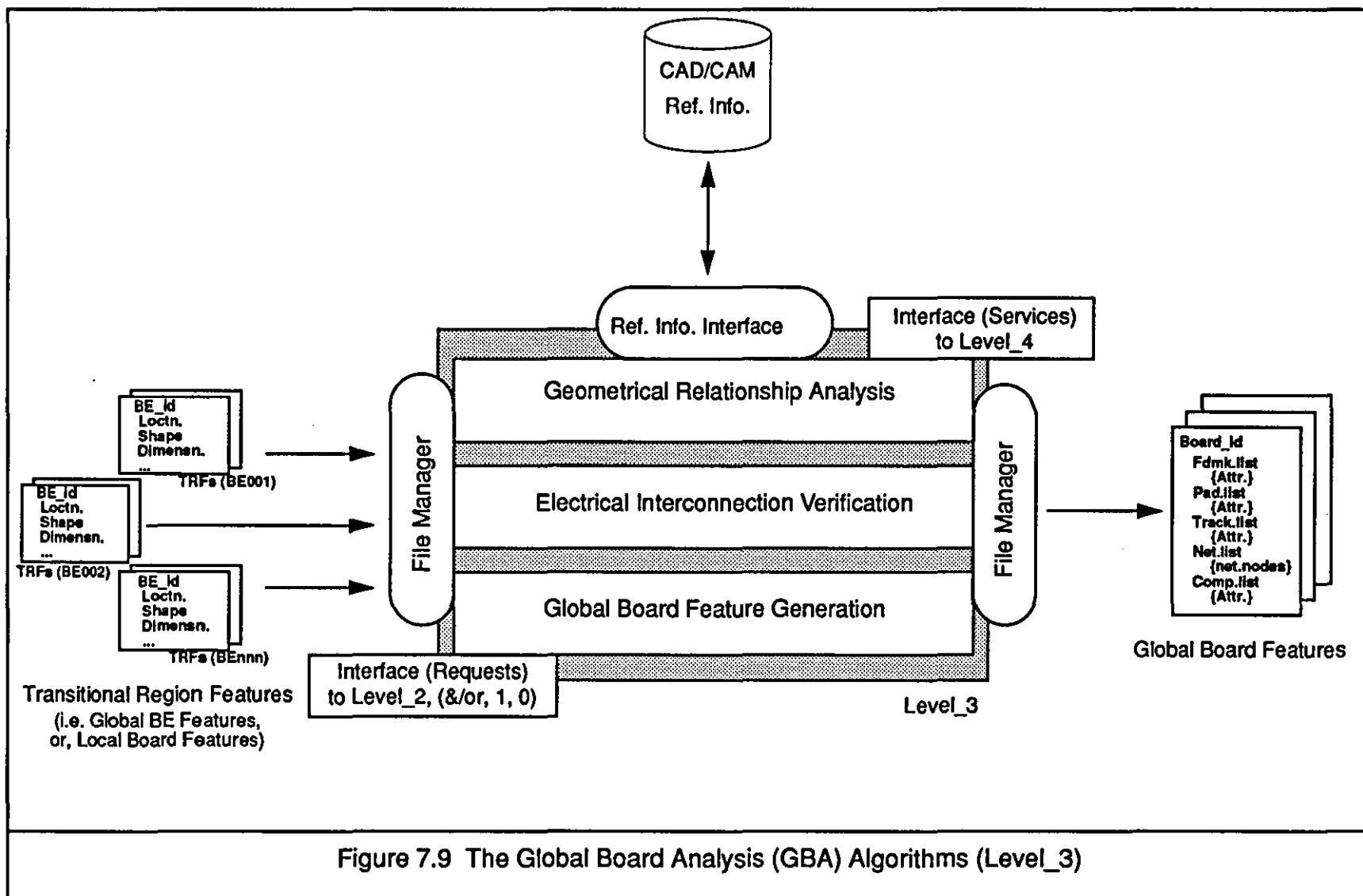
D. Level_3: Global Board Analysis (GBA) Algorithms

Algorithms used for the purpose of global board analysis (GBA) are generally concerned with the board-level inspection; for example, the analysis and inspection of the overall relationships among all the BEs which make up the PCB. Usually the inspection involves processes such as that of checking the actual relative locations between BEs, verifying their interconnection, and so on.

The inspection and analysis performed are based on the local features of the board, namely the transitional region features (TRFs) as defined earlier; therefore GBA algorithms take as input the TRFs, apply appropriate operations on the feature data and then generate global features of the board. The output will be the global features of the board, which generally include lists of each type of BEs (e.g. its name, shape and location), interconnection maps (i.e. the verified netlists), list of defects, etc. These global board features will be given the name of *global board features (GBFs)*. The functions of the GBA algorithms can be categorised and listed as follows (see Figure 7.9),

- a) Processing of local board features to examine the geometrical arrangement of BEs and the electrical interconnections between them,
- b) Morphological analysis to verify electrical interconnections (e.g. minimum allowable track widths, spacing requirements, etc.), and
- c) Generation and formatting of global board features.

Design information from a CAD system and statistical information relating to certain manufacturing variables have roles to play at this stage of board-level inspection. Typically, this CAD/CAM reference data is processed and combined to form the inspection criteria which will be utilized in the process of board-level inspection (i.e. use for feature comparisons). The CAD information to be referenced here is normally that which describes the PCB at board-level, for example the



netlists, a list of components expected to be present on the board, etc. Similarly, information contained in lower level representations (e.g. RID and ELAs) may again be useful -- hence referenced here.

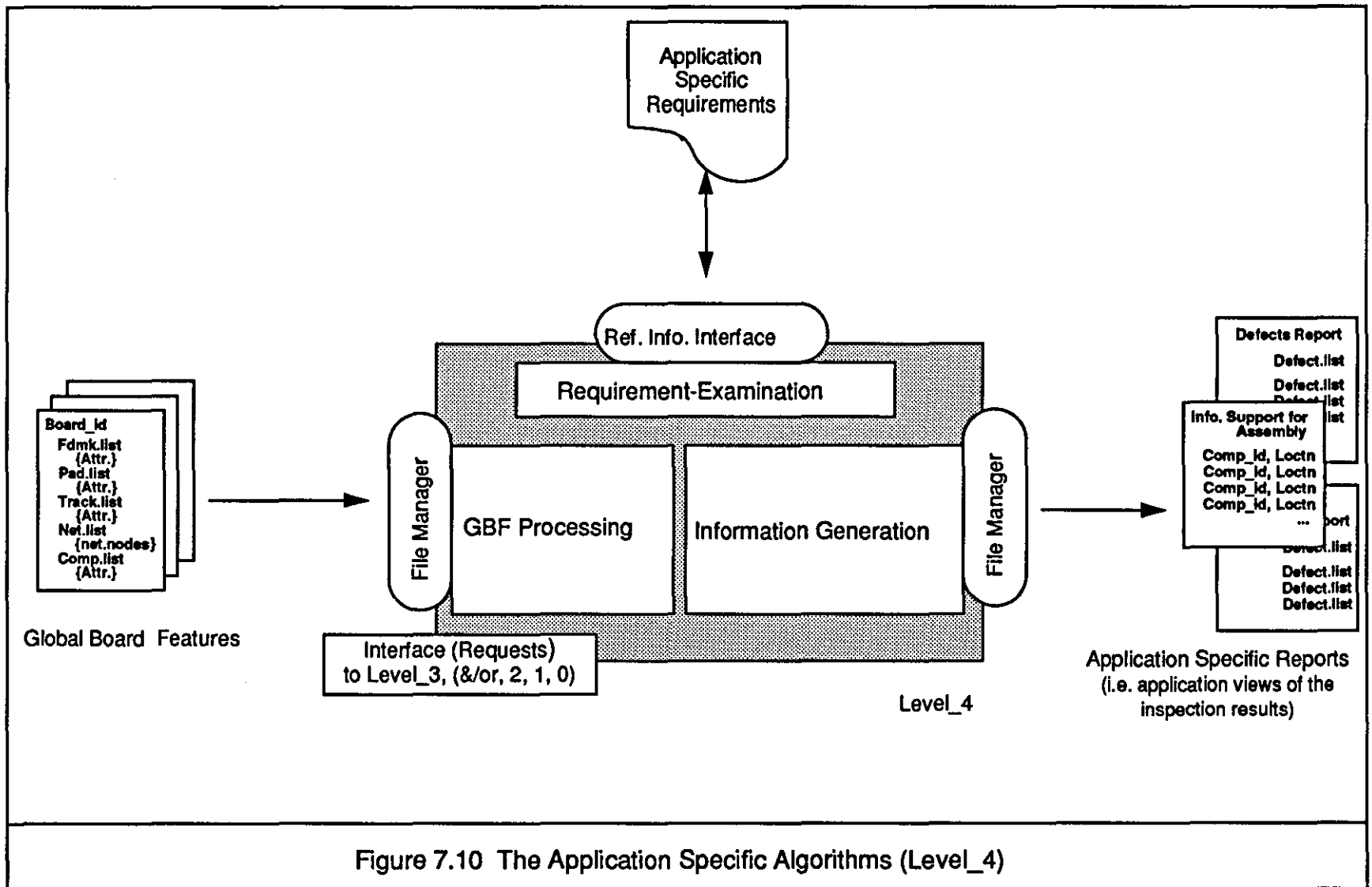
E. Level_4: Application Oriented Algorithms (AOAs)

This is the top level of the hierarchy. The inclusion of this level is in fact based on the observation that the same set of global board features (GBFs) can be catered for different applications in a different format. In other words, the same set of GBFs may be processed to provide different views to different applications. For example, in the case of *panel inspection*, the application of “feedback information to a product model” may require the report in a format which is compatible with the representation of the relevant product model sections, whereas an application requiring “defect reports” may adopt the format of a list, etc. The alternative is to report the inspection result in a fixed neutral format and different views of the information are achieved via pre/post-processing of the neutral information. In this case the pre- and post-processors are considered as part of the AOAs.

The functionality of the AOAs therefore is to analyse the GBFs and generate inspection reports for differing applications (processes) and/or for information feedback to the global product model. Thus we can categorise the AOAs as follows (see Figure 7.10),

- a) Examination of the particulars of given applications and their requirements.
- b) Accessing and processing of GBFs (and even TRFs, ELAs, and RID), and
- c) Generating required application-oriented report.

With this approach, sets of library routines may be provided for each defined class of local applications. Thus the required report (reformatted inspection



result) used for a certain class of local applications can be produced by a certain set of routines.

7.5 The Data Flow Hierarchy

Before concluding this chapter, the author believes that it is helpful to look at the image data hierarchy as it is being sequentially processed by levels of the AOI algorithms. This can help to further the understanding of the “transforming” nature of the AOI system algorithms, and of the way by which these “transformations” transfer the low level image data arrays into high level concise messages (e.g. application oriented reports).

As discussed in the previous section, the hierarchical reference architecture conceptually divides the functionality of AOI algorithms into five levels. However, from a more general viewpoint, the functionality of any vision systems (of which AOI is a particular case) can be viewed as performing a sequence of “transformations” on the original image data and, at the end of the sequence, to generate in a desirable format some sort of descriptions about the scene. Thus the reference architecture in fact breaks up the sequence of transformations and arranges them into five distinct stages (see Figure 7.5). At each stage, a set of algorithms perform certain transformations (operations) on the input data.

Accordingly, a five-level data hierarchy can thus be anticipated, namely, the raw image data (RID), the elementary local attributes (ELAs), the transitional region features (TRFs), the global board features (GBFs), and the application-specific reports (ASRs).

A. The Raw Image Data (RID)

Raw image data is considered to be in the form of matrix of discrete digital values (representing the grey levels of the image), with information content rep-

resented in an implicit manner. In other words, without appropriate further processing of these digital values (i.e. extracting of explicit features of the object that the image represents), it is difficult to infer any useful information concerning the original scene. It is argued that this is because that data element itself can not convey any unambiguous information [Beyon 1990]. The goal of image processing and understanding is to extract appropriate information from the scene encoded in the matrix of pixel values; therefore the raw image data is considered here as being at the most primitive *"Data level"*.

B. The Elementary Local Attributes (ELAs) and the Transitional Region Features (TRFs)

As defined in section 7.4, the ELAs are local features of individual BEs (e.g. number of corner points, distance between two adjacent corner points, perimeter), whereas the TRFs are the global features of the same BE (e.g. its location, name and shape identity, status of connectedness, etc.), which are generated through processing (e.g. combining and/or synthesising) fragments of ELAs.

The TRFs are also local features of the board, in the sense that they describe only a local part of the board, i.e. a BE such as a pad, a solder paste pattern, or a component. As far as the board is concerned, these ELAs and TRFs all provide partial information about it, each ELA or TRF is just one piece or fragment of "information". They are termed pieces or fragment of "information" in the sense that they provide us with at least partial knowledge about the board. For example, an ELA like PERIMETER indicates that the BE concerned "has a perimeter of the value specified by PERIMETER". Similarly, a TRF like IDENTITY would at least indicate that the board consists of that BE specified by its IDENTITY, whereas a TRF of LOCATION would tell where that identified BE is located. In this sense, ELAs and TRFs are considered to be at the *"information level"*.

It has been seen that each piece of information can only provide partial or incomplete knowledge about the whole board. However, an appropriate aggregation of these pieces of information will be sufficient to describe the board. The processing of all these information pieces should give a clear and unambiguous “message” concerning the board under inspection.

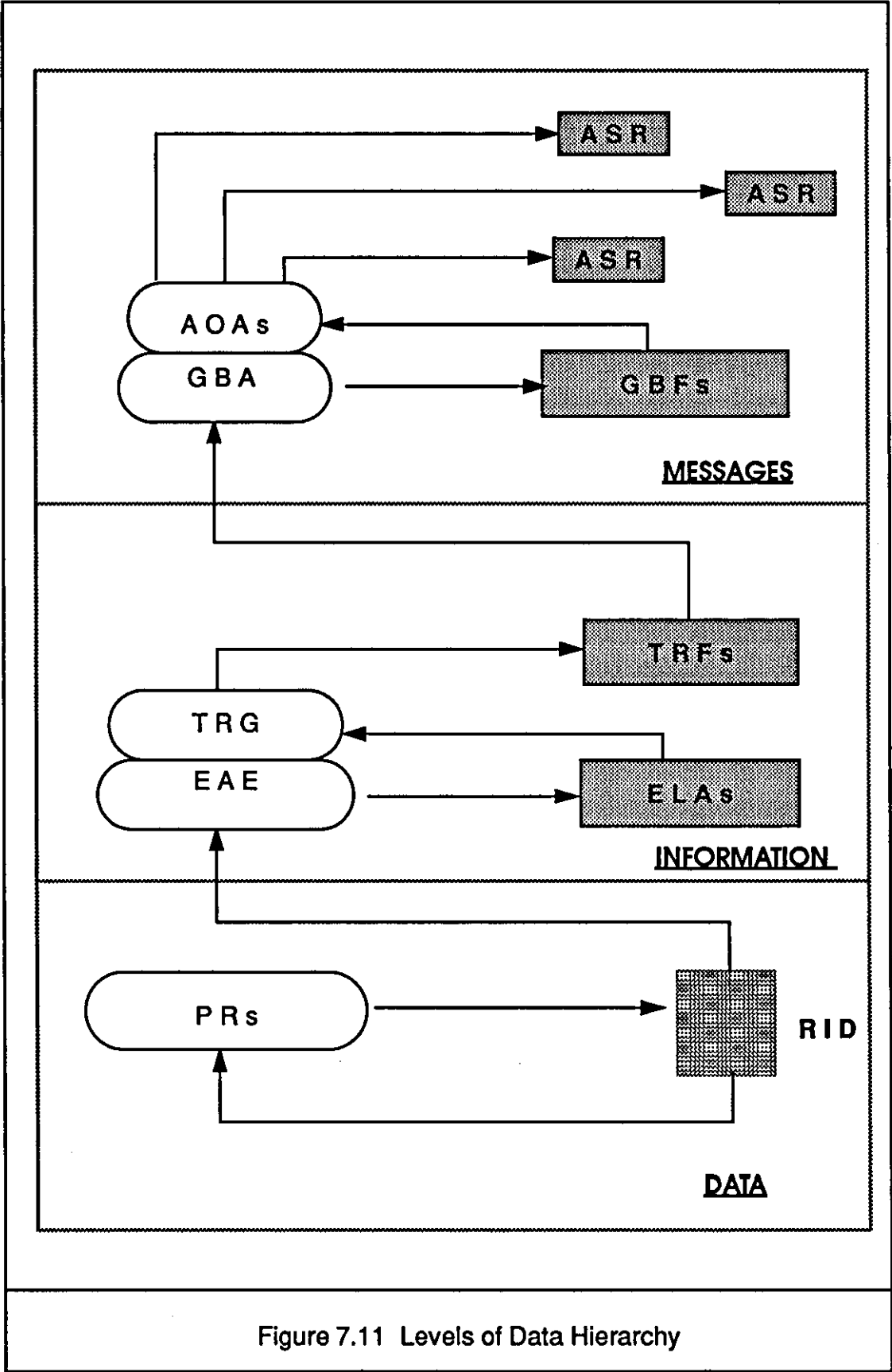
C. Global Board Features (GBFs) and Application-Specific Reports (ASRs)

The GBFs are those features that describe the board from a global point of view, i.e. they are the global features of the entire board. These features are the result of analysing all the local features of the board; therefore they provide more complete information about the whole board being inspected.

The ASRs are more or less reformulations of GBFs. If we regard the GBFs as forming a global view of the information about the board being inspected, then the various application specific reports can be considered to provide the user (application) views of the same information. Thus we can see that the GBFs and the ASRs are all based on the information fragments provided by the TRFs and ELAs, and that clearer and more complete messages about the board can be gained from the GBFs and the ASRs. Therefore the GBFs and the ASRs are considered to be at the top level of the data hierarchy, i.e. the “*message level*”.

Thus along with the process of performing an AOI task (involving image processing and understanding), we have three levels in representing the information contained in or extracted from an image (see Figure 7.11), namely

- 1) **Data** level representation: This includes the raw image data (RID) which provides most of the “*raw material*” on which the execution of a given inspection task begins.
- 2) **Information** level representation: This includes the low level fea-



tures (ELAs) and the medium level features (TRFs), which can be considered to be the “*work-in-process*”, or the “*meta-products*” of a vision system. The collection of these information fragments are to be further processed so as to form certain general messages concerning the entire board being inspected.

- 3) **Message** level representation: This includes the Global board features (GBFs) and the application specific reports (ASRs), which can be viewed as the “*final product*” of an AOI system. The provision of ASRs enables the AOI generated information to be shared and utilized by other succeeding processes or systems.

7.6 Summary

In this chapter, the characteristics of typical AOI applications have been considered where this has revealed that the execution of an AOI task can be divided into two stages. At the first stage, the individual BEs are inspected; algorithms appropriate for inspecting certain types of BEs (e.g. isolated round/square pads, interconnected pads and tracks, solder paste lumps, solder joints, surface mount components such as small outline transistor, chip capacitors, etc.) can be devised, implemented and used at this stage. At the second stage, global features of the board are examined (or extracted) based on the set of local board features generated at stage one. This serves to inspect the overall arrangement of those BEs, the electrical and geometrical relationships among them, or to generate feedback information to other processes and/or systems. A generalization of this two-step approach to general automatic PCB inspection has led to the notion of AOI task decomposition, i.e. decomposing an AOI task into *board element level inspection* and *board level inspection (and/or analysis)* as shown in Figure 7.3.

As a result of characterizing and decomposing AOI tasks, a hierarchical reference architecture has been proposed which can be used to guide the structuring of the design and implementation of AOI algorithms. In this model the overall functionality of an AOI system is divided into five levels, each of which represents one stage of the processing of PCB images carried out within the AOI system (see Figure 7.5). The functionality of the algorithms at each level has been identified and specified, and the possible applications of CAD/CAM reference information have been discussed. However, it should be pointed out that in specifying the functions of different classes of algorithms at each level, focus has been mainly on applications of the class “panel inspection”; though algorithms for other applications have been considered as fully as possible.

In next chapter, a description is presented of the author’s partial implementation of the product model based approach to integration proposed in Chapter six and the suggested AOI reference architecture introduced in this chapter.

Chapter 8

Description of a Prototype

Information Supported Machine Vision System

8.1 Introduction

This chapter provides a description of the proof of concept prototype information supported machine vision system produced by the author. The system was configured to illustrate a novel method of achieving AOI in PCB manufacture. Essentially the work reported here is the author's partial implementation of the "*product model based approach*" to achieving flexible integration proposed in Chapter 6, where the AOI system conforms to the "*AOI system reference architecture*" proposed in Chapter 7. In addition, a description is included of the software used to establish and maintain the communication link between the Matrox vision system and the AdeptOne robot manipulator system, as well as of routines used to calibrate the integrated vision-robot couple.

8.2 Description of the Software Platform SPADAT

In this section, a brief description is given of the Software Platform for AOI-algorithm Development And Test (SPADAT). This software platform was designed and used by the author for the purpose of AOI algorithm design, development, implementation and test. The main use of SPADAT is thus to facilitate the development and test of AOI algorithms in a gradual and modular fashion, and to

provide a test-bed for the examination and characterization of interactions between various AOI algorithms within the same or different layers of the implemented AOI system reference architecture, as well as the interactions between AOI and external information sources, i.e. information resources resident in other computerised stations (e.g. the CAD/CAM stations). Furthermore, many ideas and concepts relating to the design and implementation of the prototype information supported AOI system for PCB inspection have been developed and tested on this platform. Most of the algorithms introduced in the following sections were created on this platform, and can be run and tested in a interactive manner. Thus **SPADAT** provides flexibility when building AOI systems.

SPADAT itself has been built up in a modular manner, consisting basically of the following five modules (see Figure 8.1), namely,

- 1) The interpreter module (**INTERPRT**),
- 2) The function-calling module (**ACTIONS**),
- 3) The algorithm library module (**ALGLIB**),
- 4) The new algorithm development module (**DEVELOP**), and
- 5) The error handling module (**ERRHDLR**)

The software **SPADAT** is written in the programming language C and compiled, linked and maintained using the Microsoft **MAKE** utility: the relationship between these modules is illustrated by Figure 8.1, and will be further explained in the following paragraphs along with an explanation of the operating principles involved. The separation of the **DEVELOP** module (which enables algorithm development) from the **ALGLIB** module (i.e. library of tested algorithms) offers simplicity during program generation (i.e. compile, link and debug): this taking advantage of the **MAKE** utility.

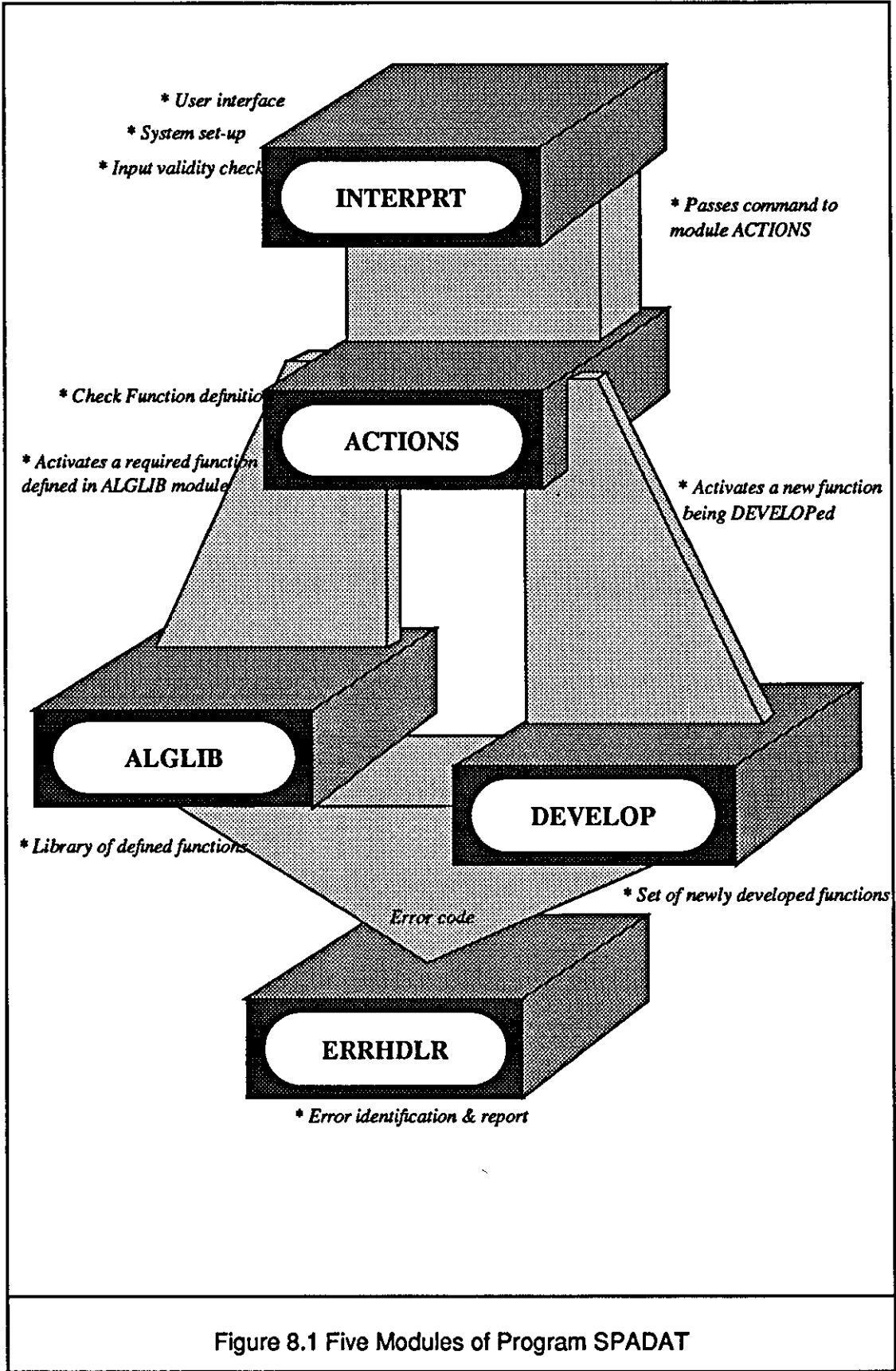


Figure 8.1 Five Modules of Program SPADAT

The **INTERPRT** module acts as a command line input interpreter which sets up the program running environment and user interface window on the screen, initializes the Matrox vision system, and then waits for and accepts user input from the keyboard. The input expected is a string of characters corresponding to one of the AOI algorithms (functions) either already defined in the **ALGLIB** module or being newly developed in the **DEVELOP** module (for new algorithms). A syntax, validity check is performed by the **INTERPRT** to check illegal characters or to skip non-functional inputs (e.g. a carriage-return, etc.).

Having performed these functions, the **INTERPRT** module passes the input command to the **ACTIONS** module where a comparison is made between the input and a list of functions defined in the **ALGLIB**; if a match is achieved, the desired image processing or product inspection function is activated (i.e. is called). If the required function is not defined by routines available within the **ALGLIB** module (i.e. no match is made between the input and the list), the input command is passed to the **DEVELOP** module. If it has been defined there, the newly defined function is activated and tested; otherwise an "Unknown function command" error code is generated and passed to the **ERRHDLR** module where the error is reported to the user interface window.

When a new algorithm has been developed and tested satisfactorily, it can then be transferred to the **ALGLIB** module; therefore the **DEVELOP** module is kept at a minimum size (i.e. only includes new functions). Modification of algorithms in the **ALGLIB** module is possible, but should be kept to a minimum, as such modifications slow down the processes of the Microsoft **MAKE** operations.

The software development platform had a very important role to play in assisting the author to develop and test new ideas and algorithms. When developing vision algorithms, it was often necessary to run the vision system in an interactive manner and have control via the platform over the execution of certain aspects of the prototype image processing software. Frequently, common software utilities

such as image input and output, frame store manipulation, histogram calculation and display, image thresholding, boundary chain code generation and analysis, etc. are required when testing a new algorithm; such as that for extracting certain local features of the board under inspection. Moreover, changes of utility parameters are also often needed so as to test the performance of a new algorithm. In the absence of a development platform such as **SPADAT**, there would be a need for those development functions (or utilities) to be included in the new program code; this leading to duplicated work, making the new program bulky and difficult to handle (i.e. slowing down the compilation and linking, complicating the debugging of program), and at worse, may alter characteristics of the program execution (e.g. in terms of execution times, etc.).

A flow chart illustrating the important operational features of the **INTERPRET** and **ACTIONS** modules, as well as the interactions between them and other modules, is given in Figure 8.2. Since both the **ALGLIB** module and **DEVELOP** module are a collection of different types of algorithm, the combined operation of these two modules will depend on the choice of algorithms activated -- hence this is not detailed in the figure. The **ERRHDLR** module is relatively simple and is mainly concerned with interpreting the error codes and sending error messages to the user interface window.

8.3 Towards Achieving Flexible Information Support for AOI

As demonstrated in Chapter 5, one means of providing information support for AOI tasks can be realized by linking specific CAD and AOI systems through post-processing the CAD information so that it can be reutilised within the operational context of the AOI system. The features of such a customised software link for the purpose of information extraction and representation will be highly dependent on properties of the two end systems. Although there will be cases where such a one-off bespoke approach to integration can be appropriate or even effective,

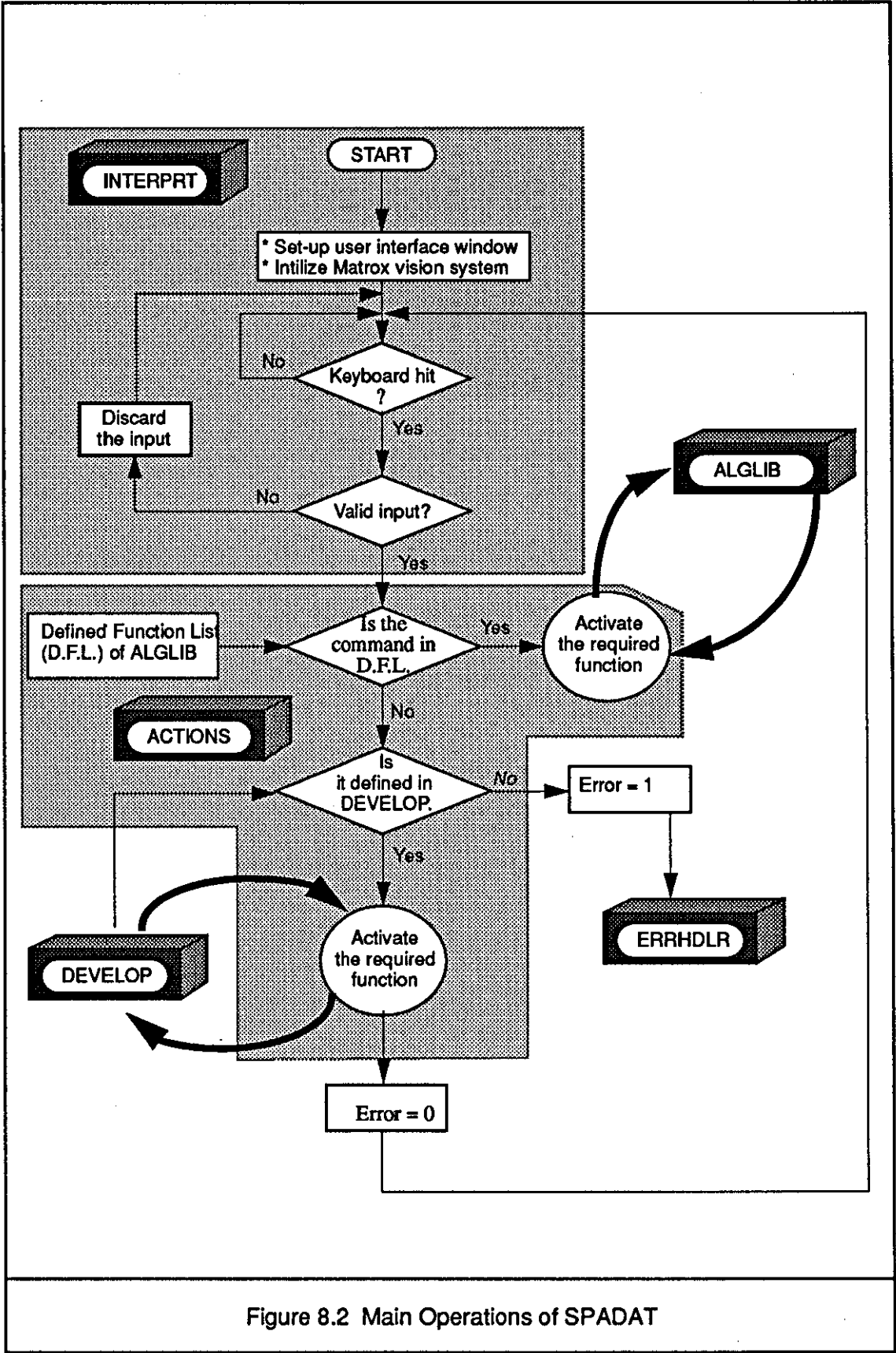


Figure 8.2 Main Operations of SPADAT

the integrated system couple based on this type of bespoke approach will generally be inflexible, in the sense that changes made to the AOI system and/or its information requirements (e.g. as a result of system upgrading or change of application) will not be facilitated; this potentially leading to major software re-writes and hence significant penalties in terms of financial cost and development time.

In this section, details are given of the proof-of-concept implementation of the “product model based approach” to integration presented in chapter 6. The implementation is intended to demonstrate how such an approach can be used to achieve flexible information support for AOI applications with inherent flexibility which can result in enabling change and allowing much wider-scope integration project to be contemplated. As previously mentioned in Chapter 6, the underlying concept is one of using product models as “information pools” in which information concerning the realization of predefined products (e.g. a PCB) is stored. More specifically, physical product models (as defined in Chapter 6) are employed to store the actual data which defines the product: a global product model holding overall information which potentially is useful to all processes, whereas the various local product models provide adequate and direct information support for each of the individual local applications. Through adopting non-proprietary data structures and information formats for representing the information stored in the global product model, it has shown that (i) engineering effort can be significantly reduced when establishing information support links with many different local applications, and (ii) the integrated system can have less dependence on specific features of the component proprietary systems.

8.3.1 Implementing a Global Physical Product Model

Thus a neutral data structure and information representational format (NDSIRF) was defined by the author: this being illustrated in Figure 8.3. Since there is no unanimously-agreed neutral format available at the present time,

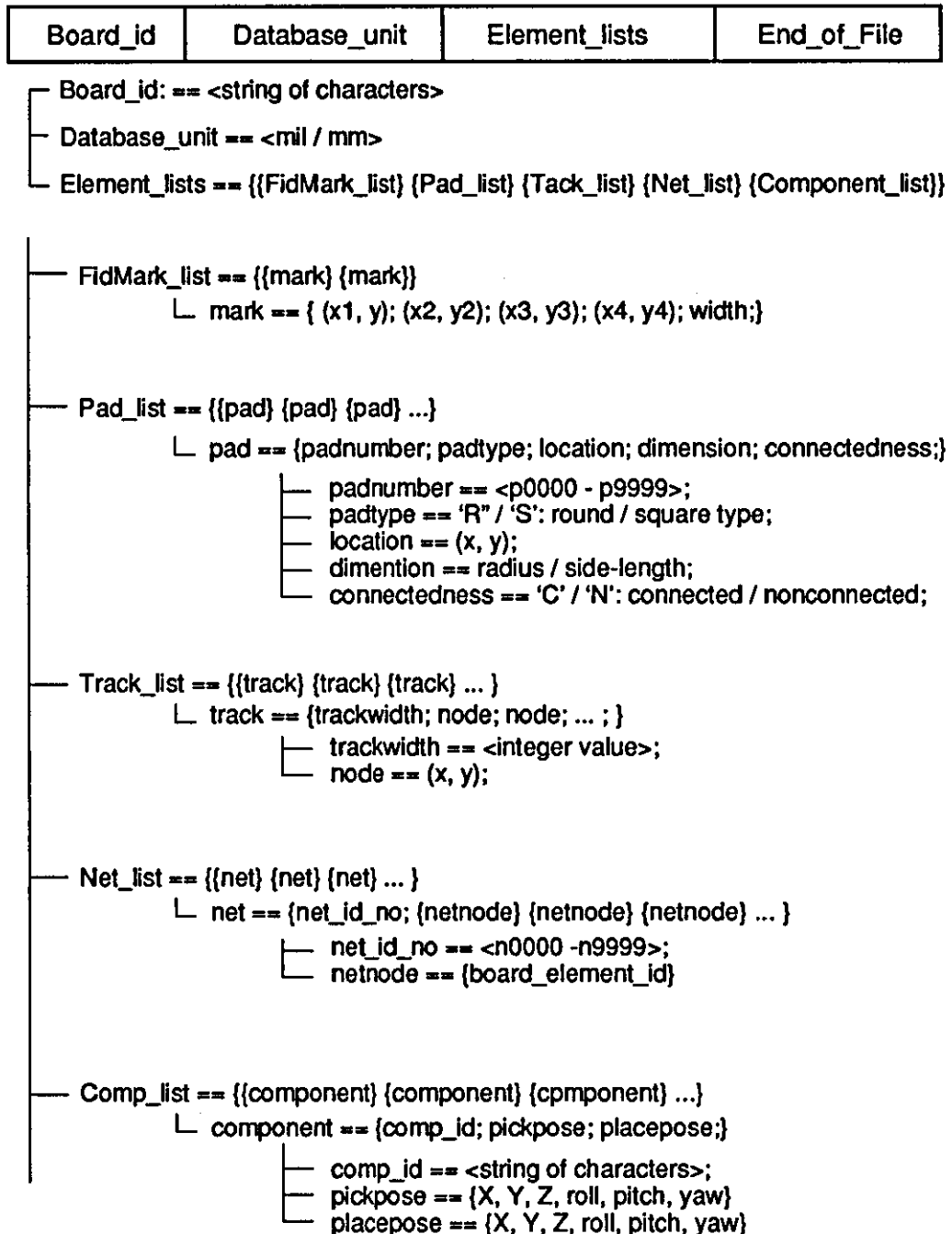


Figure 8.3 A Neutral Data Structure and Information
Representational Format (NDSIRF)

the NDSIRF used in the implementation has to be custom designed but serves to demonstrate the essential concepts involved. For simplicity, the NDSIRF defined here is intended only to encapsulate the necessary information required for AOI applications which fall into the category of “Panel Inspection”: this including inner/outer layer inspection, bare board inspection, and artwork film inspection (see Chapter 7). On the other hand, in order not to lose generality, an extra section (Comp_list) has been included in the format, which could be used to represent information useful for PCB assembly (e.g. using a robot). The information contained in the global product model would include such items as board identity number (or code) and several lists of data items which provide sufficient information on fiducial marks, soldering pads, conductive tracks, and components assembly locations. Again, these information items are indeed application dependent.

8.3.2 Software Development

The inter-system links (in terms of information interchange and information support) are established by using software which performs the major operations related to information manipulation (e.g. information extraction, interpretation and reformation). The software can be considered within two categories, namely, a) software for extracting global information from CAD systems (or CAD/CAM stations), and b) software to provide adequate information support to individual local applications.

A. Software to Extract Global Information from a CAD Database

As mentioned above, the information stored in the global model is expected to support many local applications (e.g. bare board inspection, PCB assembly, etc.). In accordance with Chapter 5, such information will be extracted from the design database of a CAD system by means of a “Software InfoGen” (for simplic-

ity, the term “Software InfoGen” will be further abbreviated and hereafter referred to as SIG).

Thus the SIG needs to represent its output (i.e. the extracted information) in the format defined by Figure 8.3. Since the input to the SIG used in this particular system implementation will be P-CAD database output files in a proprietary PDIF format (see Figure 5.6), the Lex source and the corresponding “*yylex*” code referred to in Chapter 5 (see Figure 5.7) that performs lexical analysis on the input P-CAD files was reused to form the new SIG without any need for modification. However, as in this new case both the content and format of the required output information is different (now in an NDSIRF for use with the global model, instead of being specifically designed to be directly accessed and used by the local Matrox vision system, as in Chapter 5), different Yacc code needs to be generated to achieve the functionality required from the SIG.

B. Software to Provide Information Support for Local Applications

To meet the individual application requirements for a local view of the information stored in the global product model, a SIG will be needed for each different application. Since the input to all these SIGs will be the global product information represented by the NDSIRF of Figure 8.3, each of the individual SIGs will include identical Lex code; i.e. code that performs lexical analysis of the same input. (Note that the Lex code used here will be totally different from that referred to in Chapter 5 and used in the software for extracting global product information directly from a CAD database). However, once again, different Yacc code will be required by each of these SIGs corresponding to provide output suited to each different class of applications (Figure 8.4.a)

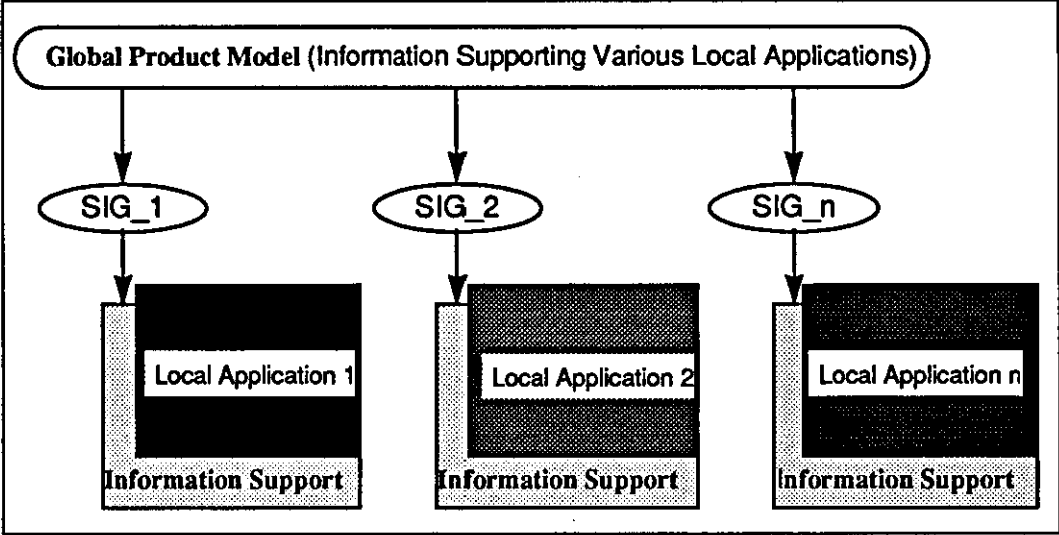
At first sight it might appear that totally different Yacc code will be required for each class of applications; and this would imply the need for significant systems engineering effort (but as explained in Chapter 9 will still be much less ^{than} that

required had a totally bespoke approach been used). However, on further examination of the problem it can be seen that this may not necessarily be the case, as the role of the Yacc codes is twofold (see Figure 8.4.b). Firstly, it performs grammar check on the tokens returned from the lexical analyser “*yyllex*”. The specification of the grammar rules is based on the data structure and representation conventions used in the input file (e.g. the neutral format, or proprietary formats such as PDIF). Therefore, this part of the Yacc code can be used by all the SIGs taking the same input (or indeed any part of the input file following the same convention). Secondly, it takes certain actions (by calling predefined routines) as soon as certain specified grammar rules are matched. The actual actions are defined by the designer of the SIG software and are application dependent (e.g. to print out a value just encountered, to copy a string or to add a set of coordinates to the track lists, etc.). Thus the essential difference between different SIGs lies in the predefined “action routines” which need inclusion.

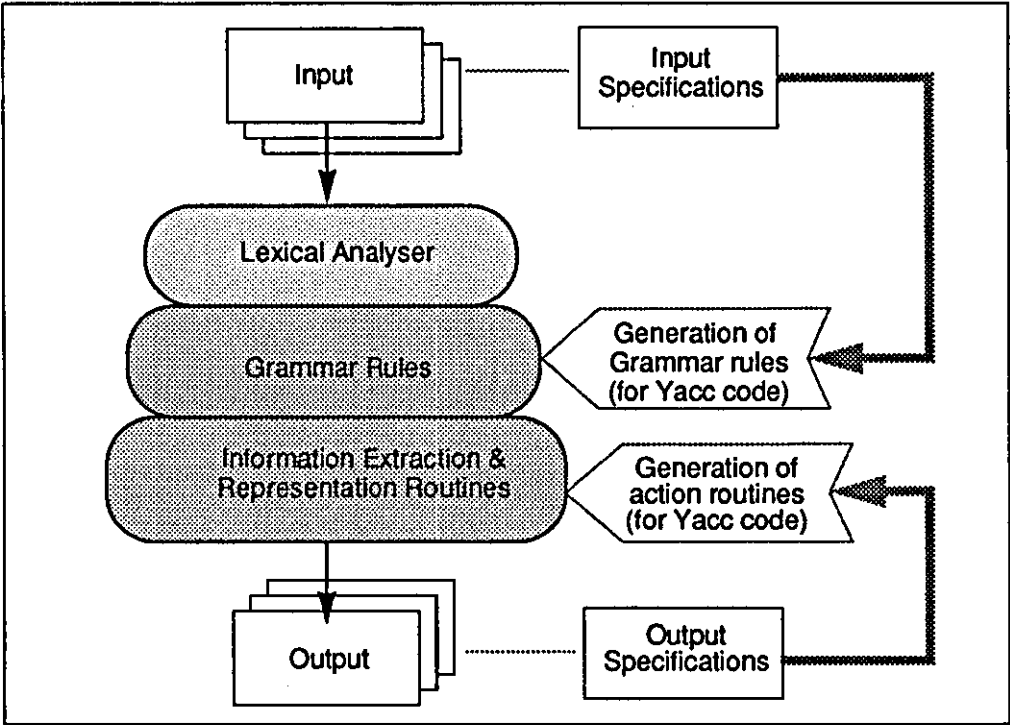
In the case of bare board inspection, the information requirements of the AOI system will be those characterised in Chapter 5. As such, the data structures and information representational format designed in Chapter 5 can be used in this particular implementation. Accordingly, a SIG was devised which functions to extract useful information from the global product model, and to present the extracted information to the AOI system; thereby supporting its use for bare board inspection.

8.4 Implementation of the AOI Reference Architecture

The product model based information support scheme presented in section 8.3 can offer significant flexibility in servicing the needs of various classes of local application. In the case of AOI applications, the approach makes it possible for the AOI system to make use of such available information to support it in performing inspection tasks. However, the benefits that will be realised by achieving system integration (and hence information support) will depend upon how the AOI system



(a) Uses of SIGs to Provide Local Applications with Information Support



(b) Basic Structure of a Software InfoGen (SIG)

Figure 8.4 Software InfoGen: Uses and Basic Structure

can make use of the information it is provided with, which in turn will also depend upon the use to which the information supported AOI system is put.

To further investigate this issue and the potential benefits that can be realised, the AOI reference architecture proposed in Chapter 7 was partially implemented by the author. Also this allowed the feasibility to be assessed of using such a reference architecture to guide the design, implementation and organization of software algorithms used by AOI systems. Most of the software algorithms and routines implemented when carrying out this investigation are listed in Appendix A. In the following sections, where appropriate, issues relating to the re-utilisation of CAD/CAM information available from the global product model will be highlighted and discussed in some detail, along with details of some of the more novel and interesting algorithms implemented.

• 8.4.1. The Primitive Routines

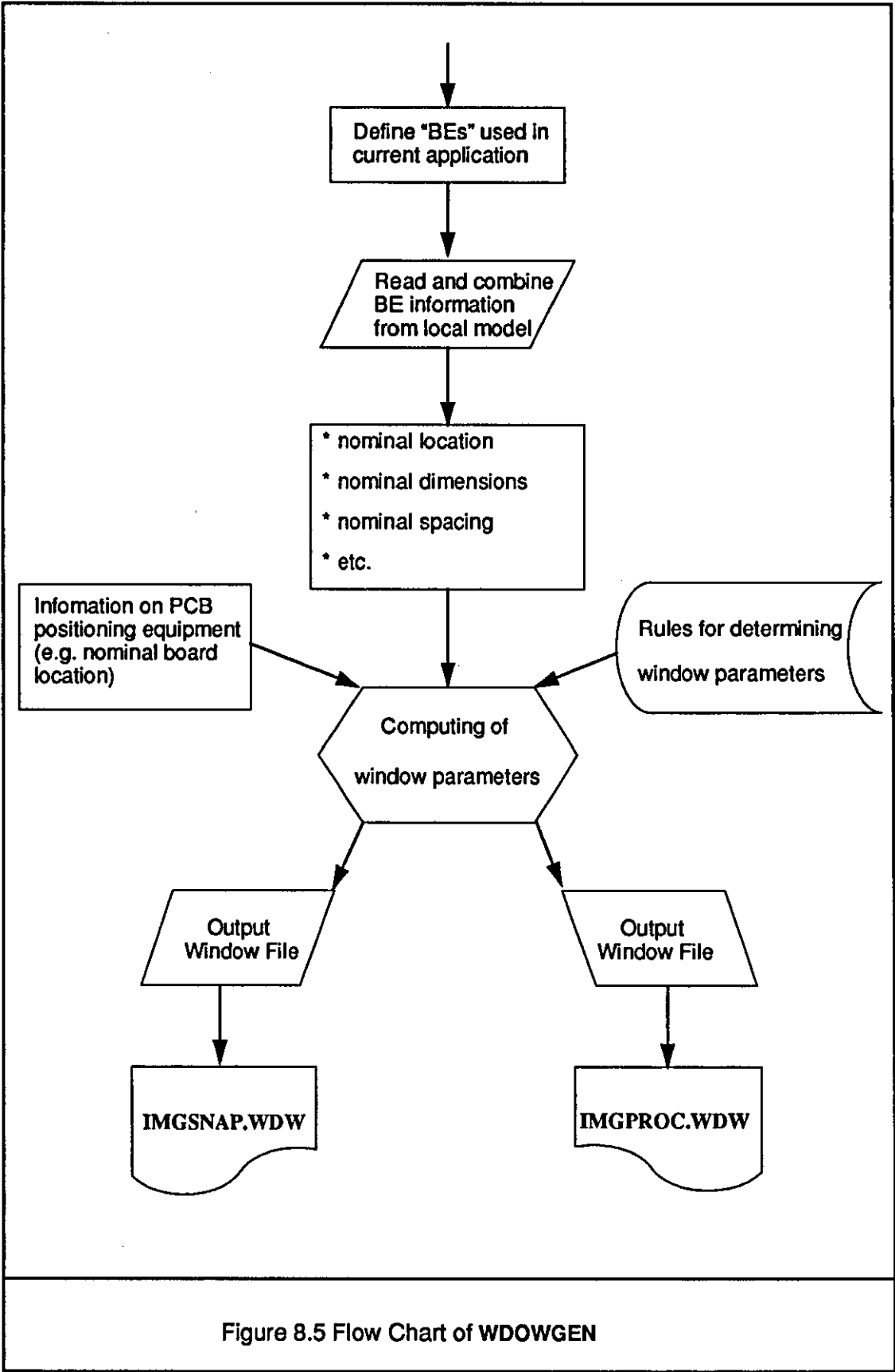
In the author's implementation, many of the library routines supplied with the Matrox vision processor have been included as the Level_0 primitive routines, with a view to drawing on previously available software and systems where that was possible, so that time and effort could be saved: this allowing the author to concentrate on an implementation of the higher levels of the reference architecture. However, generally speaking, the routines supplied in the Matrox library have very limited direct use in achieving automatic image interpretation for applications such as PCB inspection. Many useful features (such as perimeter, area, centroid, object boundary code, etc.) which are frequently used for object identification and/or product inspection can not be generated directly from the routines. Nonetheless, they provide some useful elementary subroutines (e.g. various edge operators, histogram calculation, etc.) which may be included as building blocks in user-generated programs to perform certain image analysis functions (e.g. for generating fragments of edge information, etc.). In fact, some of these routines have been included within

much of the author's software which carries out the higher level image processing and analysis functions (and indeed more exceptionally some of the author's software corresponding to the lowest level).

The potential use of available CAD information and process information was considered during the design of this new software: e.g. for the generation of window parameters that specify which subset of the image (or portion of the scene) is to be processed at which stage of the execution of an AOI task.

The use of a windowing technique can be very important during image processing as it allows analysis to be focused on a subset of the image (or a portion of the scene as one image). This is particularly so during PCB inspection: such as where certain board elements are expected, where they can be inspected without having to process the entire image (which is usually computationally more expensive). In fact in the author's study, the use of such windowing techniques was found to be especially suitable for AOI applications, where (as has been discussed in Chapter 7) the inspection task can be decomposed and the entire image divided into a set of sub-images containing certain board elements; algorithms can then be devised to extract local and global features of an element for use in a later stage of the AOI application.

Thus to test out this idea, a routine (**WDOWGEN**) was devised (see Figure 8.5) which takes as input the CAD information, as well as fragments of process information (e.g. PCB positioning accuracy) and generates two window data files whereby window parameters are held and can be used later by image processing algorithms. The CAD information used is represented in a format which is deemed suitable for the local application concerned. In the case of bare board inspection, the format designed by the author and referred to in Chapter 5 was used (as already mentioned in previous subsection 8.3.2.B). However, other applications may require different information format.



The first file generated by **WDOWGEN** is called **IMGSNAP.WDW**. This is used to specify locations where snapshots are required (see Figure 8.6). The file has two sections, namely, a *“fiducial mark section”* (**FIDSEC**) and an *“artwork section”* (**ARTSEC**). The first section, i.e. **FIDSEC**, is used to store information which specifies the expected locations of images representing (two or three) fiducial marks. In this case, mobility is often required of either the image acquisition subsystem (e.g. the camera) or the product to be inspected (e.g. on an X-Y table), and the locations should be specified in a way which is most suitable to the specific system involved. In this particular implementation, “relative mobility” was realized by means of the vision-robot integrated couple, with the camera mounted on the robot end-effector. It is also assumed that the nominal position of the fiducial marks are roughly known (i.e. without a requirement for high precision PCB positioning): in this implementation the effect of a rough positioning was simulated by manually positioning the board. Thus the images of the fiducial marks can be captured at these locations to enable subsequent processing and analysis so that the exact locations of the fiducial marks can be established, thereby providing a reference frame for the board.

In the second section of file **IMGSNAP.WDW**, i.e. **ARTSEC**, information is stored which specifies the locations at which images of a PCB (or sections of it) are to be captured. In cases where the PCB is sufficiently large that processing a single image of the board can not yield the required resolution, it is desirable that multiple images be acquired and processed individually, with each of them corresponding to only one section of the board. To serve this purpose, the file would have to contain more than one field of location data. In other cases, however, if processing one image of the PCB can achieve the required resolution, a single field of location data can be included in the window data file. Information contained in this file will mainly correspond to those **Level_0** routines relating to image acquisition.

File **IMGSNAP.WDW** /*stores locations where snapshots are to be taken;
used for acquiring images of fiducial makers (i.e. file section **FIDSEC**)
and images of sections of artwork (i.e. file section **ARTSEC**) */

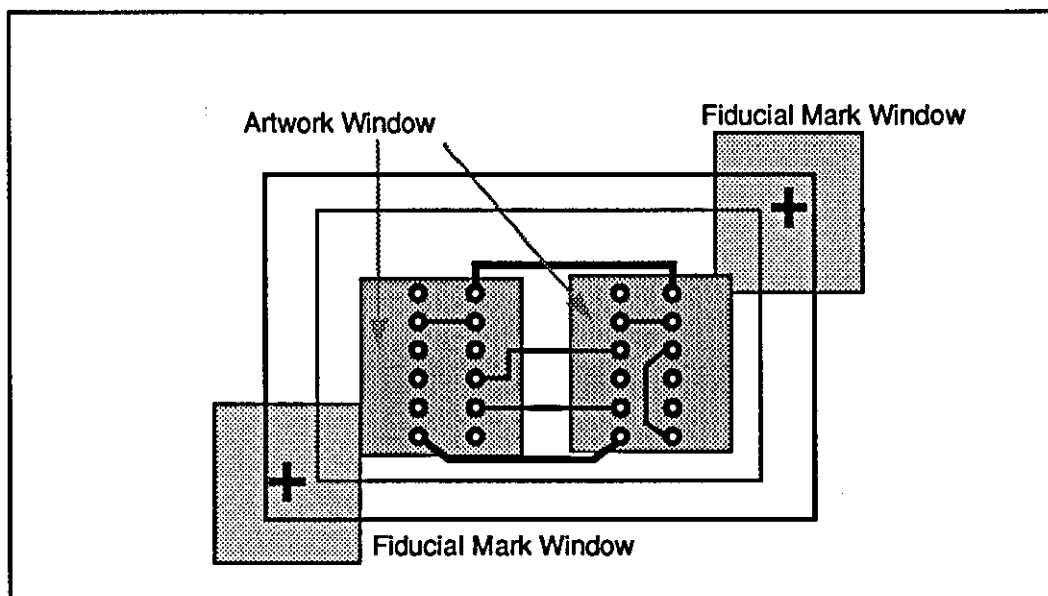
FIDSEC

```
{
  SNAPLOC { x, y };
  SNAPLOC { x, y }; /* Locations where fiducial marks are expected */
}
```

ARTSEC

```
{
  SNAPLOC { x, y };
  SNAPLOC { x, y };
  SNAPLOC { x, y };
  ... /* Locations where images of PCB sections to be taken */
}
```

(a) **IMGSNAP.WDW** File Structure



(b) Example Uses of Information stored in **IMGSNAP.WDW**

Figure 8.6 Window File Used for Image Acquisition

The second file generated by **WDOWGEN**, called **IMGPROC.WDW** (see Figure 8.7), stores information which specifies windows (i.e. subsets of an image) used by further image processing and/or analysis algorithms, typically in the class of **EAE** (element attribute extraction) algorithms. This would enable succeeding processing and/or analysis to focus on only those pixels within the defined window. In this file, a window is defined by means of the coordinates of its upper-left and bottom-right corners. The coordinates themselves are given in the image frame of reference. At runtime, routine **WDOWGEN** can be called by **Level_1** **EAE** algorithms to provide appropriate window information (by passing arguments which specify the information which needs to be returned in the format used by **IMGPROC.WDW**).

8.4.2. The Element Attribute Extraction (EAE) and Transitional Feature Generation (TFG) Algorithms

Routines classified as **EAE** and **TFG** algorithms have been implemented mainly for the purpose of extracting information fragments from the preprocessed image array data. More specifically, these are to be used for the extraction of local features (e.g. using **EAE** algorithms), and global features (e.g. using **TFG** algorithms) of individual board elements which are to be inspected.

A number of routines have been implemented which have been used mainly for processing binary images. A grey scale image is binarized by means of a thresholding operation (**THRESHLDG**, **THRESHMDFY**), often with the aid of intensity histograms of the original grey level image (**HSTOGRAM**). Chain code has been used intensively in the author's research study due to its efficiency and usefulness in representing object boundaries and in aiding further extraction of certain local features of the board elements. A routine (**CHAINCODE**) for generating such code has been implemented and used to extract object descriptions from the array image (see Figure 8.8). The advantages of binary image processing methods are speed and simplicity, as a result of the reduced image data to be processed.

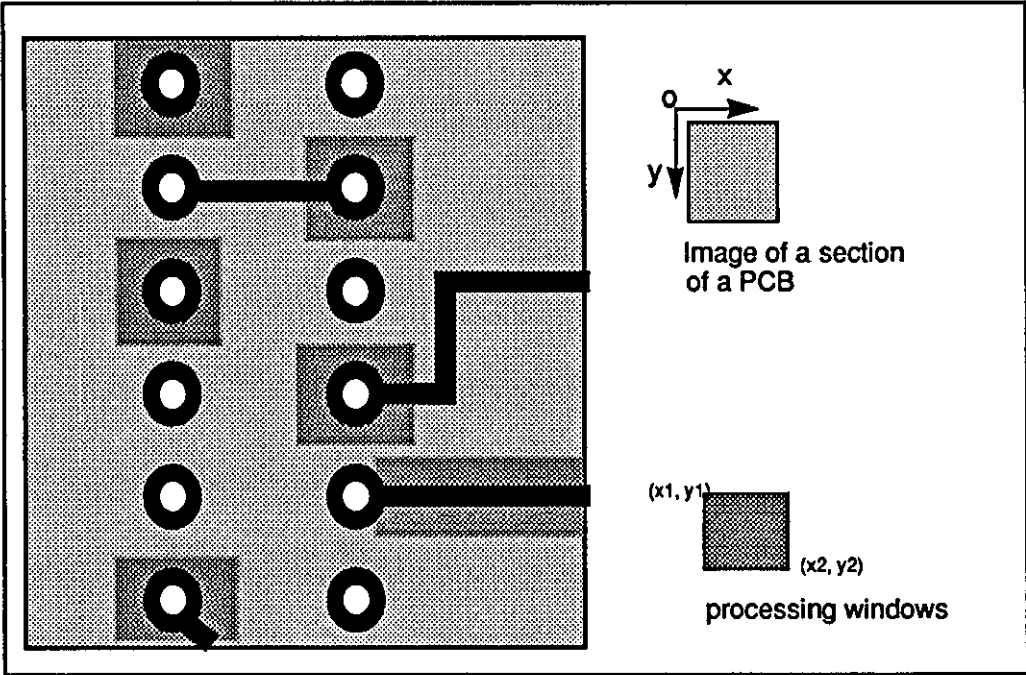
File **IMGPROC.WDW** /* Stores data specifying which subsets of current image are to be processed at a certain stage; file to be generated by calling routine **WDOWGEN**; */

PCB_Id: **abc.xyz** ; /* PCB Identity */
Section_no: **nn** ; /* sequential number referring to sections of a PCB */

Snap_location: **(x0, y0)** ; /* identifying image origin */

{
PROCWDOW **(X1, Y1, X2, Y2);**
PROCWDOW **(X1, Y1, X2, Y2);**
PROCWDOW **(X1, Y1, X2, Y2);**
 ...
 } /* set of data specifying coordinates of up-left & bottom-right corners of each of the processing windows; coordinates being pecified relative to the origin of current image, i.e. (x0, y0); */

(a) **IMGPROC.WDW** File Structure



(b) Example Uses of Information stored in **IMGPROC.WDW**

Figure 8.7 Window File Used for Image Processing

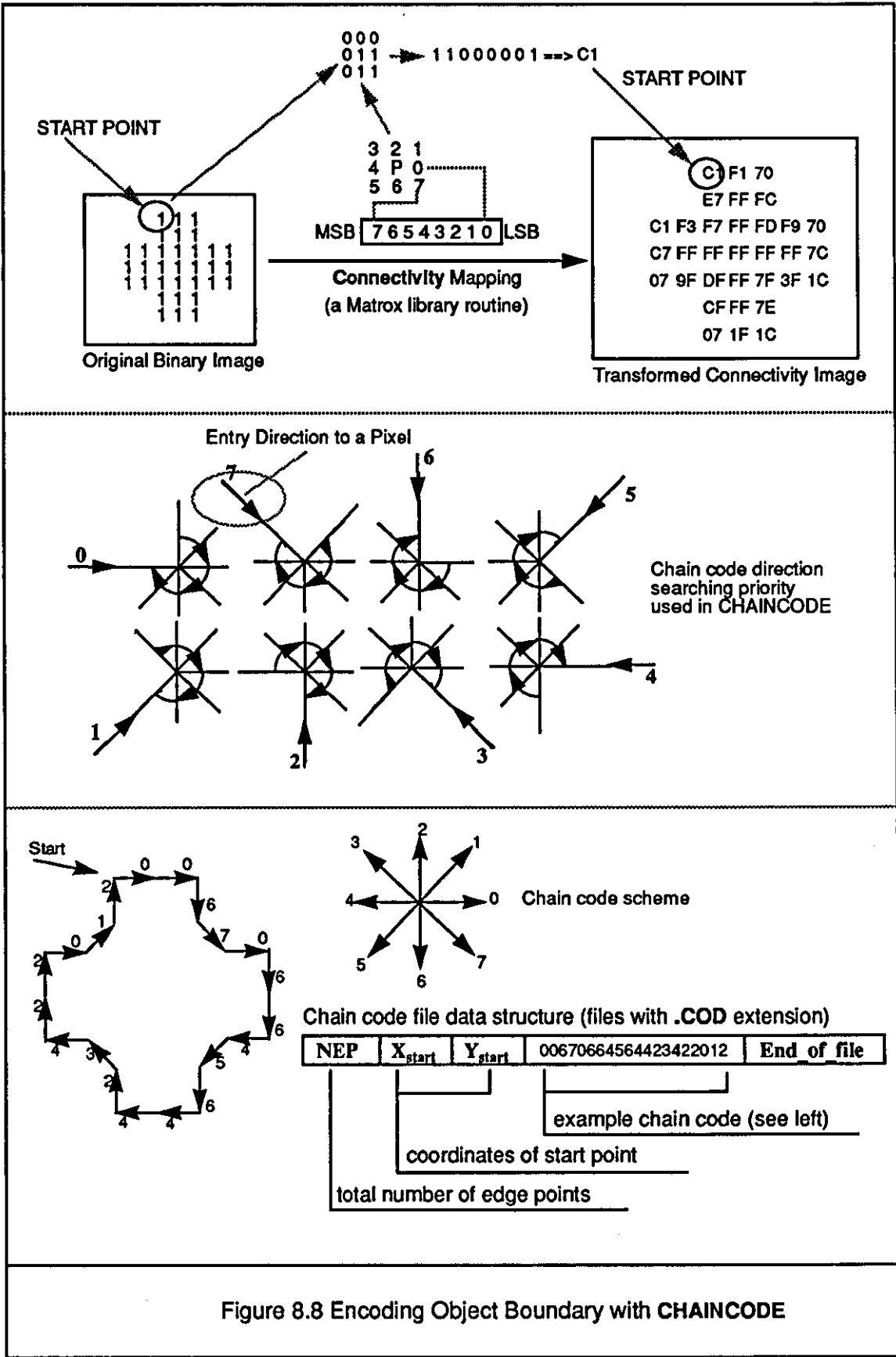
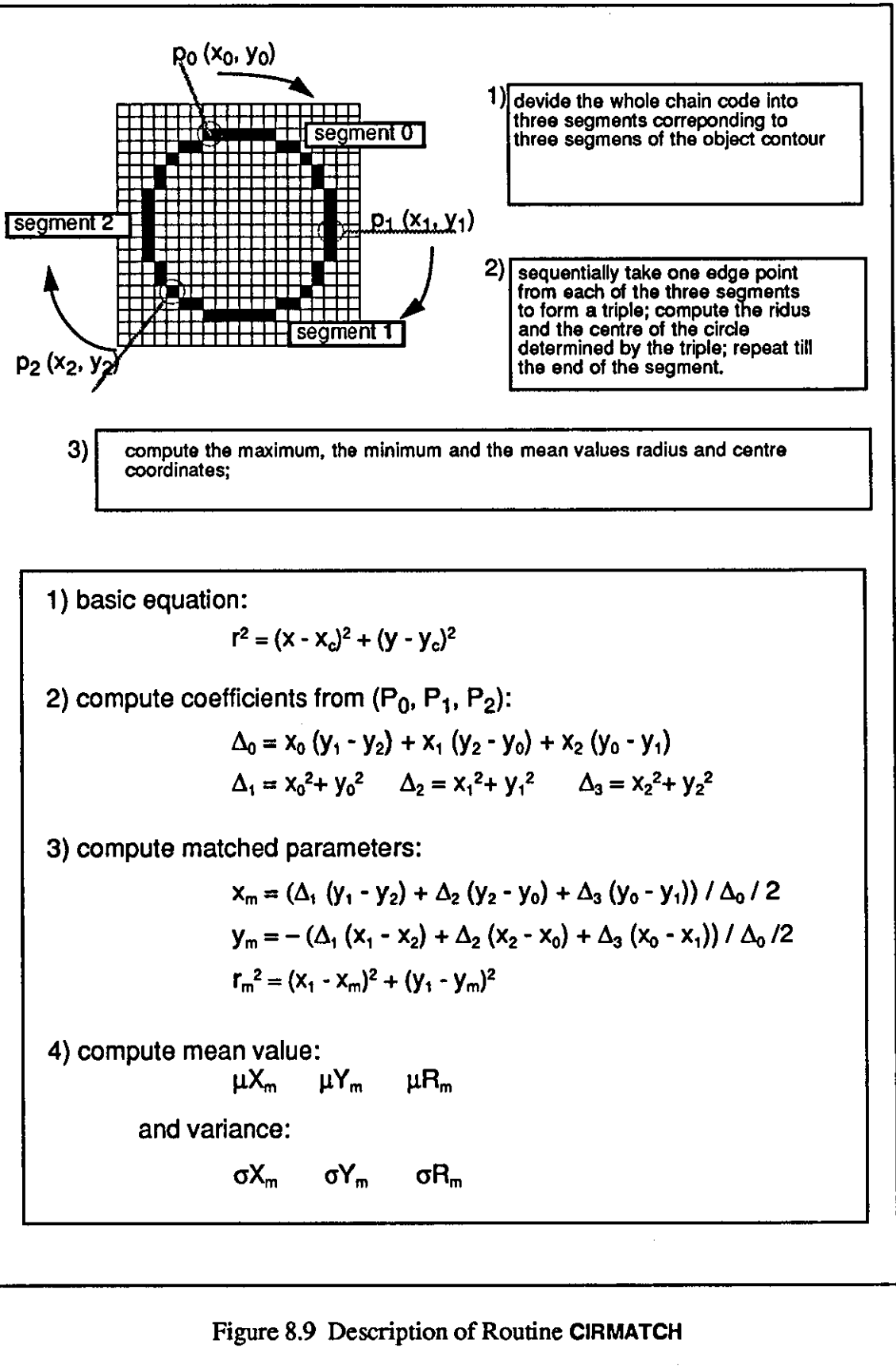


Figure 8.8 Encoding Object Boundary with CHAINCODE

Taking the chain code as input, several algorithms have been implemented which can be used to extract various local features of the original subset of binary image that the chain code was used to represent. For example, **CIRMATCH** (Figure 8.9) performs circle matching on the chain code and returns such features as the matched minimum, maximum and mean radius, area, and a measurement of the “closeness” of the object with respect to an ideal circle; **CNRPOINT** (Figure 8.10) takes as input a chain code representation and extracts corner information about the encoded object; **QUIKEXTRA** returns a set of simple features (e.g. area, perimeter, normalised roundness, centroid, maximum/minimum X/Y coordinate values, etc.) immediately upon finishing reading the chain code. A combination of these routines gives another more compact routine **LFEXTRA** which returns a set of the above mentioned local features of the object.

The local features of the object represented by a binary image (which may correspond to an expected board element) are further analysed by Level_2 algorithms, i.e. the TFG algorithms, to extract global features (referred to as *transitional region features*) of the same object. Attempts will also be made to recognise the object represented by the binary pattern. For this purpose, several TFG algorithms have been devised; including, for example, (i) **IDFYSHAPE** which identifies the shape of a particular board element (i.e. Round, Square, Others), based on its available local features, (ii) **CHEKCONN** which checks if a particular board element (mainly pads) is isolated or connected with other elements (for example, checks if a pad is connected with a track, based on the concavity/convexity of the boundary at corner points: see Figure 8.11.a), and (iii) **INSPECONN** which traces a track from its one end to another to verify the expected connectivity and track width. It is assumed in **INSPECONN** that the board frame has been aligned with the image frame in such a way that tracks will appear in the image either horizontally, vertically or with a 45-degree slope. This is based on the fact that in most CAD systems, tracks are only allowed in these directions relative to the board frame of reference: this constraint in fact actually being imposed by manufacturing process requirements, particularly in



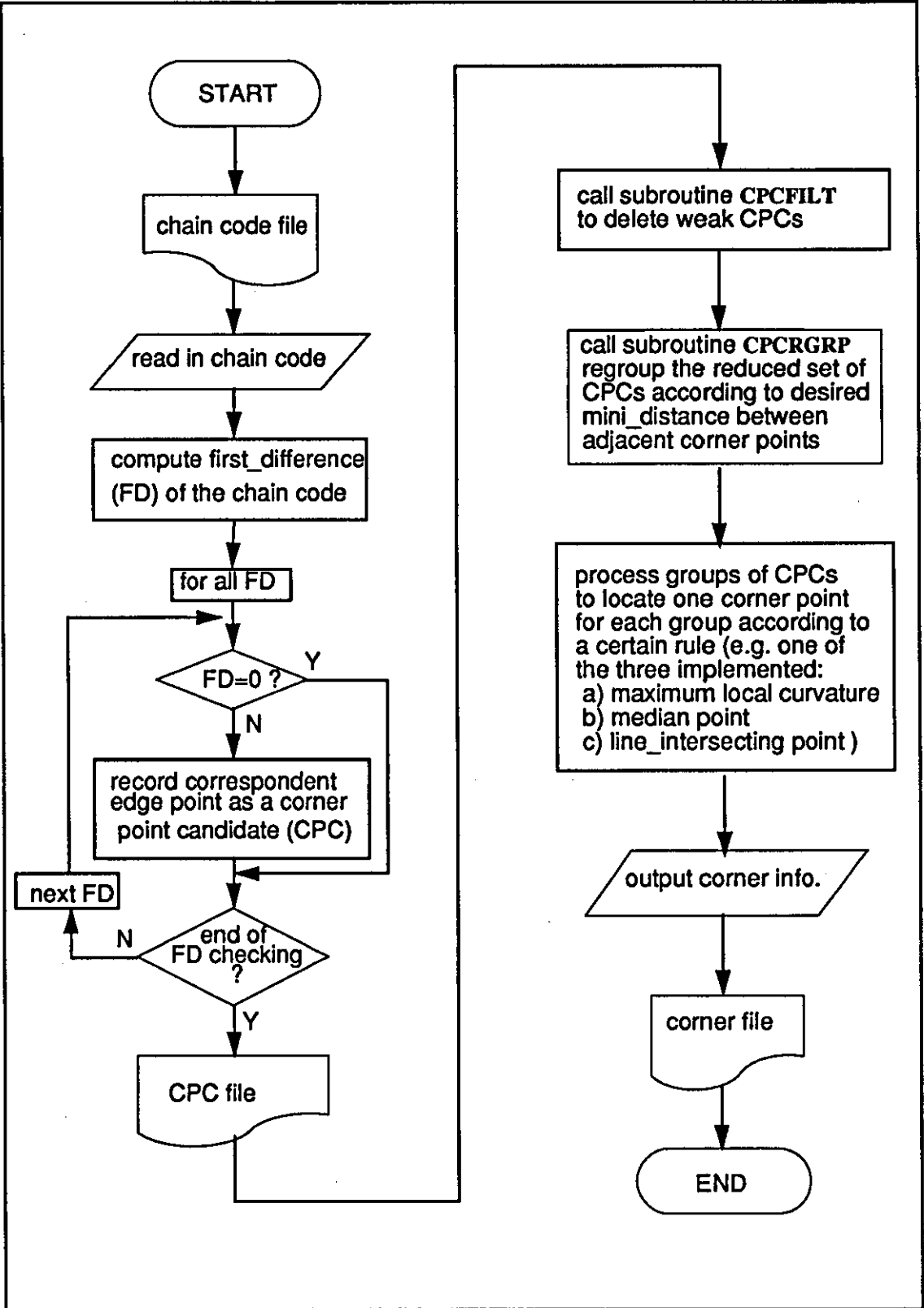
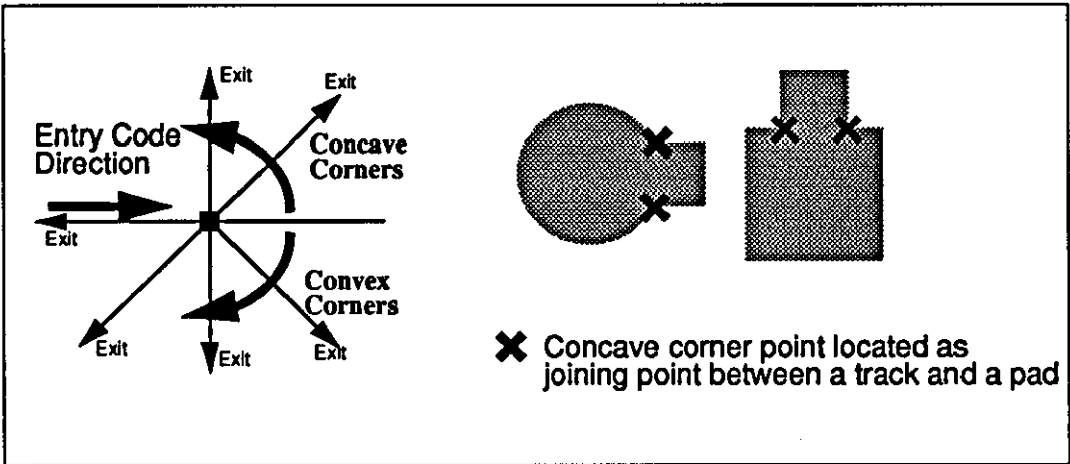


Figure 8.10 Flow Chart of CORPOINT

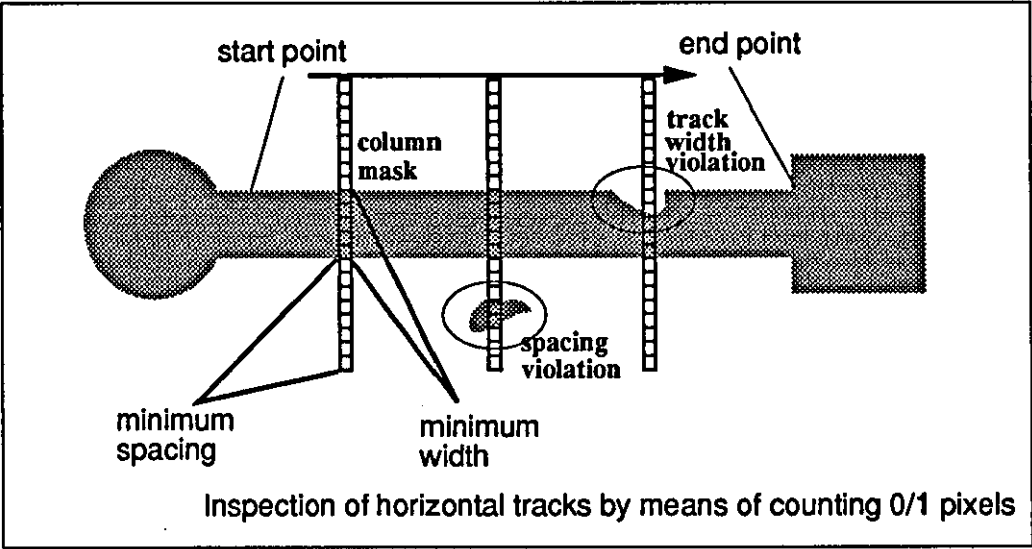
regard to bare board manufacturing (e.g. etching). With an alignment between the two frames, the operations performed during track width inspection can be simplified to that of checking the vertical pixel numbers for horizontal tracks or the horizontal pixel numbers for vertical tracks (a simple mathematical computation could make it possible to inspect tracks with a 45-degree slope --- or indeed in any other directions) (Figure 8.11.b).

Design information extracted from the CAD system has been utilised here to provide a geometrical model of the object being inspected. With such *a priori* knowledge being available, the inspection process of an AOI system effectively becomes a process of verifying the presence of certain expected features relating to the object under inspection, and of detecting the presence of unwanted exceptional features often characterizing product defects. Furthermore, the CAD information has also been utilised to guide the extraction of certain features of the object concerned. For example, when inspecting such board elements as pads, design information about the expected size and shape of the pads has been utilised in routine CNRPOINT to guide the grouping of adjacent corner point candidates into larger groups within which corner points are to be decided.

It is suggested by the author that the information extracted from a CAD database be distinguished as either “*referential*” or “*judicial*”. Here, “*referential* information” is considered to specify object features which will not be critical in determining whether or not the object concerned is defective, whereas “*judicial* information” is considered to be setting up rigid criteria (or rules) regarding the quality of the object being inspected. For example, in the case of PCB inspection, design information regarding the shape of certain pads may be considered as being “*referential*”, as the non-conformance of the actual pad shape with the design specification may not necessarily result in functional failure of the soldering pad (however, its size maybe so). On the other hands, design specifications concerning the electrical interconnections (i.e. the netlists) should always be regarded as “*judicial*” and hence used in deciding whether the board is defective.



(a) CHEKCONN to locate joining points between a track and a pad



(b) INSPECONN to trace electrical interconnection and to locate violations of track width and spacing

Figure 8.11 CHEKCONN and INSPECONN

Another example use of “referential CAD information” is in the case of locating fiducial marks, e.g. using the routine FIDLOC. In this case, CAD information describing design features of cross-shaped fiducial marks is used very effectively for extracting corner information relating to a fiducial mark. This corner information has been used to locate the fiducial mark very successfully, even if the mark has been partially occluded or deformed.

8.4.3 The Global Board Analysis (GBA) Algorithms and Application Oriented Algorithms (AOAs)

As defined in Chapter 7, GBA algorithms are concerned with processing and analysing the global BE (board element) features generated by the middle level algorithms. These global features typically characterize a board element by means of its measured area, dimension, shape, location, detected connectivity as well as a measure of conformance of these features to the expected one, and so on. However, from the board’s perspective, these features represent only local properties of the board, as they characterize only one specific local element of the board. A collection of these transitional features, at each and every subarea of the board, form a complete set of local board features that can be further processed or analysed to extract higher level, global features of the entire board being inspected: this type of further processing and analysing being typical tasks performed by GBA algorithms and/or AOAs that generate and present the inspection results (i.e. global board features) either in a generic format (i.e. the output of GBA algorithms) or in a specific format suitable for a specific application (i.e. the output of AOAs).

An implementation of GBA algorithms has focused mainly on analysis of the global relationships between board elements. Again these relationships are considered from two perspectives, namely, (i) the geometrical relationship and (ii) the electrical relationship (i.e. electrical interconnections, known as connectivity maps or electrical netlists). Geometrical relationships between individual board ele-

ments are examined by **GEANALYS**, which accomplishes its task by means of processing two tables known as “geometrical distribution tables” each of which consists of a list of items like BE identity (e.g. an identity number) and its location, as illustrated in Figure 8.12. The first table is extracted from the CAD information, and the second is compiled from the extracted global features of these board elements. The utilization of the results obtained by performing such a geometrical analysis has been considered from the following two perspectives, viz:

-) Detection of global geometrical distortions caused by proceeding manufacturing processes: For example, if all board elements have been manufactured with a common translation in location relative to the board frame, on average a consistent location error would be detected as a result of applying **GEANALYS** for each of the board elements. If such a detected “distortion” were to be considered to be severe and/or had persisted for a statistically valid number of cases, it would^{be} appropriate to request that adjustment be made to the relevant manufacturing processes.
-) Calibration of design model used within the CAD system and the provision of information support for succeeding applications: If common location errors are detected, then constant offset values which define the relative location of the theoretical and actual position of board elements (e.g. a set of pads forming a footprint of a surface mount component) can be determined and used to direct subsequent processing operations. Thus rather than treating the board as being functionally defective, the detected common location error could be downloaded to succeeding processes, such as PCB assembly. Such a procedure can be viewed effectively as calibrating the CAD information, so as to provide optimal location information for assembly.

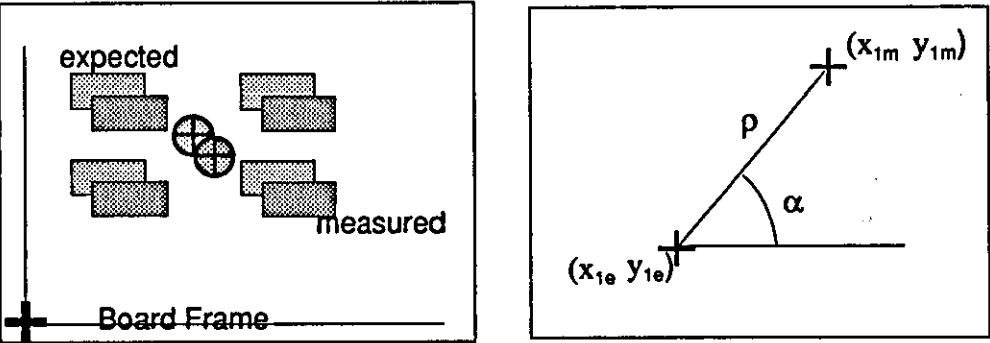
Table #1

BE_ID	Expected Location	
BE001	x_{1e}	y_{1e}
BE002	x_{2e}	y_{2e}
BE003	x_{3e}	y_{3e}
BE004	x_{4e}	y_{4e}
...

Table #2

BE_ID	Measured Location	
BE001	x_{1m}	y_{1m}
BE002	x_{2m}	y_{2m}
BE003	x_{3m}	y_{3m}
BE004	x_{4m}	y_{4m}
...

(a) Example Tables



(b) Compute Location Error for All BEs in the Two Tables

Figure 8.12 Illustration of GEANALYS

It should be pointed out that **GEANALYS** has been used mainly for analysing geometrical relationships between pads which together form a footprint of a particular component, or between component footprints present on the board. The latter operation is usually performed based on the inspection results of the former (i.e. optimal footprint locations are returned).

The author's implementation of GBA algorithms for electrical interconnection inspection (EII) is incomplete in the sense that only a few representative subroutines have been generated, which collectively can be used as component routines to build up a software for EII. These subroutines include (i) **ERODNET** which erodes an image, (ii) **DILTNET** which dilates the image, and (iii) **COMPANET** which compares two interconnection lists. **COMPANET** can be used to compare an extracted netlist with that specified by CAD information, allowing detection of interconnection violations, or to compare a netlist extracted from the original image with that extracted from either the eroded image or the dilated image, this in turn allowing detection of violations of track width or spacing specifications. However, there still remains the need for a routine that can be used to extract an interconnection list from a given image (which can be either the original image, the eroded image or the dilated image). Once this is done, **COMPANET** can be called on to perform comparison between the two given net lists and to pinpoint disparity between them.

The inspection results (i.e. the global board features extracted by the GBA algorithms) are arranged in a predefined consistent data format, thus facilitating further processing and analysis by the application oriented algorithms (AOAs). This format is basically similar to the one given in Figure 8.3, except that an extra section (see Figure 8.13.a) has been included for representing defects found during inspection processes.

AOAs are required to process global inspection results, and can provide information support for other specific applications. Three routines have been implemented, all taking as input the same set of global board features (i.e. the inspection

result). The first (**ERREPORT**) has been used to provide a summary of the flaws (errors) detected by the automatic PCB inspection process. The format of the output obtained using **ERREPORT** is illustrated in Figure 8.13.b. The second (**ERRTREND**) was used to analyse the input global board features file and the required type of the output, and then calculates a histogram of the defects found. The output Y-axis is the number of the defects found, and the X-axis is the type of defects (see Figure 8.13.c). Obviously, any definition of defect types will be application dependent. The third (**ERRDISP**) has been used to display the detected defects on a video monitor, superimposed on the background of the image of the PCB being inspected.

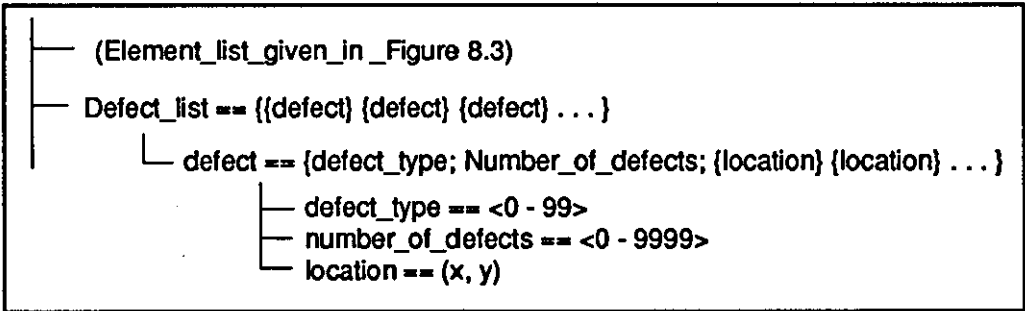
8.5 The Vision-Robot Integrated System and its Calibration

The “mobility” of the machine vision inspection system (or more exactly the mobility of the camera used by the vision system) is included in the implementation and is realized by the integrated vision system and AdeptOne robot couple, with the camera mounted on the robot (see Chapter 4 for details). Here the focus is on introducing some details of the software involved.

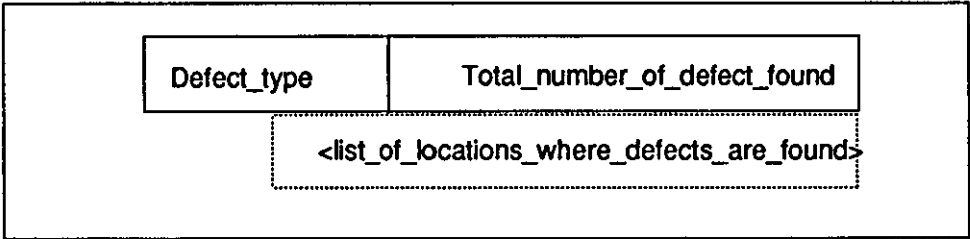
8.5.1 The Communication Link for the Vision-Robot System

The communication link between the vision system and the robot system is established and maintained by two routines, one for each end system. On the vision end, this is done by the routine **VIROCOMM** and on the robot end by the routine **COMMLINK**. Routine **VIROCOMM** was written in the C programming language and has been included in the **ALGLIB** module of the software platform **SPADAT** introduced in section 8.2. Thus it can be executed in a interactive manner. In Figure 8.14, a flow chart is given of the routine **VIROCOMM**.

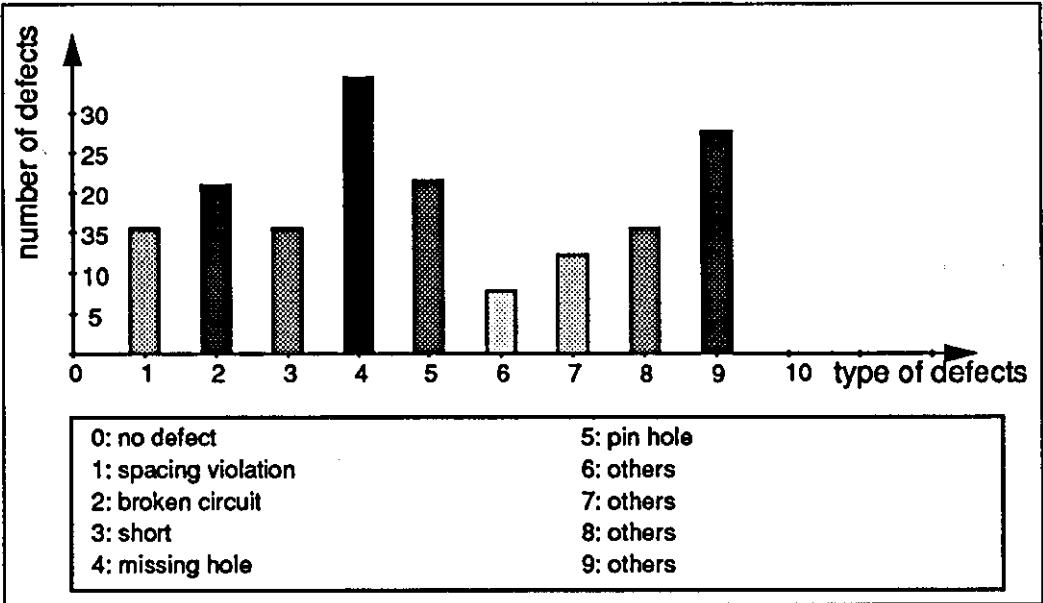
Routine **COMMLINK** was written in the VAL-II robot programming language and run within the VAL-II control system. When **COMMLINK** is invoked, the



(a) An Extra Defect_Section Used for Global Board Features



(b) Format Used for Defect Report by ERREPORT



(c) Explanation of ERRTREND

Figure 8.13 Manipulation of High Level, Global Board Fetures

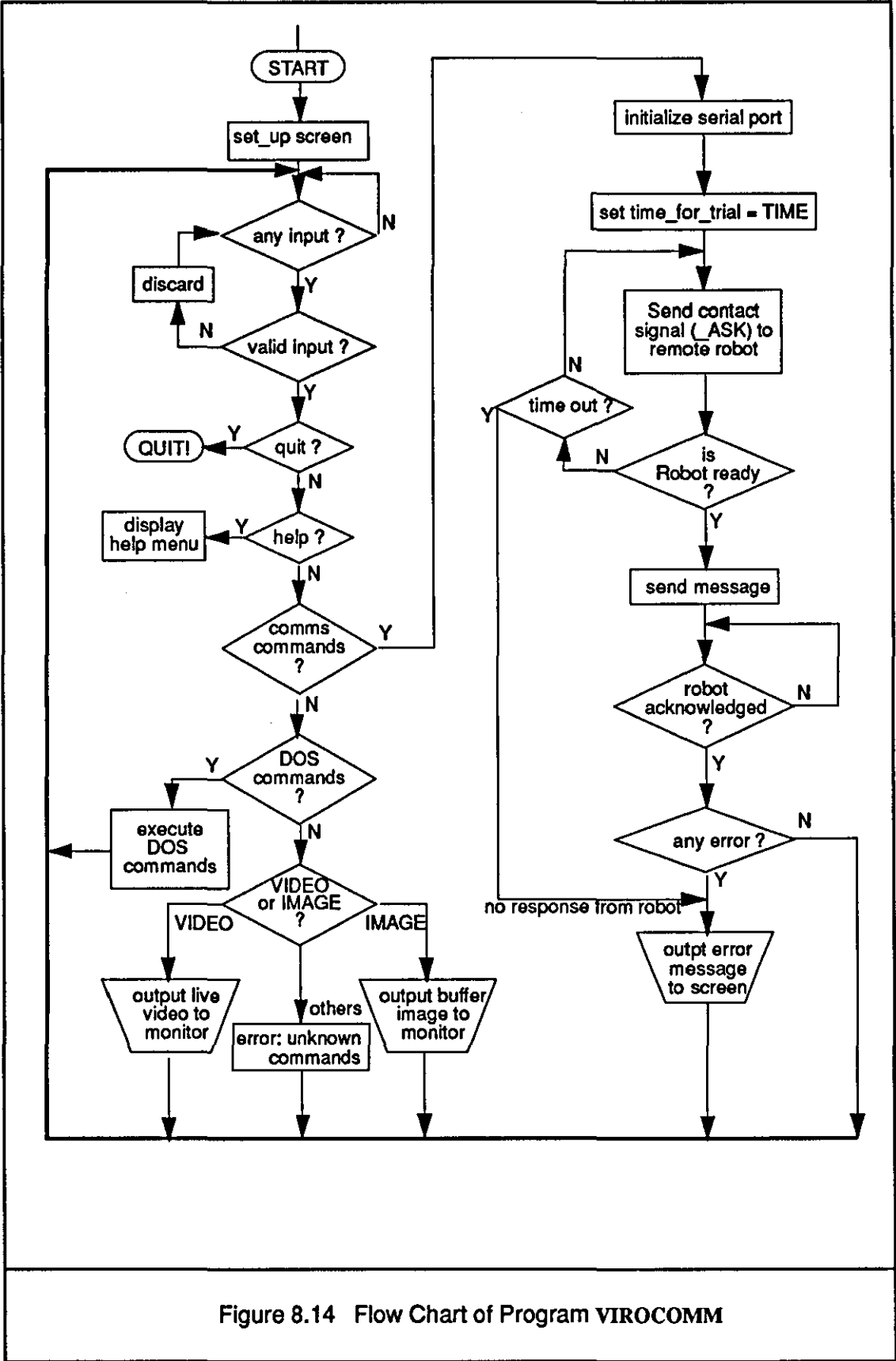


Figure 8.14 Flow Chart of Program VIROCOMM

robot waits for instructions to be downloaded via RS232 from the remote vision system. The instruction would typically be a motion control command used to drive the robot (with a tool-mounted camera) to a specified location so that an image can be acquired at that location. Before executing the received motion control instruction, a safety check is performed which ensures that the required motion is within a safety range. If it is safe to carry out the specified motion, the robot is driven to the demanded location and a message (including the final robot location reached) sent back to the vision system acknowledging the accomplishment of the demanded action. If an inappropriate movement is required an error message is sent back to the vision system, and no motion will be carried out. Other instructions may also be sent to the robot; for example to request vision-robot calibration data, the current location of the robot, or the status of the robot system. A flow chart of the routine **COMMLINK** is given in Figure 8.15.

8.5.2 Calibration of the Vision-Robot System

Calibration of the vision-robot integrated system is carried out in two aspects, namely,

- 1) Calibration of the relationship between the camera reference frame and the robot reference frame. This serves to establish a transformation from the robot tool space to the image space, thus allowing the location of the origin of the image space to be calculated from the robot location. This is done using a calibration utility (i.e. **A.TCAMCAL**) supplied with the AdeptVision system.
- 2) Calibration of the scale factors for the optical system: This serves to establish a quantitative relationship between a value (represented in pixel units) and its equivalent in a user specified unit (e.g. in inch or millimetre). Often the scale factors along the image X-axis and Y-axis are different, thus both need to be cali-

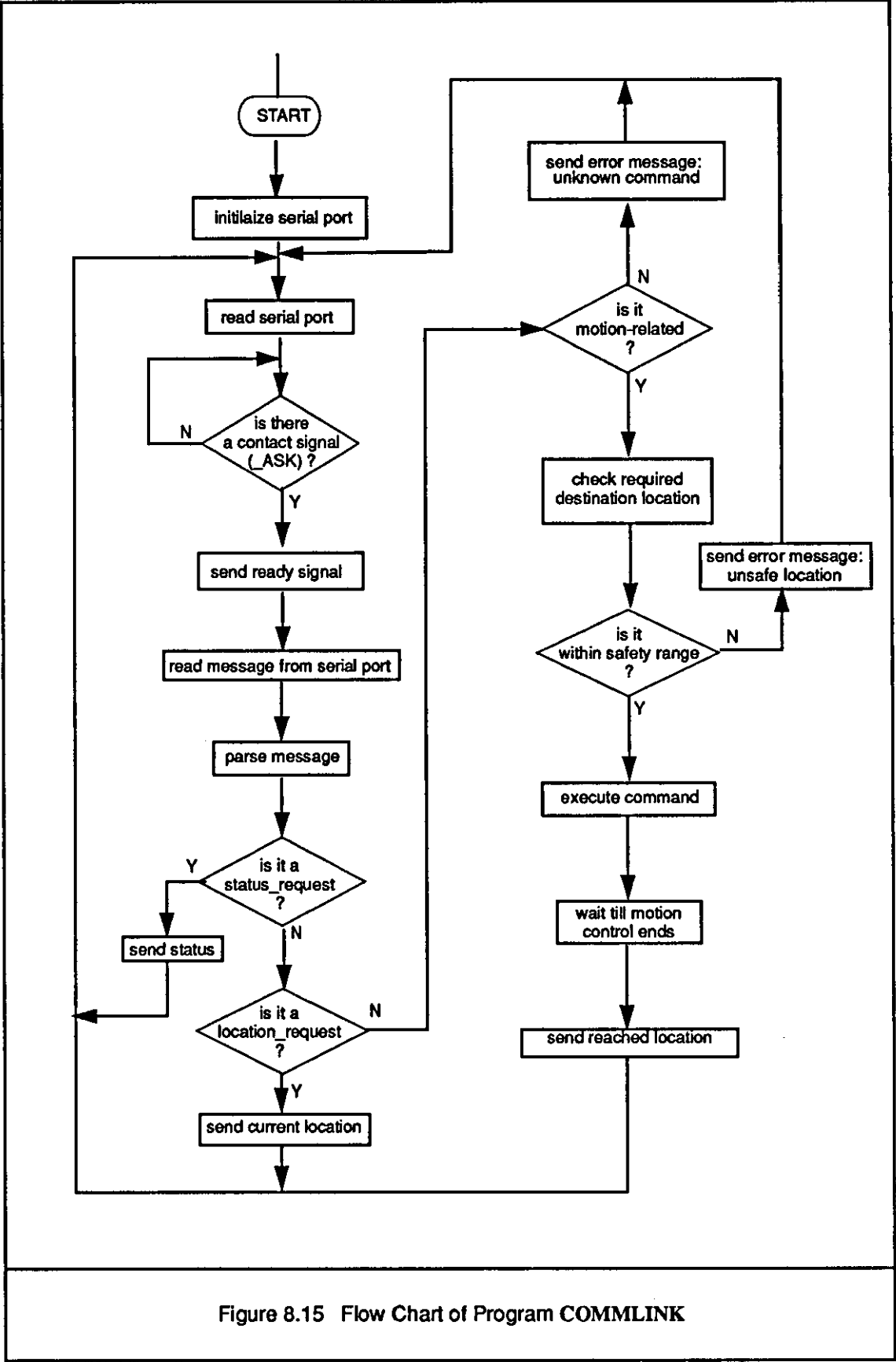


Figure 8.15 Flow Chart of Program COMMLINK

brated. A routine, i.e. **VISCAL**, was designed and implemented to carry out the calibration task (Figure 8.16). For simplicity, it is assumed that the object plane is parallel to the image plane.

8.6 Experimental Results: A case study

8.6.1 Introduction

This section presents the results of an application case study which demonstrates important features of the author's proof of concept information supported AOI system; the case study being based on illustrating how the automatic inspection of PCB artwork can be achieved. Photographs of the PCB artwork and its sampled images, which were processed and analysed by the system, are included together with examples of the results obtained.

The term "PCB artwork" is used here to refer to a pictorial representation of the PCB layout, which shows exact locations, shape and dimensions of all tracks, pads, via holes and fiducial marks, and is used as the master artwork for mass production of the designed board. As such, inspection of this type of PCB artwork is highly desirable in order that defects present in the artwork can be detected (and thus eliminated) as completely as possible before the artwork is released for mass production, thus preventing the manufacturing of defective products as defined by the artwork. In the case study system, photoplotted PCB artwork (designed using the PCAD system and represented by PDIF files in the form referred to in Appendix B.1) was inspected using the information supported machine vision system. Photo 8.1 illustrates an example photoplotted PCB artwork created in this way to prove the system operation.

8.6.2 System Configuration

The system used in this case study consists of a Matrox vision system and an AdeptOne robot manipulator system, both of these component subsystems

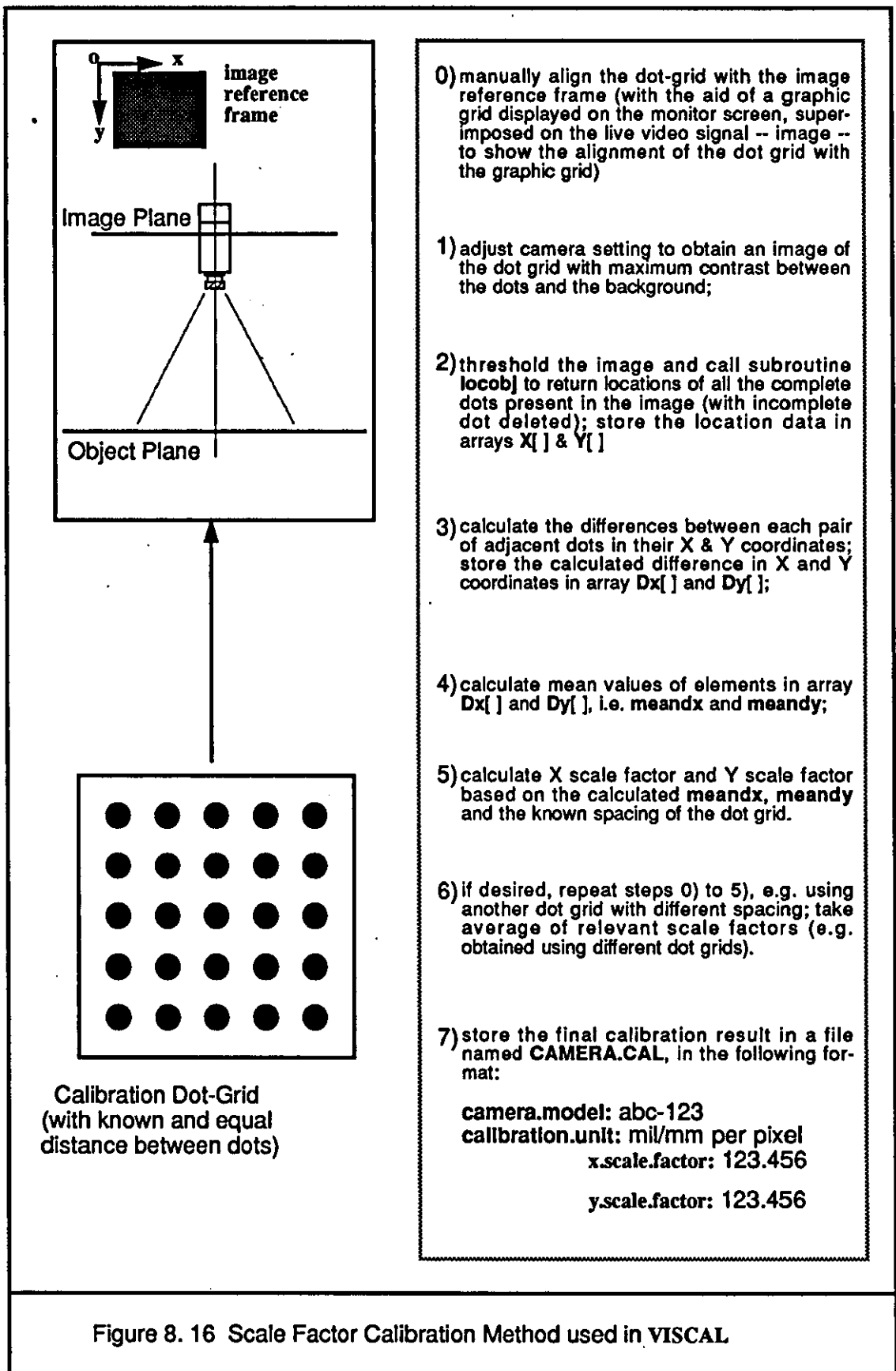


Figure 8. 16 Scale Factor Calibration Method used in VISCAL

have been described earlier in Chapter 4. The PCB artwork was placed under the camera, above a light box which provides back lighting. Mobility of the inspecting camera was incorporated as a means of achieving the desired resolution; this being facilitated by the integrated robot and vision couple described in Chapter 4. The arrangement of the entire system is similar to that illustrated in Figure 4.1, except that an additional monitor was used to display and monitor communications between robot and vision system (see Photo 8.2). Example displays on each of the three operator interaction screens are shown in Photos of 8.3.a to 8.3.c respectively (one for the display of communications commands received, one for the display of grey level or binary images and the other for the initiation, control and execution of the vision inspection commands).

8.6.3 Inspection/Location of Fiducial Marks

One of the demonstrated benefits of using an automatic vision system is that the registration of the PCB artwork (or the PCB) can be achieved using vision software, thus eliminating the requirement for precise PCB positioning. This can lead to important savings by negating the need to construct special precise positioning apparatus. Registration is a common prerequisite in many PCB manufacturing and assembly processes such as drilling operations, application of solder paste, component placement, etc. These processes often involve the accurate placement of material or component (or the drilling of holes) on the PCB. The PCB manufacturing process is such that each PCB may vary slightly from the design and; therefore PCB positioning via vision is essential to ensure the correct location of parts (or drilled holes) on PCBs.

On the other hand, in order to enable the AOI system to make use of design information about the location of board elements (so as to carry out the inspection task), it is necessary that the fiducial marks which defines the board reference frame be located in the first place. Thus the first task in PCB inspection is often concerned with establishing/calibrating the location of the board fiducial marks so that the actual board frame of reference can be determined and set up.

To meet this requirement of artwork (and PCB) inspection, the author has devised and implemented a novel algorithm (i.e. **FIDLOC**) which, through making use of the CAD information on the fiducial marks, can quickly locate the centre point of the fiducial mark and return its coordinates in the pixel reference frame. More importantly, this algorithm can accurately locate the true centre point (instead of its centroid) even if the fiducial mark is partially deformed (as long as the central part of the fiducial mark remains not contaminated), offering the capability for the inspection system to set up the board frame precisely. Examples of grey level and binary images of a normal (good) fiducial mark and two deformed fiducial marks are depicted by the photographs (see photos of 8.4.a/b/c to 8.6.a/b/c. respectively) together with the specimen final results (i.e. the coordinates of the located centre points) returned by the **FIDLOC** routine.

Once the fiducial marks are located and the actual board frame of reference is set up, expected locations of all board elements can be obtained (through referencing CAD information) with reference to this new board frame. The camera can be moved (by controlling the movement of the robot arm) to various programmed locations to capture images of PCB sections for subsequent processing and analysis. In this case study, CAD information about the nominal locations of board elements are used in routine **WDOWGEN** which automatically generates a file of locations for image capturing (see Figure 8.6). This routine also allows for changes to AOI system configuration to be accommodated by reading from a file containing configuration information such as the size of each image frame (e.g. 512 by 480 pixels), the number of lines to be overlapped between consecutive image frames both vertically and horizontally, the calibrated scale factors along X and Y direction, etc.

Other benefits of using CAD information to support AOI operations can be identified by considering the case of inspecting PCB tracks with various widths. Traditionally this has been a difficult problematic area for stand-alone AOI systems to accomplish; this typically being achieved through using a conventional “design rule” approach to track inspection, where a frequently used rule (which specifies the minimum track width) will not be applicable when tracks with various widths are present in the image. However, through the availability of support information (gen-

erated initially during design processes) which specifies the location and width of each track (or track segment), the AOI system is able to assign and apply the correct design rules (or inspection criteria) to tracks of differing widths. Again this requires the use of a novel vision algorithm (such as **FIDLOC**) for the purpose of accurate and automatic fiducial mark location.

8.6.4 Inspection of Pads

Pad inspection is required to verify that pads have been manufactured according to the shape and dimensions specified by the design information (e.g. the output of a CAD system). It is also desired to verify that a set of pads form a valid footprint (i.e. the geometrical relationships between these pads are as designed) where a certain type of component is to be mounted. For this latter requirement, the **GEANALYS** routine described in section 8.4.3 (Figure 8.12) has been used. Several other algorithms were also implemented and proven by the author for pad inspection; these include board element local feature extraction algorithms (i.e. the **EAE** algorithms) such as **LFEXTRA**, **CIRMATCH**, **LMATCH**, and board element global feature extraction algorithms such as **IDFYSHAPE**.

To illustrate examples of the results obtained, a binary image is included in Photo 8.7.a of a section of the PCB artwork illustrated in Photo 8.1. For those isolated pads (i.e. pads without tracks connected to them, like the square pad and a round pad in the image) a chain code representation can be generated directly from the binary image, and features of the pads can be extracted and analysed. Sample sets of local features extracted by **LFEXTRA** from the chain code of a square pad and a circular pad are shown in Photo 8.7.b and Photo 8.7.c, respectively. (Most of the features in the feature set are hopefully self-explanatory; Roundness is defined as: $\text{area} / (\text{perimeter})^2$, normalised to 1.0 for a perfect circle. The author found that “Roundness” and “LM_error/CM_error”, i.e. the ratio of line_match_error to circle_match_error, are two of the most powerful local features for distinguishing a circular object from a square one.)

A binary image of another section of the PCB artwork of Photo 8.1 is shown in Photo 8.8.a. Here we consider the analysis of those pads that are connected to conductor tracks. One approach implemented by the author was to apply an “open” (erosion followed by dilation) operation to remove the tracks, leaving only pads in the output image. The result of such an operation is shown in Photo 8.8.b. Analysis can be performed with ease on the pads in this image (as now essentially they are isolated). Photo 8.8.c shows the result of carrying out alternative operations which serve to remove pads and to leave only tracks in the image; this being the result of subtracting the “pad image” (i.e. Photo 8.8.b) from the original binary image. This will simplify the extraction of track information from the image, for example in establishing the start and end position of a track segment.

8.6.5 Problems of Noise in the Inspection Process

Due to noise introduced in the various operational stages (such as during the process of photoplotting, image formation and/or digitising), neither the original plotted artwork, nor the grey level (or binary) images of the artwork will look exactly the same as that defined by the CAD information. For example, consider the case of track inspection, in an ideal situation a track segment specified by the design information would typically appear as illustrated in Figure 8.17, whereas grey level and binary images of a real track segment (as illustrated in Photo 8.9.a and 8.9.b, with artificial defects added) will have noise pixels along the edge of the track. Thus if the AOI system operates with a relatively low resolution (i.e. with big values of X and Y scale factors), the system would be unable to distinguish real defects from noise; this will typically result in either false alarms or missed defects.

As a result, in order to validate the operation of CAD information supported AOI, one must take into account the effect of the noise, while trying to minimise its effects (for example, in some situations an “averaging operation” can be performed before thresholding so as to remove high frequency noise that would usually result in an unsmooth object edge in the thresholded binary image). The resolution of the AOI system should also be high enough to enable the noise (such as that introduced by digitising) to be ignored, whereas at the same time genuine defects

need to be identified. Although it has not been proven by the author, with the availability of CAD information about locations and widths of tracks, the AOI system should be able to automatically zoom in when inspecting a fine track so as to achieve a higher resolution, and zoom out when inspecting a thick track so as to reduce the amount of data need to be processed.

8.6.6 The role of the Proposed AOI Reference Architecture in This Case Study

The AOI reference architecture proposed and described previously in this chapter (and chapter 7; see Figure 7.5) has helped the author to design and structure the AOI algorithms used in this case study. As discussed in chapter 7, the design information extracted from a typical CAD system (such as the PCAD system used by the author) is inherently hierarchical. Thus the author believes that better use of this information can be achieved if the AOI system is designed with an architecture that reflects the natural hierarchical nature of the process of performing inspection tasks and the information used and generated by the AOI system. In this case study, the author has followed the AOI architecture he proposed and recommended when building the proof of concept AOI system (e.g. in determining the various algorithms needed to perform the required inspection tasks; the suitable levels to which particular algorithm classes should be assigned and thus the data structures and formats used to represent the input to and output from those algorithms; the information support required by particular algorithms; the actual features extracted by the algorithms; etc.).

For example, a routine (WDOWGEN) has been devised which operates at the bottom level (i.e. Primitive Routines level) to read in CAD information (i.e. directly read information from the local product model) and to generate a file of image-capturing locations, where the contents of the file can be used to control the movement of the robot (and thus the location of the camera). This approach enabled the entire PCB to be analysed automatically in a section-by-section manner and to achieve this result with a higher resolution as required. Similarly, at other levels of the architecture, algorithms have been designed for the purpose of performing

inspection tasks, while making as much use as possible of the CAD information support. In total, the author has created, used and proved a total of 25 or so algorithms, each operating within the reference architecture proposed. Many of these algorithms necessarily had been implemented with novel features to reflect the ability to utilise supporting information or to provide information feedback.

8.7 Summary

This chapter has described the author's partial implementation of the "product model based approach to integration" (advocated in Chapter 6) with particular application focus on proving concepts in respect to "AOI system reference architecture" (proposed in Chapter 7) and its use in PCB inspection. A specific application case study has been described which serves to illustrate how the author's proof of concept information supported machine vision system can be applied in the inspection of PCB artwork.

Global and local physical product models have been implemented in the context of providing information support to advance automatic bare board inspection methods. As outlined earlier, at the global level it is desirable and potentially beneficial to employ neutral data structures and information representational format (NDSIRF). Information stored in a local product model is intended to be directly accessible and usable by the local application concerned. Once the data structure and the information representation format has been devised and considered suitable for use with the local application, a SIG can then be designed which extracts the required information from the global product model data and presents it to the local subsystem. It should be pointed out that no formal implementation has been done regarding global and local conceptual models as defined in Chapter 6. However, the idea of using conceptual models has been utilised in structuring the steps followed by the author when designing the global and local physical product models, especially in the sense of determining the structure of information items for each of the

physical models implemented, as illustrated by Figure 8.18. Further research is recommended into various aspects of such work.

The case study presented in last section has demonstrated the feasibility and benefits of using CAD information to support AOI applications. One of the key conclusions that can be drawn from which is that proper use of such information support can lead to improved AOI performance, for example, in terms of shortened system set-up times, and better and more reliable inspection results. As CAD information is in nature defect free, it provides a more reliable “template” with which real product can be compared.

The implementation work presented in section 8.4 as well as the case study described in section 8.6 are mainly demonstrative. Necessarily they have also been fairly sharply focused, as the methodologies and reference architecture can have very wide implications. Thus but one of the many application areas of AOI systems has been investigated; this area being the so-called “panel inspection” typified by bare board inspection, layer inspection and artwork film inspection (see Chapter 7). Furthermore, the implemented algorithms are mainly in the category of “binary image processing”. It should also be pointed out that studies relating to information support of AOI have largely limited to the area of re-utilising available CAD information. Despite the need for focusing, the components of the proof of concept system have provided a vehicle for testing much broader concepts.

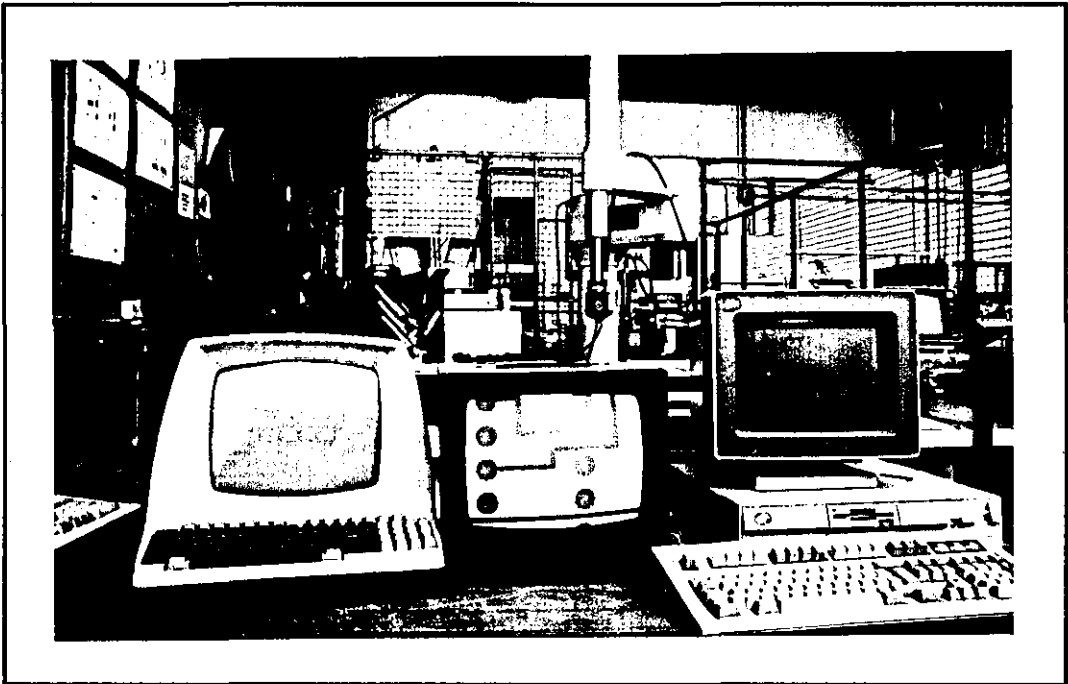


Photo 8.1 An Example of PCB Artwork

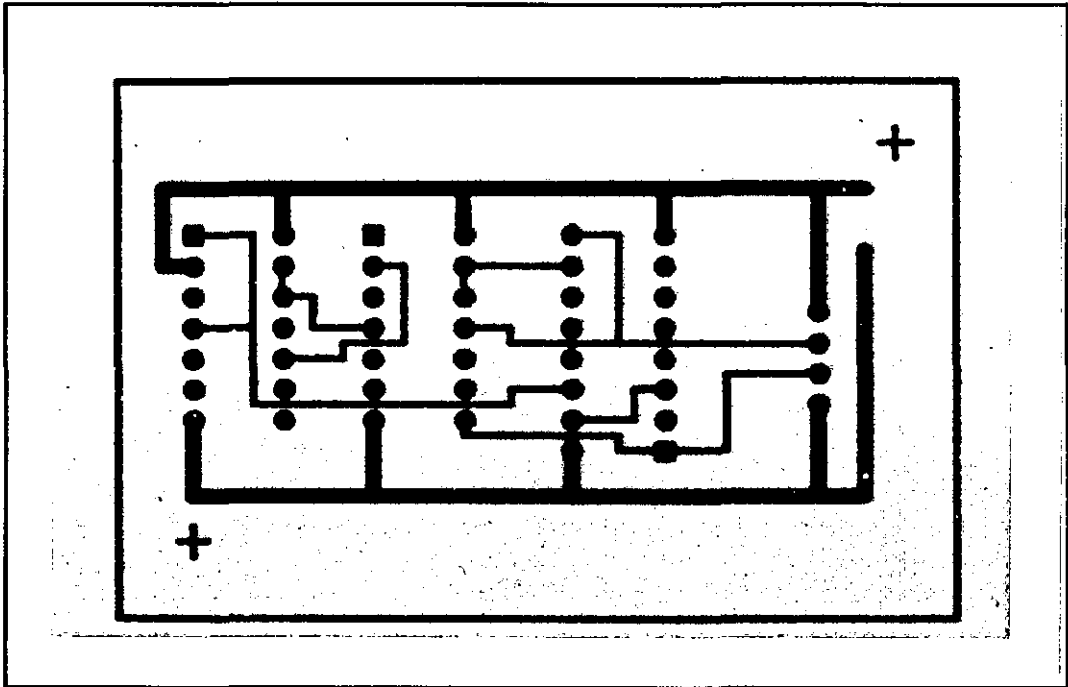
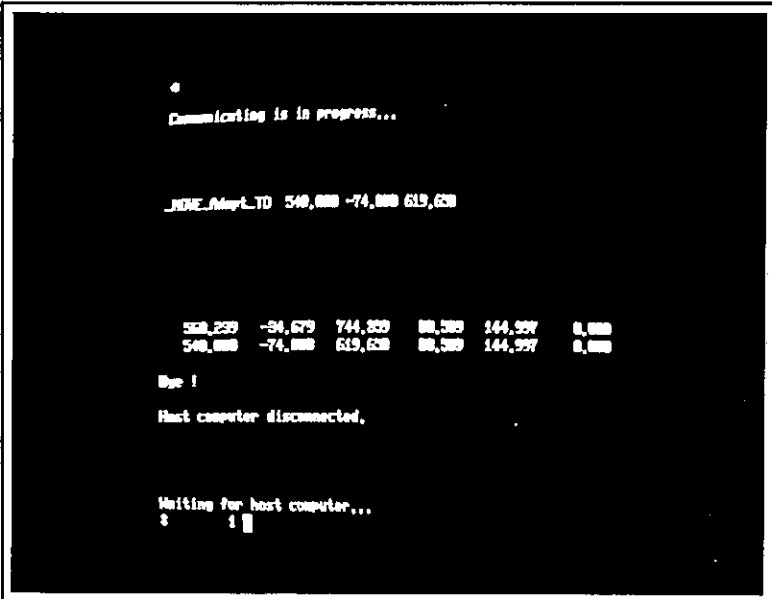


Photo 8.2 System Setup

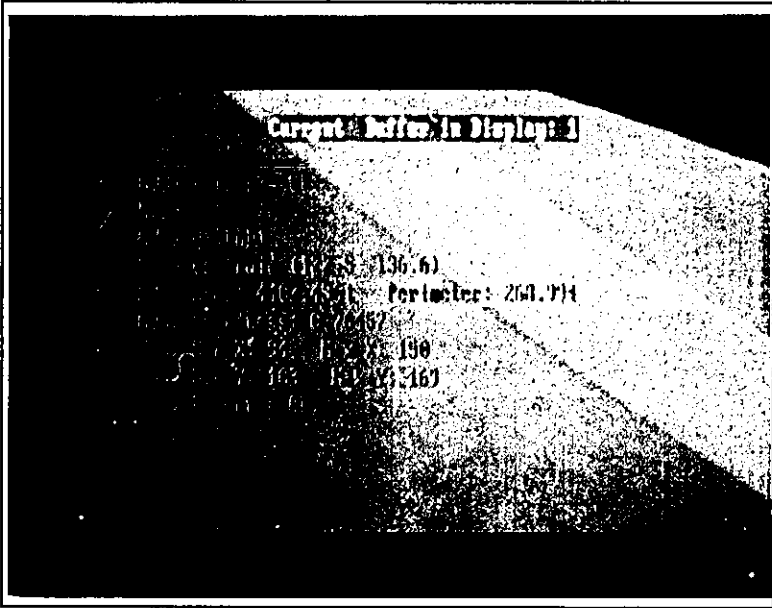
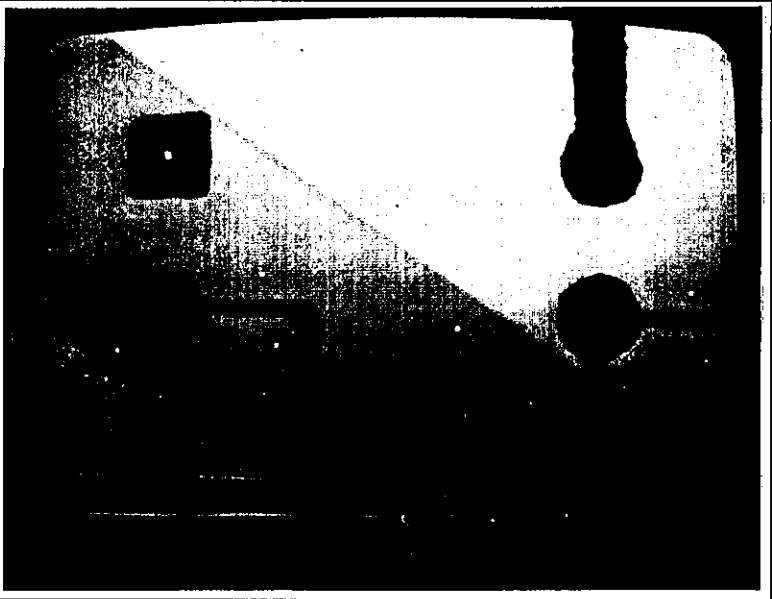


← Photo 8.3.a

An example of robot vision communications message displayed on the monitor: the 2nd line showing the command received from vision; the 3rd and 4th line showing robot locations before and after executing the command respectively

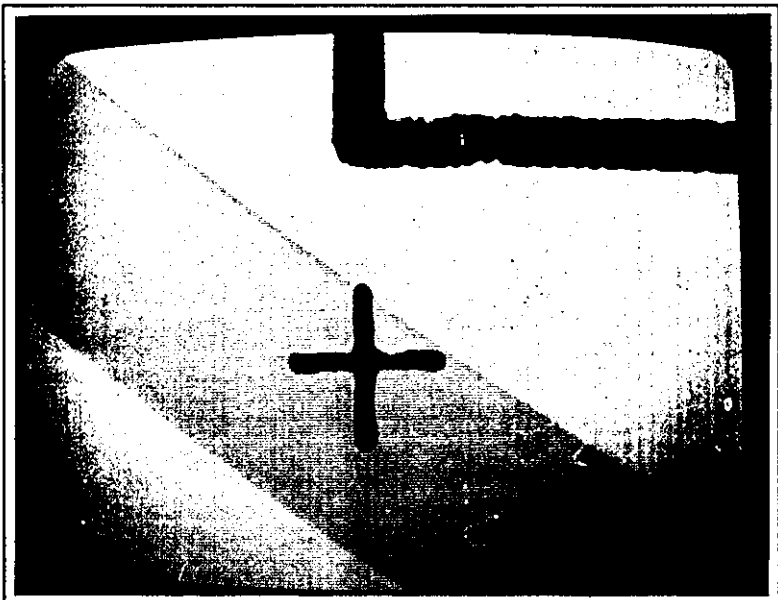
Photo 8.3.b →

An example of images being processed and analysed in the author's case study



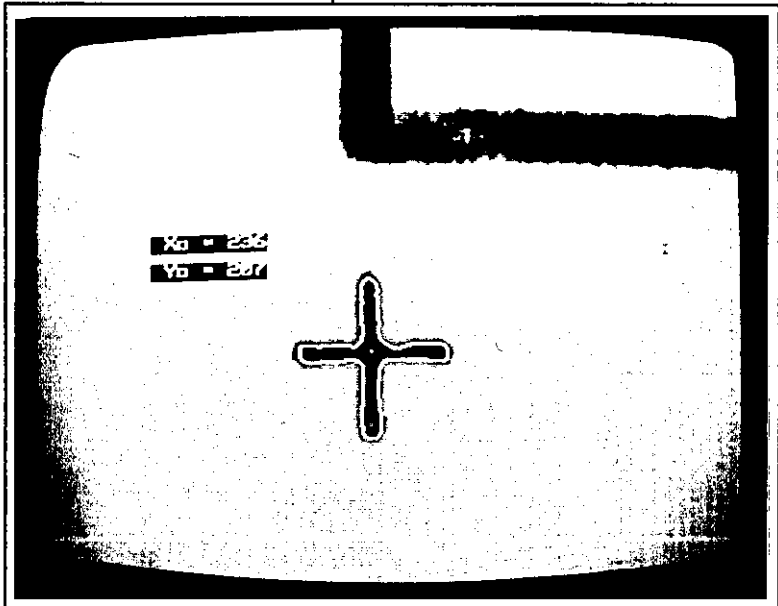
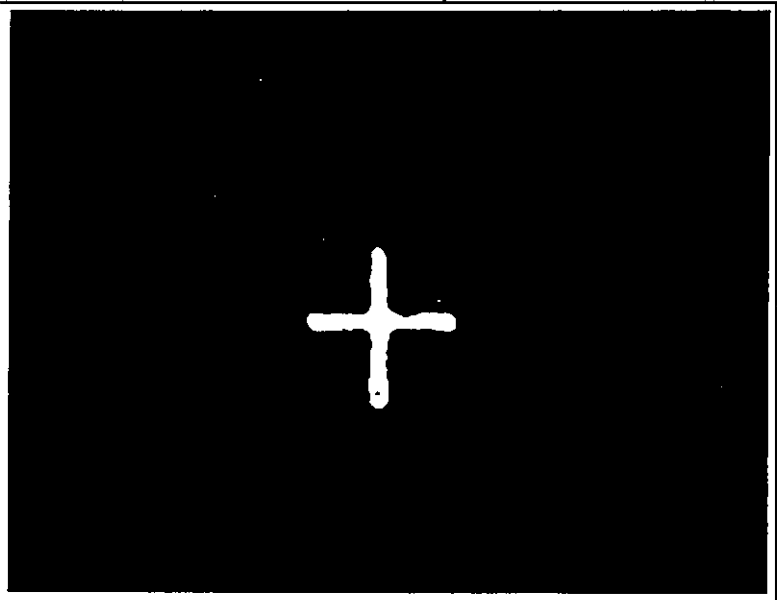
← Photo 8.3.c

An example of vision system operating commands, here showing the command "QX" (QUIKEXTRA) and the set of simple features extracted from chain code representation.

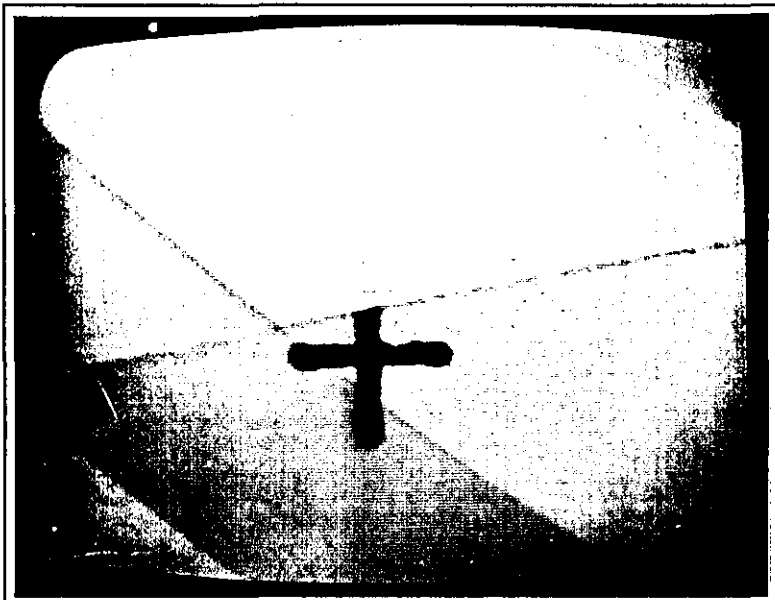


← Photo 8.4.a
Grey level image of a normal (good) fiducial mark.

Photo 8.4.b →
Binary image of the same fiducial mark after thresholding.



← Photo 8.4.c
Centre point of the same fiducial mark located using FIDLOC, coordinates are given in pixels.

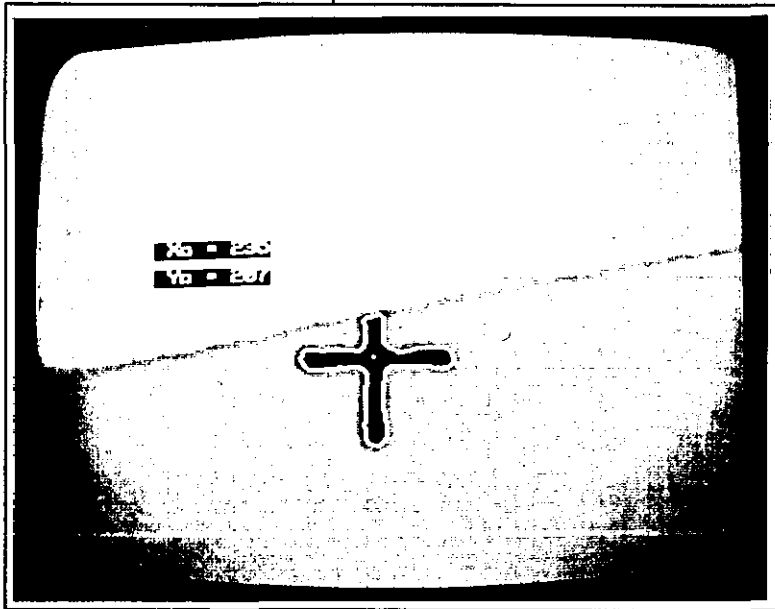
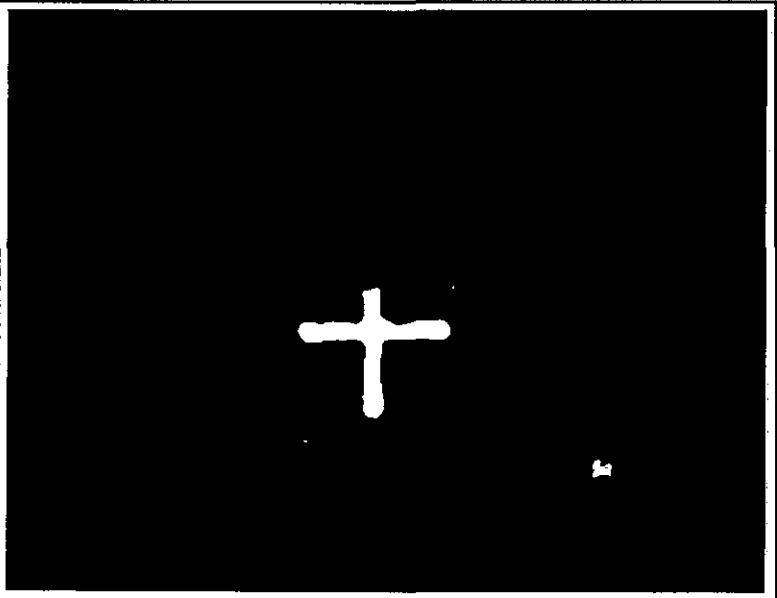


← Photo 8.5.a

Grey level image of a deformed fiducial mark (where one branch is broken).

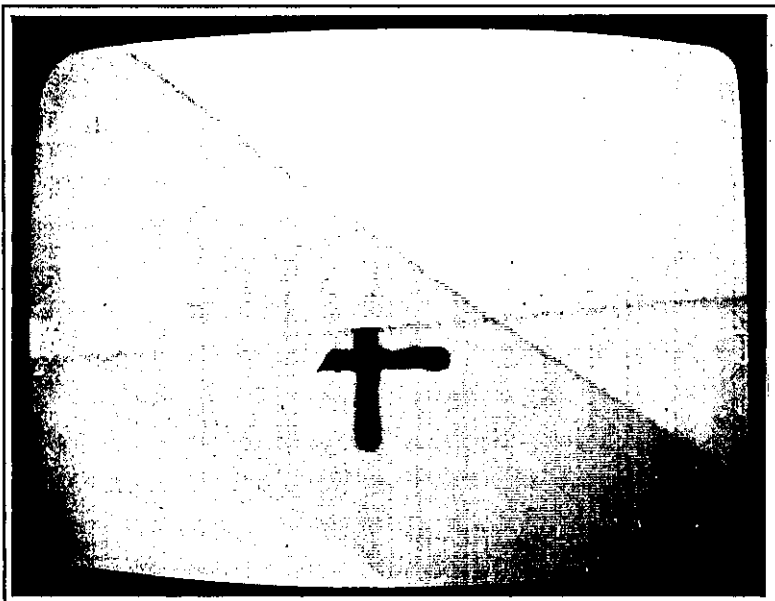
Photo 8.5.b →

Binary image of the same fiducial mark after thresholding.



← Photo 8.5.c

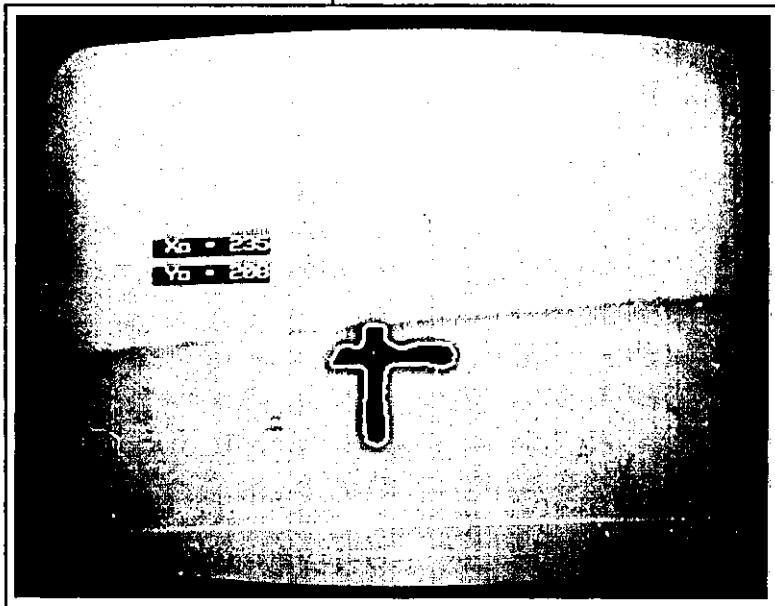
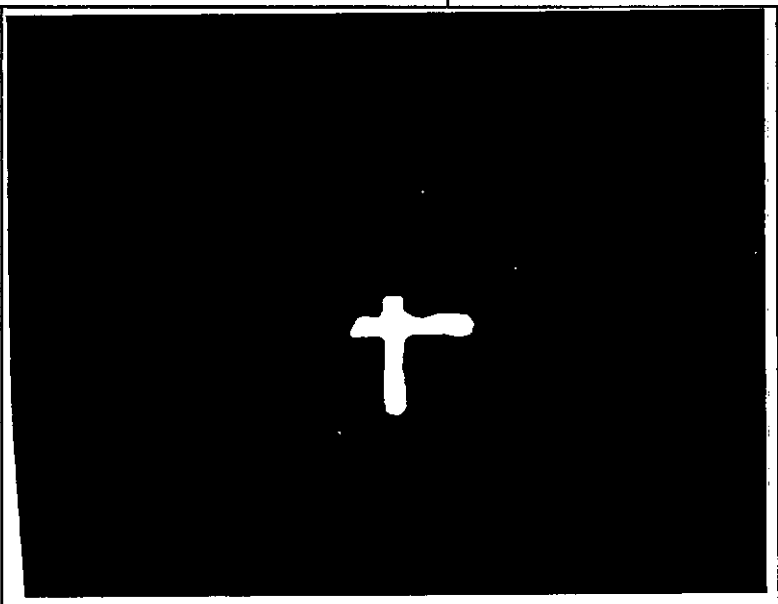
Centre point of the same fiducial mark located using FIDLOC, coordinates are given in pixels.



← Photo 8.6.a

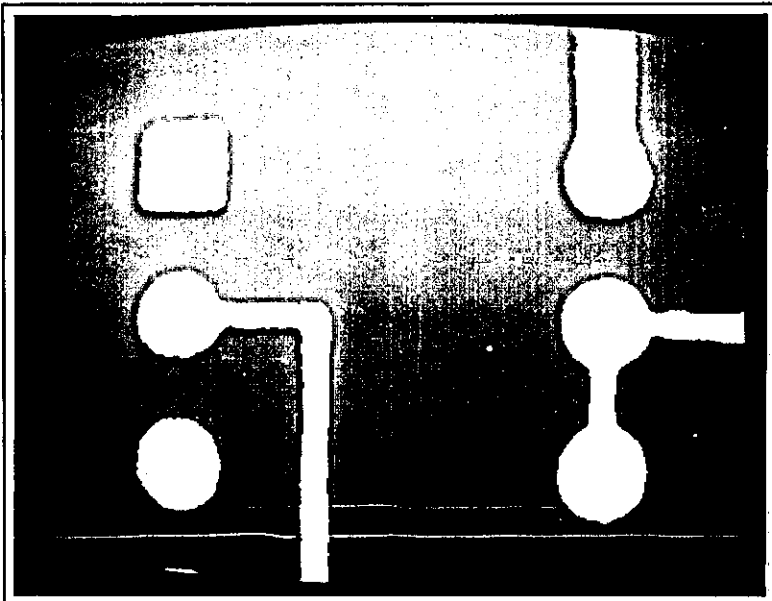
Grey level image of another deformed fiducial mark (where two branches are broken).

Photo 8.6.b →
Binary image of the same fiducial mark after thresholding.



← Photo 8.6.c

Centre point of the same fiducial mark located using FIDLOC, coordinates are given in pixels.



← Photo 8.7.a

Binary image of a sample section of the PCB artwork shown in Photo 8.1.

Photo 8.7.b →

A set of local features extracted by LFEXTRA from the chain code representation of a square pad.

```
Centroid (122.5 136.6)
No. of Corners (proposed): 4
Roundness: 0.7646
Area from chain code: 4462.44
Area from circle_match: 4452.38
Relative error in area: 1.134%
Circle_match error: 3.3886
Line_match error: 2.3589
LM_error / CM_error: 0.712
```

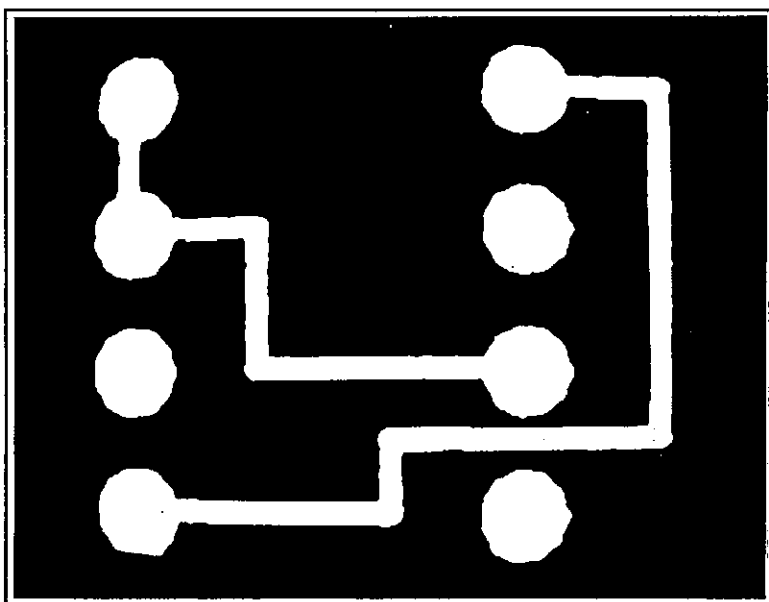
Press any key . . .

```
Centroid (118.1 351.6)
No. of Corners (proposed): 4
Roundness: 0.8585
Area from chain code: 3376.84
Area from circle_match: 3379.65
Relative error in area: 0.065%
Circle_match error: 0.7101
Line_match error: 6.6882
LM_error / CM_error: 9.386
```

Press any key . . .

← Photo 8.7.c

A set of local features extracted by LFEXTRA from the chain code representation of a circular pad.

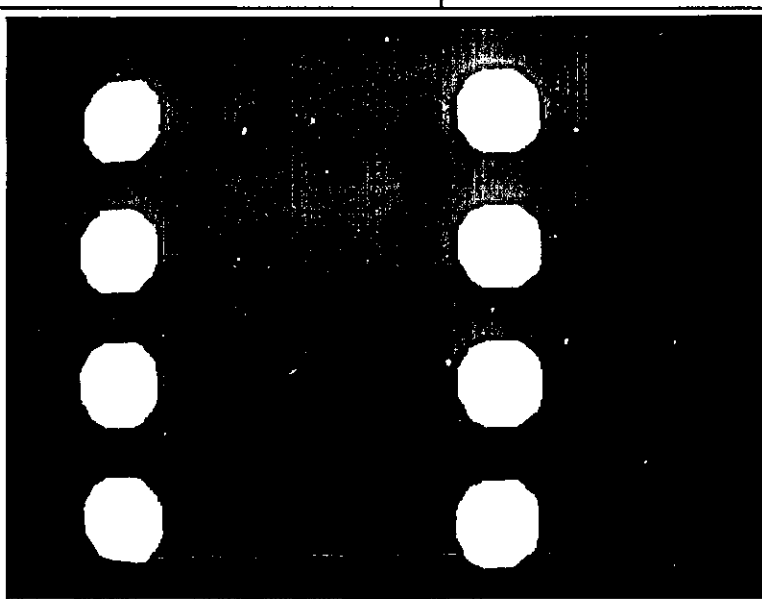


← Photo 8.8.a

Binary image of another sample section of the PCB artwork shown in Photo 8.1.

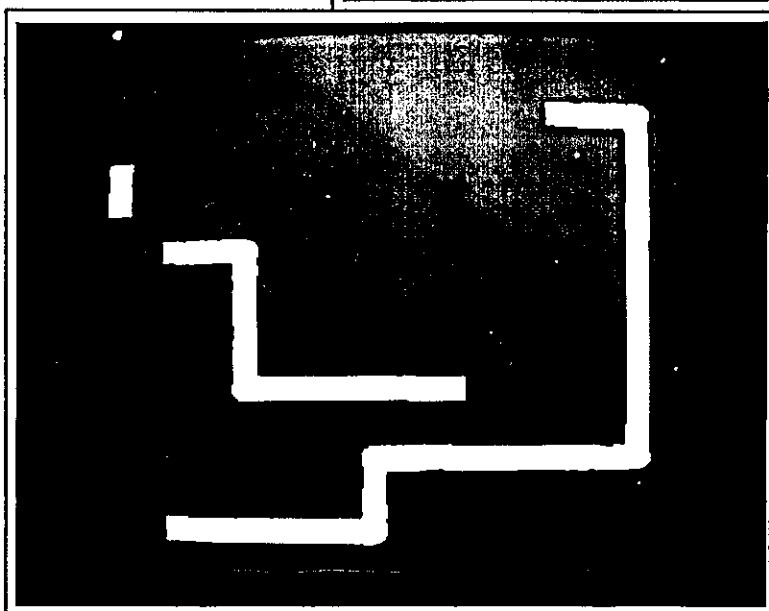
Photo 8.8.b →

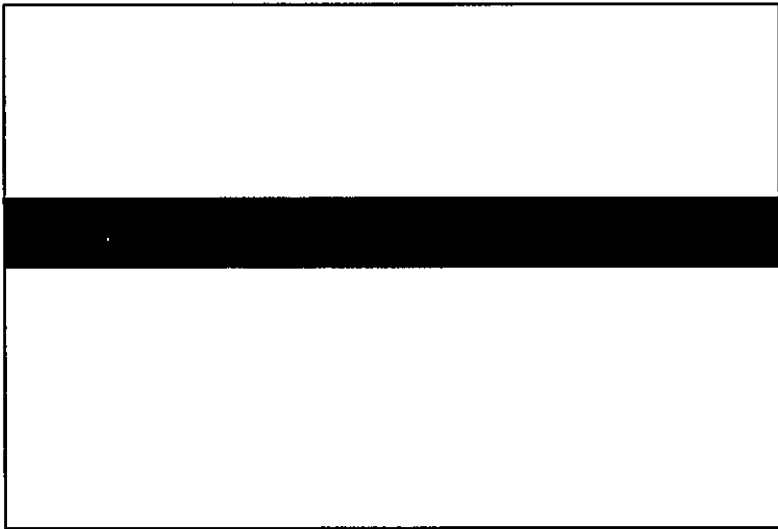
Result of performing an "open" operation on the image shown in Photo 8.8.a to remove track from the image.



← Photo 8.8.c

Result of performing an alternative operation on the image shown in Photo 8.8.a to remove pads from the image.



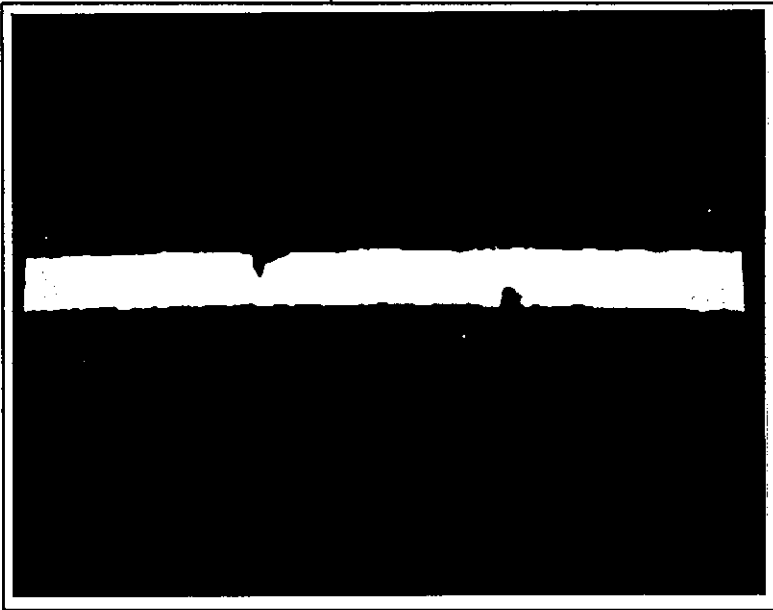
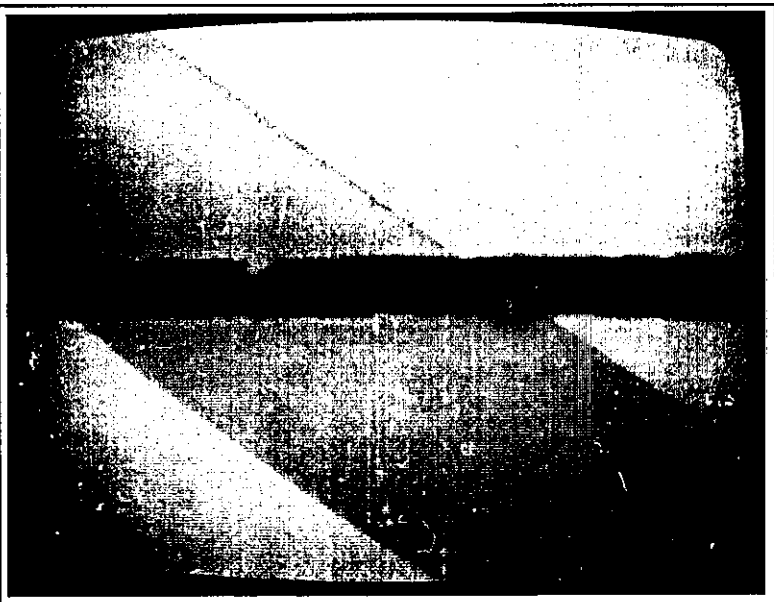


← Figure 8.17

A graphical representation of a sample segment of an ideal track specified by the CAD information.

Photo 8.9.a →

Grey level image showing a track segment from the plotted PCB artwork (with artificial defects added).



← Photo 8.9.b

Binary image of the same track segment showing the digitising noise along the edge of the track segment. High resolution is needed to make the noise ignorable.

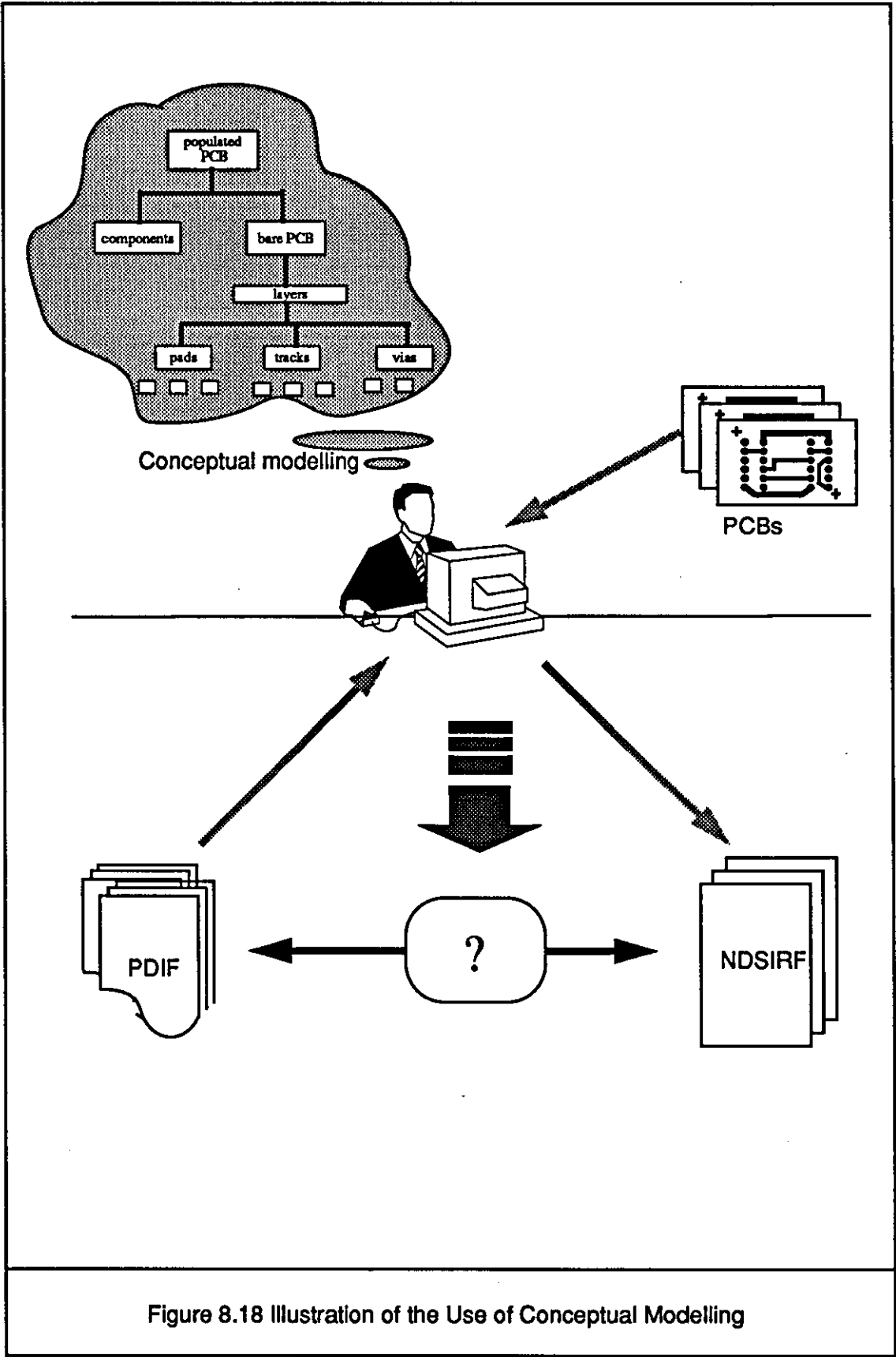


Figure 8.18 Illustration of the Use of Conceptual Modelling

Chapter 9

Conclusions and Recommendations

9.1 Conclusions

Although it is still in its infancy, computer vision technology has already had a significant impact on manufacturing methods in various industrial sectors. In particular, AOI (automatic optical inspection) has found an important role to play in the arena of automatic PCB (printed circuit boards) inspection. The main body of this thesis reports on a study which has been focused on advancing the functional capabilities and hence extending the potential application areas of AOI systems, with particular reference to the processes involved in PCB manufacture. In achieving this objective the author has

- Evaluated the role of AOI and its potential applications in PCB manufacture; which concluded that the use of the AOI should by no means be confined to ensuring that faulty PCBs do not leave the factory and enter the marketplace, and to prevent value being added to defective boards (i.e. by facilitating the detection and discard of faulty layers as early as possible). It should also be used to provide information support for statistical process control, or other local processes such as PCB assembly.

- Classified common features of present generation AOI systems; where the classification has highlighted the need for novel approaches to integration and hence means of achieving informa-

tion support of AOI processes.

- Conducted preliminary empirical research into systems integration and information support mechanisms; this involving the integration of vision system/robot manipulator couple and that of a vision system with a proprietary CAD system.
- Added to the understanding of the role of product models in achieving information support to local processes (such as AOI); by proposing and partially implementing a *product model based approach* to flexibly integrating AOI systems within computer integrated PCB manufacturing.
- Investigated and classified various generic characteristics of typical AOI applications in PCB manufacture; the work leading to a proposal and partial implementation of a reference architecture for structuring the design and implementation of AOI algorithms based on a concept of AOI task decomposition.

2.2 Contribution to Knowledge

9.2.1 The Product Model Based Approach

Based on the experience drawn from the study of systems integration and hence facilitating information support of AOI processes, a “*product model based approach*” has been proposed to systemise the process of integrating AOI systems in computer integrated PCB manufacture. With this approach, product models are employed as a means of providing, in a flexible way, adequate information support to individual local applications. In order to overcome the disadvantages which result from the use of proprietary data structures and information formats (the use of which is commonplace with the current component elements of CIM systems), the

use of a neutral data structure and information representational format (NDSIRF) for representing global information (i.e. stored in the global physical product model) potentially stored as a number of information fragments at various computer systems was conceived by the author.

To demonstrate the benefits of using such an approach, a custom NDSIRF was defined and implemented in the proof of concept partial implementation. From the literature survey it is clear that much research has been targeted on the broader issue of integration methodologies and reference framework. However, little of this work has been centred specifically on addressing the integration needs of machine vision and, more specifically, AOI. Hence as a basis of comparison, the author has taken as a benchmark the notion of a bespoke (or custom designed) integration couple of the type described in chapters of four and five; namely, the pair-wise integration approach.

The improvement in “flexibility” offered using the author’s *product model based approach* can to some extent be qualified with respect to the following perspectives,

- Change of one or more end systems (e.g. change of the CAD system, AOI system, or component placement machines, etc.)

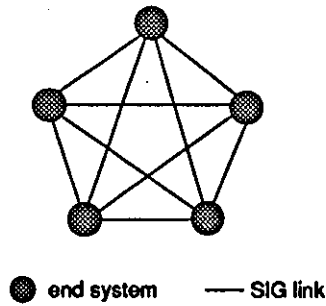
When an end system is changed (e.g. removed, added, or modified), the effect can be considered with respect to two types of change.

Firstly, if a local system is replaced (or simply modified or upgraded), but the same information requirements and generation capabilities remain unchanged, the local change will have no effect on other end systems nor indeed require change to its associated SIG (software information generator); for example, if a different CAD system is used but still produces design informa-

tion using similar data structures and representational formats as its predecessor, there will be no effect on other end systems.

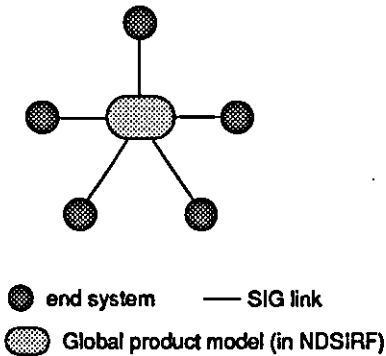
However, a second type of change can be characterised by changed information requirements as a result of modification or upgrading of the end system. Here there will be a need to modify the SIG used by the system component. However, as mentioned earlier (see chapters of six and eight), such a modification need not necessarily imply a total re-engineering of the SIG; for example, if a local system requires information support through accessing the global product model, then at least the “*lexical analyser*” (that interfaces to the global product model) can be reused without modification (assuming the global product model has not been changed). If a new CAD system is introduced for which the design information is represented using different data structures and formats, only a new SIG needs to be designed which can extract information from such a CAD system and represent the extracted information in the defined NDSIRF used with the global product model (other systems will not be affected).

Clearly the integrated system using the *product model based approach* exhibits much improved flexibility (in terms of less need for software re-engineering) against the second type of change mentioned above, as compared with the conventional “pair-wise approach”. See Figure 9.1. As the number of end systems increases, so will the potential benefits of the author’s approach. It is important to point out that there will be an initial overhead required for providing an integration infrastructure using the author’s approach (i.e. a minimum of two SIGs, and a need for designing the NDSIRF). For example, when *only* two end-systems are to be integrated, the bespoke pair-wise integra-



(a) The Pair-Wise Approach
(5 end systems; 10 SIGs needed)

- * If n end-systems are to be integrated using the pair-wise integration approach, a total of $n(n-1)/2$ SIGs will be needed (initial effort);
- * If one end system is changed, all other end-systems connected with it will be affected; a total of $n-1$ SIGs will need to be modified or re-engineered (effort for change);
- * If a new end-system is added and integrated with the existing n end-systems using the pair-wise approach, a total of n extra SIGs will be required (effort for system expansion).



(b) The Product Model Based Approach
(5 end systems; 5 SIGs needed)

- * If n end-systems are to be integrated using the product model based approach, a total of n SIGs will be needed (initial effort);
- * If one end-system is changed, the rest will remain unaffected; only the SIG associated with the changed end-system will need to be modified (effort for change);
- * If a new end-system is added and integrated with the existing n end systems using the product model based approach, only one extra SIG will be required (effort for system expansion)

Number of end systems to be integrated	Total number of SIGs needed		Number of SIGs need to be modified when one end system is changed	
	Pair-wise approach	Product mode based approach	Pair-wise approach	Product mode based approach
2	1	2	1	1
4	6	4	3	1
5	10	5	4	1
10	45	10	9	1
20	190	20	19	1
n	$n(n-1)/2$	n	$n-1$	1

(c) Table of comparison

Figure 9.1 Comparison of Two Different approaches to Integration

tion approach seems to be simpler than the author's approach. However, the author believes that such a "two-only" (or even "three-only") integration requirement will not likely be prevalent among modern manufacturing industries. On the contrary, there will always be a need for changes in system configuration and/or for system expansion.

-- Change of the global data structure and information format (i.e. the NDSIRF) used in representing information stored in the global physical product model.

Such a change is likely to have great impact on the integrated system, as it will affect all end systems; this being one disadvantage of implementing the global product model in the form of flat files. The effect of change to the NDSIRF can be alleviated to some extent using a database (such as a relational database) to store the *global physical product model*, as such storage mechanisms can provide, in theory, end systems with "standard" methods (i.e. independent of the physical data structure, format and storage medium) of accessing information [Gray 1991]; for example, the use of high level query languages [Beyon 1990] such as the Sequential Query Language (SQL) would be enabled. Had more time been available, the author would have re-implemented the product models using a database approach. Thus further investigation into this issue is recommended. However, even with the use of flat file storage mechanisms better control of information (in terms of representation, storage and access issues) can be realised using the author's approach as opposed to pairwise approaches to integration.

-- Introducing a new product.

Typically, the introduction of new PCBs will not necessitate change in the data structures and information formats used to represent such products. This will be the case for both the global and local product models, provided that the previously defined data structures and information formats are sufficiently generic to represent the information required to define the new product. If this is the case, obviously re-engineering of the software used to enable information sharing and/or support will not be necessary. However, if new information items which have not been considered previously need to be included, in either local or global product models, modifications will be required (e.g. re-design the data structure and information format, or various SIGs used, to suit the new requirements). In the case of modifying the SIGs, the effort involved will be proportional to the number of SIGs concerned. As the number of SIGs needed with the *product model based approach* is much less than that needed when using conventional pair-wise approach, especially so when the number of end systems is greater than five (refer back to Figure 9.1), the effort involved in re-engineering of the SIGs will be much less using the former approach than using the latter approach.

Furthermore, the improvement of flexibility will be evident when considering the expansion of the integrated system, i.e. adding in a new end system. Obviously when using the *product model based approach*, only one extra SIG will be needed for the newly added end system in order for it to share information with the rest of the end systems, whereas when using the “pair-wise” approach, the number of new SIGs required will be equal to the number of end systems with which the new end system wishes to share information. Essentially relative advantages of this type can

be viewed as being an inheritance of a structured approach to integration which decouples the effect of change through appropriate decomposition of the solution.

Thus in broad terms the proposed *product model based approach* can be viewed as a methodology for creating integrated systems which are essentially application-context dependent (i.e. dependent on information requirement of classes of local applications), rather than being proprietary-system dependent; the later being the typical result of using conventional rigid approach to integration. This results in much improved robustness and greater flexibility when realising information sharing and information support.

9.2.2 A Proposed AOI Reference Architecture and Its Partial Implementation

The driving force for proposing and using a reference architecture to structure the design of AOI systems stems from the following observations made and opinions held by the author, viz:

- The potential benefits of using information supported AOI will not be realised to any great extent until AOI systems themselves are restructured to take full advantage of such opportunities. Generally present generation AOI systems are designed and marketed to operate in a stand-alone fashion, thus they cannot make ready use of useful information which resides within many other computerised systems. In addition, AOI systems can themselves be important generators of information. Therefore in their dual role as consumer and producer of information, next generation AOI systems should be designed to suit emerging systems integration requirements. As a subset of the more global requirements, new approaches to AOI and methodologies relating to the re-utilisation

of information created during product design and collected during product realisation should also be sought.

- In the context of automatic PCB inspection, the author believes that there are many similarities evident in inspection processes. Thus such processes lend themselves to potential simplification through appropriate task decomposition.

Such a decomposition can be based on the observed hierarchical nature exhibited, this being evident in terms of the physical construction of PCB products, the product design information maintained in CAD systems and in terms of manufacturing information typically stored in various CAM stations. Thus it is suggested that decomposition of PCB inspection tasks should reflect such hierarchical characteristics, leading to better and more efficient use of the available information and hence an advancement of various inspection tasks. This again necessitates the need for a reference architecture that can be used to provide design guidelines for a new generation of AOI systems.

Through an examination of the generic characteristics of typical AOI applications in PCB manufacture, the author has proposed a method of decomposing the general AOI inspection task into two component inspection stages, namely BE (board element) level inspection and board level inspection (and/or analysis). Based on such a decomposition, an *AOI system reference architecture* has been proposed which can be used to structure the implementation of AOI algorithms in order to facilitate flexible exchange of information between AOI and CAD/CAM systems and thus better re-utilization of available CAD/CAM information by AOI system.

The proposed reference architecture has been partially implemented in order to evaluate the feasibility of using such a reference architecture to structure the design and implementation of AOI algorithms. Although in the partial implemen-

tation, only selected algorithms have been implemented and tested, the problems addressed are of a generic nature. Those algorithms implemented have various novel features when compared with those used in stand-alone AOI systems. These features have been devised and included to facilitate and benefit from application requirements for information sharing (with the AOI system functioning as both consumer and producer of information).

Clearly it has only been possible to broadly qualify (as opposed to quantify) the benefits of the AOI system reference architecture proposed, nor has it been possible to fully specify and quantify characteristics of the algorithms at each level. However, it is suggested that the proposal and its partial proof of concept implementation may represent an important contribution to knowledge which following researchers and IT (information technology) system implementors could usefully utilise. In particular the author has conducted a more in-depth investigation into methods for re-utilising information typically available in CAD/CAM systems. For example, CAD/CAM information has been utilised for generating window parameters (**WDOWGEN**) and supporting the localisation of fiducial marks (**FIDLOC**); this leading to the automatic BE (board element) inspection being enabled. Thus it is believed that this work could ultimately lead towards the building of next generation open and information supported AOI systems.

Generally speaking, different AOI applications will have to satisfy different requirements, hence it may be argued that the optimum structural design of AOI algorithms will be application dependent. However, to facilitate information sharing and information support between AOI and other PCB product realisation processes and systems there is a definitive need for some general guidelines which will systemise system building without being necessarily restrictive. The methods proposed for decomposing AOI tasks and the proposed and partially implemented AOI reference architecture for building AOI systems are intended to serve this purpose. Furthermore, some of the algorithms (or functional modules) created and implemented by the author may be reused when building new AOI systems; good examples of

this are algorithms relating to CAD information re-utilisation (e.g. **WDOWGEN**, **FIDLOC**, etc.), chain code generation (e.g. **CHAINCODE**), corner point detection (e.g. **CNRPOINT**), geometry fitting (e.g. **CIRMATCH**, **LMATCH**), track inspection (e.g. **INSPECONN**, **CHEKCONN**), feature extraction (e.g. **LFEXTRA**, **QUIKEXTRA**), geometrical relationship analysis (e.g. **GEANALYS**, **COMPANET**), inspection result processing and analysis (e.g. **ERRTREND**, **ERREPORT**, **ERRDISP**), calibration of the scale factors of the optical system (i.e. **VISCAL**).

Another novel feature of the proposed AOI system reference architecture (as compared with more conventional stand-alone, closed AOI architecture) is that it offers further opportunities for AOI generated information to be easily accessed and processed to support various application requirements. This is facilitated by representing AOI generated information (i.e. the global board features) in a neutral (ideally standard) format, and by the inclusion of application-oriented algorithms at the top level of the AOI software hierarchy which can be used to extract desired information from the global inspection results and to present the extracted information in a format suitable for the particular application. It is important to point out that the ultimate realisation of the potential benefits of re-utilising AOI generated information will again depend on the intended local process or system, meaning that it will depend on how the information is utilised within the local process or system concerned, not depending on the AOI system. For these reasons, only three simple representative application-oriented algorithms (AOAs) (i.e. **ERREPORT**, **ERRTREND** and **ERRDISP**) have been implemented to demonstrate the concept. Obviously much more in-depth investigation, from a local process perspective, into the issue will be required in order to provide more sophisticated AOAs which can cater for the specific local requirements (i.e. to achieve more sophisticated re-utilisation of AOI generated information). Necessarily this work has been outside the scope of this PhD study and thesis.

9.3 Recommendations for Future Work

Although AOI systems have been used for around a decade, this is still a far from mature application area of computer technology. New methodologies and systems are required to address many of the outstanding problems related to AOI.

Even the notion of AOI systems utilising information from individual sources such as CAD is very much in its infancy, not to mention the wider issue of achieving fully information supported AOI in the much wider context of computer integrated manufacturing. As such, a variety of research areas need to be addressed to enable information supported AOI to be fully realized and practically applied. The author's primarily recommendations for future work are listed below:

- (1) As far as the proposed AOI reference architecture is concerned, additional research work needs to be carried out to verify its use in complementary application areas which demonstrate differing information consumption and production requirements. Work is also required in more completely defining the required functionality of the algorithms at each level, especially at the highest level (i.e. the application oriented algorithms). Obviously, this itself would involve an investigation of the information requirements of additional local AOI applications and of other classes of local application requiring AOI to assume a role of information producer; for example, robotic placement of components, or statistical process control of say etching (e.g. based on layer inspection results), soldering (e.g. based on solder joint inspection results), solder paste application (e.g. based on solder paste application inspection results), etc.
- (2) Fully information supported AOI can be taken to imply a maximum possible re-utilisation of information available from other

systems and processes. Thus studies are recommended by the author, which require the collection and modelling of process variables that have potential effects on the quality of the work-in-process (e.g. layers of PCBs); for example information that reflects properties or characteristics of manufacturing processes such as drilling, etching, plating, lamination, solder paste application, automatic component insertion/onsertion/assembly. The availability of information relating to such “upstream” processes, coupled with the design information from CAD systems, could advance the industrial use of information supported AOI through realising more effective and reliable inspection tasks.

- (3) Further evaluation of the role of product models, acting as “information pools” to provide adequate information support capability to individual local applications, including AOI. As mentioned in subsection 9.2.1, research relating to other methods of implementing product models is also recommended: where they can be represented using data modelling language (such as EXPRESS) and stored in a more structured and easily managed fashion (such as a relational database).

REFERENCES

- [Adept 1985] -- "Adept manipulator system VAL-II reference guide" and "Adept manipulator system basic operation", 1985 by Adept Technology, Inc., 1212 Bordeaux Drive, Sunnyvale, CA 9089, USA
- [Ahuja 1979] Ahuja, V. "Routing and flow control in systems network architecture", IBM Systems Journal, Vol. 18, No. 2, pp 298-314, 1979
- [Amam 1986] --- "Putting MAP to work", American Machinist & Automated Manufacturing, pp 75-9, January 1986
- [Anderson et al 1990] Anderson, A., Jene, T. & Mikkilineni, K. "CIM architecture: One perspective", Proceedings of CIMCON '90, pp 506-24, 1990
- [Babb 1987] Babb, M. "Data acquisition for control: Opening up to computer architecture", Control Engineering, VOL. 34, No. 2, pp 85-90, February 1987
- [Bailey 1989] Bailey, S. J. "Factory sensors rise to the need of cell, line, and network", Control Engineering, Vol. 36, No. 15, pp 55-7, December 1989
- [Balius 1990] Balius, P. "CIM: the Next Step", PC Fab., pp 22-37, July 1990
- [Ballard and Brown 1982] Ballard, D. H. & Brown, C. M. "Computer Vision", Prentice-Hall, Inc. 1982
- [Bartlett et al 1988] Bartlett, S. L., Besl, P. J., Cole, C. L., Jain, R., Mukherjee, D. & Skifstad, K. D. "Automatic solder joint inspection", IEEE Trans. Pattern Anal. Mach. Intelligence, Vol. PAMI-10, No. 1 pp 31-43, 1988
- [Batchelor 1985a] Batchelor, B. G. "Lighting and viewing techniques", in Batchelor, B. G., Hall, D.A. & Hodgson, D.C. (eds) "Automated visual inspection", IFS publication, pp103-179, 1985
- [Batchelor 1985b] Batchelor, B. G. "Principles of digital image processing", in Batchelor, B. G., Hall, D.A. & Hodgson, D.C. (eds) "Automated visual inspection", IFS publication, pp329-400, 1985
- [Batchelor et al 1985] Batchelor, B.G., Hall, D.A. & Hodgson, D.C. (ed.) "Automated visual inspection", IFS publication, pp 479-534, 1985
- [Bauer and Alt 1990] Bauer, M. & Alt, L. "CIM at Work", PC Fab., pp 80-87, Aug. 1990
- [Beeckman 1989] Beeckman, D. "CIM-OSA: Computer integrated manufacturing - Open systems architecture", Int. J. of CIM, Vol. 2, No. 2, pp94-105, March/

April 1989

- [Benhabib et al 1988] Benhabib, B., Charette, C., Park, S., Williams, R. & Smith, K.C. "An automatic inspection system development for printed circuit boards: Image acquisition and filtering", Proc. of IASTED Int. Conf. on Reliability and Quality Control, pp170-173, Paris 1988
- [Benhabib et al 1990] Benhabib, B., Charette, C.R., Smith, K.C., & Yip, A.M. "Automatic visual inspection of printed circuit boards: An experimental system", Int. J. of Robotics & Automation, Vol. 5, No. 2, pp 49-58, 1990
- [Bentley 1980] Bentley, W. M. "Automated optical inspection of multilayer printed circuit boards", Proc. IEEE Optics in Metrology and quality Assurance, Vol. 220, pp 102-9, 1980
- [Besant and Lui 1986] Besant, C. B. & Lui, C. W. K. "Computer-aided design and manufacture", 3rd edition, Ellis Horwood, 1986
- [Besl et al 1985] Besl, P.J., Delp, E.J. & Jain, R. "Automated visual solder joint inspection", IEEE Trans. Pattern Anal. Mach. Intelligence, Vol. PAMI-10, No.2 pp 167-192, 1988
- [Beyer 1990] Beyer, G. H. "Aspects of automation in printed wiring board manufacturing", Proceedings of Printed Circuit World Convention 5, pp (C11/1-1) -(C11/1-10), 12-15 June 1990
- [Beyon 1990] Beyon, D. "Information and data modelling", Blackwell Scientific Publications, 1990
- [Blakeslee 1989] Blakeslee, D.A. "Verification: The neglected portion of AOI", PC Fab., pp24-30, Dec. 1989
- [Bloor and Owen 1991] Bloor, M. S. & Owen, J. "CAD/CAM product-data exchange: the next step", Computer-Aided Design, Vol. 23, No. 4, May 1991, pp 237-43, 1991
- [Böhms and Tolman 1990] Böhms, M. & Tolman, F. "RIA: Reference model for industrial automation", Proceedings of CIMCON '90, pp114-32, 1990
- [Boyle and Thomas 1988] Boyle, R. D. & Thomas, R. C. "Computer Vision: A First Course", Blackwell Scientific, pp 55 -87, 1988
- [Boylin 1990], Boylin III, R. E. "CAM-I CIM reference model", Proceedings of CIMCON '90, pp 35-41, 1990
- [Braggins 1989] Braggins, D. "New sensors for vision systems", Automation (UK), pp 29-31, Mar./Apr. 1989

- [Braggins 1990a] Braggins, D. "A vision of the future", *Sensor Review*, pp 121-2, July 1990
- [Braggins 1990b] Braggins, D. "A clear view of image processing", *Image Processing: Yearbook and Exhibition Guide '90*, pp 58-61, Oct. 1990
- [Braggins and Hollingum 1986] Braggins, D. & Hollingum, J. "The machine vision sourcebook", IFS Publications Ltd. 1986
- [Brandli and Mittelstaedt 1989] Brandli, N. & Mittelstaedt, M. "Exchange of solid models: Current state and future trends", *Computer-Aided Design*, Vol. 21, No. 2, pp 87-96, Mar. 1989
- [Breeze 1990] Breeze, P. "Everything you always wanted to know about MAP", *Automation*, pp 26-7, May 1990
- [Bretschi 1981] Bretschi, J. "Automated inspection systems for industry: Scope for intelligent measuring", IFS publication, 1981
- [Buckley 1989] Buckley, D. "SMT advances, *Electronic Production Supplement*", pp31-3, Oct. 1989
- [Buckley 1990a] Buckley, D. "SMT Primer Part 2: Surface mount assembly", *Electronic Production*, Vol.19, No. 8, pp 37-40, Aug. 1990
- [Buckley 1990b] Buckley, D. "SMT Primer Part 4: Cleaning, inspection", rework and testing. *Electronic Production*, Vol.19, No. 10, pp 33-34, 37, Oct. 1990
- [Burson 1987] Burson, D.C. "PC-Board Applications of EDIF", in *Using EDIF 200 for PCB Data*, Workshop, Racal-Redac Ltd. Tewkesbury, Glos. UK 1987
- [Busby et al 1990] Busby, J. S., Hey, D. G. & Hutchison, D. "Linking computer systems in a manufacturing company", *Int. J. of CIM*, Vol. 3, No. 2, pp73-83, 1990
- [CAMI 1983] CAM-I Standards Committee, "A discrete parts manufacturing model (DPMM)" R-83-SC-01, Draft issue 1.0, August 1983
- [Capson and Eng 1988] Capson, D. W. & Eng, S. K. "A tiered-color illumination approach for machine inspection of solder joints", *IEEE Trans. Pattern Anal. Mach. Intelligence*, Vol. PAMI-10, No.3 pp 387-393, 1988
- [Castan 1977] Castan, S. "Image enhancement and Restoration", in Simon, J.C. & Rosenfeld, A. (eds.), *Digital Image Processing and Analysis*, pp 47-62, Noordhoff International Publishing, 1977
- [Challis 1989] Challis, H. "Production control -- today's issues", *Automation (UK)*, pp 33-5, Mar./Apr. 1989

- [Charif 1986] Charif, S. "For better quality, get control of data", *Production Engineering*, Vol. 33, No. 6, pp 66-7, June 1986
- [Chen 1976] Chen, P.P.S. "The entity-relationship model --Toward a unified view of data", *ACM TODS*, Vol.1, No. 1, pp 9-36, 1976
- [Chen 1990] Chen, S. "Automatic solder inspection and process control based on established visual criteria", *Printed Circuit Assembly*, pp 36-42, Mar. 1990
- [Chin and Harlow 1982] Chin, R.T. & Harlow, C.A. "Automated visual inspection: A survey", *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. PAMI-4, No. 6, pp 557-73, Nov. 1982
- [Chin 1988] Chin, R.T. "Automated visual inspection: 1981 to 1987", *Computer Vision, Graphics, and Image Processing* 41, pp 346-81, 1988
- [Chow et al 1985] Chow, W. M., MacNair, E.A. & Sauer, C. H. "Analysis of Manufacturing Systems by the Research Queueing Package", *IBM Journal of Research and Development*, Vol. 29, No. 4, pp 330-42, July 1985
- [Clocksin et al 1983] Clocksin, W.S., Davey, P.G., Morgan, C.G. & Vidler, A.R. "Progress in visual feed-back for arc-welding of thin sheet steel", in Pugh, A. *Robot Vision*, pp 187-98, IFS (Pub.) Ltd. 1983
- [Corey 1990] Corey, R.L. "SMT solder process control", *Circuits Manufacturing*, Vol. 30, No. 7, pp 19-20, July 1990
- [Corr and Neal 1979] Corr, F. P. & Neal, D. H. "SNA and emerging international standards", *IBM Systems Journal*, Vol. 18, No. 2, pp 244-62, 1979
- [Cowie 1990] Cowie, D. "Monitoring process capability", *Electronic Production*, Vol. 19, No.5, page 33, May 1990
- [Cox 1988] Cox, L. "Wave soldering: A study in process control", in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS Publications, pp 245-21, 1988
- [Craig 1986] Craig, J. J. "Introduction to robotics: Mechanics and control", Addison-Wisley 1986
- [Crowder 1985] Crowder R. "The MAP specification", *Control Engineering*, pp 22-5, Oct. 1985
- [Cruz and Damens 1985] Cruz, F. D. & Damens, J. "MS-DOS KERMIT Manual", Version 2.28, Colubia University, CA, USA, June 1985
- [Danielsson and Kruse 1979] Danielson, P. E. & Kruse, B "Distance checking algorithms", *Computer Graphics & Image Processing*, Vol. 11, pp 349-376, 1979

- [Darwish and Jain 1988] Darwish, A. M. & Jain, A. K. "A rule based approach for visual pattern inspection", IEEE Trans. Pattern Anal. Mach. Intelligence, Vol. PAMI-10, No. 1 pp 56-68, 1988
- [Date 1986] Date, C.J. "An introduction to database system", Addison-Wesley publication, Vol. 1 & 2, 4th Edition, 1986
- [Davies 1984] Davies, E. R. "Design of cost-effective systems for the inspection of certain food products during manufacture", in Pugh, A (ed.), "Proc. of 4th conf. on Robot Vision and Seneory Controls", pp 437-446, 1984
- [Davies 1986] Davies, E. R. "Image space transforms for detecting straight edges in industrial images", Pattern Recognition Letters (North-Holland) Vol. 4, pp 185-92, 1986
- [Davies 1987] Davies, E. R. "A high speed algorithm for circular object detection", Pattern Recognition Letters (North-Holland) Vol. 6, pp 323-33, 1987
- [Davies 1989a] Davies, E. R. "Finding ellipses using the generalised Hough transform", Pattern Recognition Letters (North-Holland) Vol. 9, pp 87-96, 1989.
- [Davies 1989b] Davies, E. R. "Minimising the search space for polygon detection using the generalised Hough transform", Pattern Recognition Letters (North-Holland) Vol. 9, pp 181-92, 1989
- [Davies 1989c] Davies, E. R. "Edge location shifts produced by median filters: Theoretical bounds and experimenatal results", Signal Processing, Vol. 16, pp 83-96, 1989
- [Davis et al 1975] Davis, L. S., Rosenfeld, A. & Weszka, J. S. "Region extraction by averaging and thresholding", IEEE Trans. on Syst. Man and Cybern. pp 383-7, May 1975
- [Davy 1988] Davy, J. G. "Wave solder defects", in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 230-6, 1988
- [Day and Zimmermann 1983] Day, J. D. and Zimmermann, H., "The OSI reference model", Proc. of the IEEE, 71 (12), 1983
- [Dolberg and Kovarsky 1989] Dolberg, S. "The search for the true golden panel", PC Fab., pp 48-59, Dec. 1989
- [Dooner 1989] Dooner, M. "Models of production systems", Computer-Aided Design, Vol. 21, No. 3, pp 182-3, April 1989
- [Doyle 1990] Doyle, K. G. "Automatic optical inspection-The way forward", Proceedings of Printed Circuit World Convention, pp (B10/1-1)-(B10/1-7),

June 1990

- [Dresser 1990] Dresser, D. "IR imaging for PCB testing", Advanced Imaging, pp 46, 48 & 69, April 1990
- [Driels and Lee 1988] Driels, M. R. and Lee, C. C. "Feature selection for automatic visual inspection of solder joint", The Int. J. of Advanced Manufacturing Technology, 3 (4) pp 3-32, 1988
- [Duda and Hart 1972] Duda, R. O. and Hart, P. E. "Use of the Hough transform to detect lines and curves in pictures", Comms of the ACM 15, pp 11-5, 1972
- [Duda and Hart 1973] Duda, R. O. and Hart, P. E. "Pattern classification and scene analysis", Wiley, NY, 1973
- [Du Feu 1988] Du Feu, P. H. "A guide to advanced manufacturing in electronics", Peter Peregrinus, Ltd., 1988
- [Dunbar 1986] Dunbar, P. "Machine vision", Byte, pp 161-73, January 1986
- [EDIF-PCB-TSC 1990] EDIF PCB Technical Sub-Committee, "Conceptual model of a PCB", Version 9, Part 1 & Part 2, 1990
- [Edwards 1990] Edwards, J.M. "Machine vision and its integration within CIM systems in the electronics manufacturing industry", Computer-Aided Engineering Journal, pp 12-8, Feb. 1990
- [Eckes 1990] Eckes, G. "Obtaining management involvement in the PWB quality improvement process", Proceedings of Printed Circuit World Convention 5, pp (C11/3-1) -(C11/3-9), 12-15 June, UK 1990
- [Ejiri 1973] Ejiri, M., Uno, T., Mese, M. & Ikeda, S. "A process for detecting defects in complicated patterns", Comput. Graphics Image Processing, Vol. 2, pp 326-339, 1973
- [Eldan 1990] Eldan, E. "The current state and future of AOI", Circuits Manufacturing, pp 49-55, June 1990
- [EP Report 1989] EP Editorial, "PCB verification", Electronic Production, Vol. 18, No. 7, pp13-6, July 1989
- [Express 1989] EXPRESS Language Reference Manual, ISO TC184/SC4/WG1, Doc. No. N442, December 1989, obtainable from CADETC, Leeds Univ., UK.
- [Esposito 1989] Esposito, D.J. "AOI vendor update", PC Fab., pp 33-4, 39-47 & 109, Dec. 1989

- [Eurich and Roth 1990] Eurich, J. P. & Roth, G. "EDIF grows up", IEEE Spectrum, Vol. 27, No. 11, pp 68-72, November 1990
- [Fairhurst 1988] Fairhurst, M.C. "Computer Vision for Robotic Systems", An Introduction, pp 55-85, Prentice Hall 1988
- [Farowich 1986] Farowich, S. "Communicating in the technical office", IEEE Spectrum, Vol. 23, No. 4, pp 63-7, April 1986
- [Farrimond 1988] Farrimond, B.T. "Specification of a support system for use in the manufacture of printed circuit boards using a neutral format", M.Sc Thesis, Department of Computer Science, University of Manchester, 1988
- [Foong and Hoang 1991] Foong, T. & Hoang, K. "Quantifying CIM", Proceedings of Int. Conf. on Computer Integrated Manufacturing, ICCIM '91, pp 23-9, October 1991
- [Foster et al 1990] Foster III, J.W., Griffin, P.M., Messimer, S.T. & Villalobos, J.R. "Automated visual inspection: A tutorial", Computers Ind. Eng., Vol. 18, No.4, pp 493-504, 1990
- [Frenkel 1990] Frenkel, K.A. "The politics of standards and the EC", Comm of The ACM, Vol. 33, No. 7, pp 41-51, July 1990
- [Franklin et al 1986] Franklin, G. F., Powell, J. D. & Emami-Naeini, A. "Feedback Control of Dynamic Systems", Addison-Weseley, 1986
- [Fu 1980] Fu, K.S. "Recent developments in pattern recognition", IEEE Trans. on Computers, Vol. C-29, No. 10, pp 845-54, 1980
- [Fu 1982] Fu, K.S. "A general (syntactic- semantic) approach to picture analysis", in Fu, K.S. & Kunii, T.L. (eds): Picture Engineering, pp 56-74, Springer-Verlag 1982
- [Fu et al 1987] Fu, K. S., Gonzalez, R. C. & Lee, C. S. G. "Robotics: Control, sensing, vision, and intelligence", McGraw-Hill Book Company, 1987
- [Fu and Mui 1981] Fu, K.S. & Mui, J. "A survey on image segmentation", Pattern recognition, Vol. 13, pp 3-16, 1981
- [Fu and Rosenfeld 1984] Fu, K. S. & Rosenfeld, A. "Pattern recognition and computer vision", Computer, pp 274-82, October 1984
- [Galbiati Jr. 1990] Galbiati Jr., L. J. "Machine Vision and Digital Image Processing Fundamentals", Prentice-Hall International Inc., 1990
- [Garakani and Cobb 1986] Garakani, A., & Cobb, J. "Machine vision meets the challenge for small measures", Robotics world, pp 60-8, April 1986

- [Gascoigne et al 1987] Gascoigne, J.D. Weston, R.H. & Sumpter, C.M., "Robot protocols and interfacing to LANs", *Int. J. of Production Research*, Vol. 25, No. 1, pp115-28, 1986
- [Geng et al 1991] Geng, Z., Fiala, J., Haynes, L., Bukowski, R., Santucci, A. & Coleman, N. "Robotic hand-eye coordination", pp (4-13)-(4-24), *Fourth World Conference On Robotics Research*, 1991
- [Gill 1990] Gill, K. "OCR aids robotic assembly", *Sensor Review*, Vol. 10, No. 4, pp 170-3, 1990
- [Gilutz 1988] Gilutz, H. "Automatic optical inspection systems", in Riely, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 346-352, 1988
- [Gilutz 1990] Gilutz, H. "AOI applications for the future: PCB and Beyond", *PC Fab*, pp 89, 93, Sept 1990
- [Gonzalez and Wintz 1987] Gonzalez, R. C. & Wintz, P. "Digital image processing", 2nd ed., pp 331-90, Addison-Wesley, 1987
- [Graube and Mulder 1984] Graube, M. & Mulder, M. C. "Local area networks", *Computer*, pp 242-47, October 1984
- [Graves et al 1988] Graves, G. R., Yelamanchili, B. & Parks, C. M. "An interface architecture for CAD/CAPP integration using knowledge-based systems and feature recognition algorithms", *Int. J. of CIM*, Vol. 1, No. 2, pp 89-100, 1988
- [Gray 1991] Gray, P. A. "Open System: A Business Strategy for the 1990s", McGraw-Hill Book Company, 1991
- [Gray and McNeill 1979] Gray, J. P. & McNeill, T. B. "SNA multiple-system networking", *IBM Systems Journal*, Vol. 18, No. 2, pp 263-97, 1979
- [Groover and Zimmers Jr. 1984] Groover, M. P. & Zimmers Jr., W. "CAD/CAM: Computer-aided design and manufacturing", Prentice-Hall, Inc., 1984
- [Groover et al 1986a] Groover, M. P., et al, "Industrial Robotics Technology, Programming, and Applications" (Chapter 7: Machine Vision), McGraw-Hill International Editions, pp 160-186, 1986,
- [Groover et al 1986b] Groover, M. P., et al, "Industrial Robotics Technology, Programming, and Applications" (Chapter 15: Assembly and Inspection), McGraw-Hill International Editions, pp 416-54, 1986
- [Groover 1987] Groover, M. P. "Automation, Production Systems, and Computer Integrated Manufacturing", Prentice-Hall International Editions, pp707-42, 1987

- [Gruver et al 1984] Gruver, W. A., Soroka, B. I., Craig, J. J. & Turner, T. L. "Industrial robot languages: a comparative evaluation", IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-14, No. 4, pp 565-570, 1984
- [Gurian 1990] Gurian, M.I., "Fine-line processing: The '90s are here!", PC Fab, pp 64-71, May 1990
- [Hansohn 1990] Hansohn, S. "Standards in data communication", PC Fab, pp 38-44, July 1990
- [Hara et al 1983] Hara, Y., Akiyama, N. & Karasaki, K. "Automatic inspection system for printed circuit boards", IEEE Trans. on Pattern Anal. and Mach. Intelligence, Vol. PAMI-5, No. 6, pp 623-30, 1983
- [Hara et al 1988] Hara, Y., Doi, H. Karasaki, K. & Iida, T. "A system for PCB automated inspection using fluorescent light", IEEE Trans. Pattern Anal. Mach. Intelligence, Vol. PAMI-10, No. 1 pp 69-78, 1988
- [Haralick 1983] Haralick, R. M. "Image segemntation survey", in Faugeras, O. D. (ed.) "Fundamentals in computer vision: An advanced course", pp 209-23, Cambridge University Press, 1983
- [Haren and Williams 1990] Haren, C. V. and Williams, T. "A reference model for computer integrated manufacturing from the view point of industrial automation", Proceedings of CIMCON '90, pp 42-62, 1990
- [Harhalakis et al 1991] Harhalakis, G., Lin, C. P., Mark, L. & Muro-Medrand, P. P. "Information systems for integrated manufacturing (INSIM): A design methodology", Int. J. of CIM, Nol. 4, No. 6, pp 351-363, 1991
- [Heginbotham et al 1983] Heginbotham, W.B., Barnes, D. F., Purdue, D. R. & Law, D. J. "Flexible assembly module with vision control robot", in Pugh, A. (ed.) "Robot Vision", IFS publications, pp 245-51, 1983
- [Hemond 1986] Hemond, R. F. "Making CIM working on the shop-floor", Production Engineering, Vol. 33, No. 9, pp 44-7, September 1986
- [Hidde 1991] Hidde, A. R. "Management of information in a system of heterogeneous distributed data base using the example of PCB assemblage", Int. J. of CIM, Vol. 4, No. 6, pp 323-30, Nov./Dec. 1991
- [Hillawi and Bennett 1986] Hillawi, J.I. & Bennett, K.R., "EDIF -- An overview", Computer-Aided Engineering Journal, pp 102-107, June 1986
- [Hitomi 1990] Hitomi, K. "Manufacturing systems engineering: the concept, its context and the state of the art", Int. J. of CIM, Vol. 3, No. 5, pp 275-88, 1990

- [Holland 1983] Holland, R. C. "Microcomputers For Process Control", Pergamon Press, 1983
- [Hollington 1984] Hollington, J. "Machine vision: The eye of automation", IFS (Pub.) Ltd., 1984
- [Horn 1975] Horn, B.K.P. "Obtaining shape from shading information", in Winston, P.H. (ed.) The Psychology of Computer Vision, McGraw-Hill, pp 115-55, 1975
- [Hospod 1990] Hospod, T.F., "PC--based pattern recognition for industrial vision", Advanced Imaging, pp 43-4 & 46, MAY 1990
- [Howard 1990] Howard, G. "Imaging and the PS/2", Advanced Imaging, pp 42-8, Jan. 1990
- [ICL-Kidsgrove-1] "ICL Manufacturing Operations: Printed Circuit Board Manufacture", Internal Document of ICL Kidsgrove, Stoke-on-Trent, England
- [ICL-Kidsgrove-2] "ICL Inspection Criteria: for solder joint inspection", Internal Document of ICL Kidsgrove, Stoke-on-Trent, England
- [Ikeuchi and Horn 1981] Ikeuchi, K. & Horn, B.K.P. "Numerical shape from shading and occluding boundaries", Artificial Intelligence, 17, pp 141-84, 1981
- [ISO 1986] International Standard Organization, "The Ottawa report on reference models for manufacturing standards", ISO TC184/SC5/WG1 N51, Version 1.1, 1986
- [Iverson 1988] Iverson, R. "Mathematical Morphology: The study of shape and position", PC Fab., pp 22-6 & 32, Nov. 1988
- [Jarema and Sussengüth 1981] Jarema, D. R. & Sussengüth, E. H. "IBM data communications: A quarter century of evolution and progress", IBM J. of Res. and Dev., Vol. 25, No. 5, pp 391-404, Sept. 1981
- [Jarvis 1980] Jarvis, J. F. "A method for automating the visual inspection of printed wiring-boards", IEEE Trans. on Pattern Anal. and Mach. Intelligence, Vol. PAMI-2, No. 1, pp 77-82, 1988
- [Jarvis 1983] Jarvis, R.A., "A perspective on range finding techniques for computer vision", IEEE Trans. PAMI, Vol. PAMI-5, No.2, pp 122-39, Mar. 1983
- [Jasany 1986] Jasany, L. C. "MAP and proprietary networks: Building a bridge to automation", Production Engineering, Vol. 33, No. 10, pp 60-62, 65, October 1986

- [Johnston 1983] Johnston, E. "A CCTV camera controlled painter", in Pugh, A. (ed.) "Robot Vision", IFS publications, pp 277-84, 1983
- [Jorysz and Vernadat 1990a] Jorysz, H. R. & Vernadat, F. B. " CIM-OSA part 1: Total enterprise modelling and function view", Int. J. of CIM, Vol. 3, No. 3 and 4, pp 144-56, 1990
- [Jorysz and Vernadat 1990b] Jorysz, H. R. & Vernadat, F. B. " CIM-OSA part 2: Information view", Int. J. of CIM, Vol. 3, Nos. 3 and 4, pp 157-67, 1990
- [Kaminski 1986] Kaminski Jr., M. A. "Protocols for communicating in the factory", IEEE Spectrum, Vol. 23, No. 4, pp 56-62, April 1986
- [Kaplan 1990] Kaplan, G. "Industrial electronics", IEEE Spectrum, Vol. 27, No. 2, pp 42-3, February 1990
- [Kay and Luo 1991] Kay, M. G. & Luo, R. C. "Intelligent control for multiple AGVs using global vision", Proceedings of 4th World Conference on Robotics Research, pp (2-53) - (2-64), 1991
- [Keeler 1988] Keeler, R. "Machine vision", in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 352-60, 1988
- [Keller 1988] Keller, J. "Solder joint acceptability", in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 227-30, 1988
- [Kent 1978] Kent, W. "Data and reality", North-holland 1978
- [Kent and Shneier 1986] Kent, E. W. & Shneier, M. O. "Eyes for automations", IEEE Spectrum, Vol. 23, No. 3, pp37-45, March 1986
- [Kiko 1984] Kiko, T. L. "Printed circuit board basics: A general reference to the printed circuit industry", PMS Industries (Pub.), 1984
- [King 1990] King, J. "ATE vs AOT", Electronic Production, Vol. 19, No. 4, pp 25, Apr. 1990
- [Kitchin and Pugh 1983] Kitchin, P. W. & Pugh, A. "Processing of Binary images", in Pugh, A. (ed.) "Robot Vision", IFS publications, pp 21-42, 1983
- [Klittich 1990] Klittich, M. " CIM-OSA part 3: CIM-OSA integrating infrastructure - the operational basis for integrated manufacturing systems", Int. J. of CIM, Vol. 3, Nos. 3 and 4, pp 168-80, 1990
- [Kohno et al 1983] Kohno, M. "Intelligent assembly robot", in Pugh, A. (ed.) "Robot Vision", IFS publications, pp 201-8, 1983
- [Koren 1983a] Koren, Y. "Computer Control of Manufacturing Systems", pp 169-

91, McGraw-Hill Book Company, 1983

- [Kosanke 1991] Kosanke, K. "Open system architecture for CIM (CIM-OSA) standards for manufacturing", Int. conf. on Computer integrated Manufacturing, ICCIM '91, pp77-80, October 1991, Singapore
- [Krotkov and Martin 1986] Krotkov, E. & Martin, J.P. "Range from focus", Proc. IEEE Robotics and Automation, pp 1093-8, 1988
- [Kutcher 1983] Kutcher, M. "Automating it all", IEEE Spectrum, Vol. 20, No. 4, pp 40-3, April 1983
- [Kutcher and Gorin 1983] Kutcher, M. & Gorin, E. "Moving data, not paper, enhances productivity", IEEE Spectrum, Vol. 20, No. 4, pp 84-8, April 1983
- [Landman 1988] Landman, K. "Overview of AOI system", PC FAB, pp 17-34, Nov. 1988
- [Leech et al 1991] Leech, M. J., Weston, R. H., Hodgson, A., Clements, P. & Ryan, A. "Incremental building of CIM systems", Proc. of Int. Conf. on Computer Integrated Manufacturing, ICCIM '91, pp 125-8, October 1991, Singapore
- [Llewelyn 1989] Llewelyn, A.I. "Review of CAD/CAM", Computer-Aided Design, Vol. 21, No. 5, pp 297-302, June 1989
- [Lloyd 1990] Lloyd, R. "The analytical technique", PC FAB, pp 83-6, July 1990
- [Levialdi 1983] Levialdi, S. "Edge extraction techniques", in Faugeras, O. D. (ed.) "Fundamentals in computer vision: An advanced course", pp 117-44, Cambridge University Press, 1983
- [Libes and Barkmeyer 1988] Libes, D. & Barkmeyer, E. "The integrated manufacturing data administration system (IMDAS): An overview", Int. J. of CIM, Vol. 1, pp 44-9, 1988
- [Low 1991] Low, A. "Introductory Computer Vision and Image Processing", McGraw-Hill Book Comp., 1991
- [Low and Chee 1991] Low, S. K. L. & Chee, S. M. K. "Expanding the integrated system network through EDI", Proc. of Int. Conf. on Computer Integrated Manufacturing, ICCIM '91, pp 118-21, October 1991, Singapore
- [Lozano 1990] Lozano, R. "Automatic optical inspection: the system of tomorrow", Evaluation Engineering, Vol. 29, No. 10, pp 40-2, Oct. 1990
- [Macri and Calengor 1980] Macri, G.C. & Calengor, C.S. "Robots combine speed and accuracy in dimensional checks of automotive bodies", Robotics Today,

pp 16-9, Summer 1980

- [Mandeville 1985] Mandeville, J. R. "Novel method for analysis of printed circuit images", IBM J. Res. Develop. Vol. 29, No. 1, pp 73-86, 1985
- [Mangin and McClelland 1987] Mangin, C.H., & McClelland, S. "Surface Mount Technology", IFS (Pub.) Ltd., UK & Springer-Verlag, 1987
- [Mangin 1988] Mangin, C. H. "Assembly challenges", in Riely, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp85-6, 1988
- [Maria 1986] Maira, A. "Local area networks -- the future of the factory", Manuf. Eng., pp 77-9, Mar. 1986
- [Markstein 1988a] Markstein, H. W. "Modern platers and etcheres", in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp15-20, 1988
- [Markstein 1988b] Markstein, H. W. "Automatic component insertion/placement systems", in Riely, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp70-6, 1988
- [Martelli 1976] Martelli, A. "An applicaion of heuristic search methods to edge and contour detection", Comms. of the ACM, Vol. 19, pp 73-83, 1976
- [Masaki et al 1983] Masaki, I. Dunne, M. J. and Toda, H. "Vision guided robot system for arc welding", in Pugh, A. (ed.) Robot Vision, pp 179-86, IFS (Pub.) Ltd. 1983
- [Matrox 1988] "IMAGER-AT Reference Manual", No. 272-MF-00, Revision 5, and "IMAGER-AT User's Manual", No. 272-MU-00, Revision 6, Dec. 1988 by Matrox Electronic Systems Ltd., 1055 St Régis BLVD, Dorval, Québec, Canada, H9p 2T4,
- [Mawby 1989] Mawby, T. "A primer on probes", Electronic Production, Vol. 18, No. 9, pp 38-43, Sept. 1989
- [Mertins and Sussenguth 1991] Mertins, K. & Sussenguth, W. "Integrated information modelling for CIM", Computer-Integrated Manufacturing Systems, Vol. 4, No. 3, August 1991
- [Mills et al 1991] Mills, J., Goldstein, D. & Lindahl, C. "A flexible, modular, computer architecture for CIM", Proceedings of Int. Conf. on Computer Integrated Manufacturing, ICCIM '91, pp 81-4, October 1991, Singapore
- [Montanari 1971] Montanari, U. "On the optimal detection of curves in noisy pictures", Comm. ACM, Vol. 14, pp 335-45, 1971

- [Moon 1985] Moon, D. "Developing standards smooth the integration of programmable factory floor devices", *Control Engineering*, pp 49-53, Feb 1985
- [Mosca 1990] Mosca, V. G. "Eliminating the need for 100% solder joint inspection", 4th International SAMPE Electronics Conference, Vol. 4, pp 616-27, 12-14 June 1990
- [Mueller and Verrecchia 1990] Mueller, D. & Verrecchia, S. "Delivering a new generation of image processing boards", *Advanced Imaging*, pp 46-7, Jan. 1990
- [Munro and Noori 1988] Munro, H. & Noori, H. "Measuring commitment to new manufacturing technology: Integrating technological push and marketing pull concepts", *IEEE Trans. on Engineering Management*, Vol. 35, No. 2, pp 63-70, May 1988
- [Murray 1988] Murray, J. "CIM -- Plan top down, implement bottom up", *Control Engineering (Special report: Factory automation)*, Vol. 34, pp 62-5, April 1988
- [Murphy 1990] Murphy, E. E. "Reconciling conflicting design-automation standards", *IEEE Spectrum*, Vol. 27, No. 3, pp 44-8, March 1990
- [Myers 1988] Myers, R. L. "CAD/CAM for PCB manufacturing" in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 500-4, 1988
- [Nakagawa 1982] Nakagawa, "Automatic visual inspection of solder joint on printed circuit boards", *SPIE Vol. 336*, pp 121-7, 1982
- [Niemann 1979] Niemann, H. "Digital image processing", in Stucki, P. (ed.) "Advances in Digital Image Processing: Theory, Application, Implementation", pp 77-122, Plenum Press, N.Y., 1979
- [Nitzan et al 1987] Nitzan, D., Bolles, R., Kremers, J., & Mulgaonkar, P. "3-D vision for robot applications", in Wong, A. K. C. and Pugh, A. (eds.) *Machine Intelligence and Knowledge Engineering for Robotic Applications*, Springer-Verlag, pp 21-81, 1987
- [Offen 1985] Offen, R. J. (ed.) "VLSI Image Processing", Collins, 1985
- [Owen and Bloor 1987] Owen, J., & Bloor, M.S. "Neutral formats for product data exchange: The current situation", *Computer-Aided Design*, Vol. 19, No. 8, pp 436-43, Oct. 1987
- [Panse 1990] Panse, R. "CIM-OSA: A vendor independent CIM architecture", *Proceedings of CIMCON '90*, pp 177-96, 1990

- [Pao 1984] Pao, Y.C. "Elements of Computer-Aided Design and Manufacturing", John Wiley & Sons, 1984
- [Parunak and White 1987] Parunak, H.V.D. & White, J.F. "A synthesis of factory reference models", ITI TR-87-29 (International Technology Institute, PO Box 1485, Ann Arbor, Michigan), 1987
- [Pavlidis 1982] Pavlidis, T. "Algorithms for graphics and image processing", pp65-73, Computer Science Press, 1982
- [PCAD 1989a] "System utilities--user's guide", by Personal CAD Systems, Inc., 1989
- [PCAD 1989b] "Printed circuit board design--PCAD illustrated user's guide", Personal CAD Systems, Inc., 1989
- [PCAD 1989c] "PDIF User's Manual", Personal CAD Systems, Inc., 1989
- [PE Staff Report 1986] Production Engineering Staff Report "Integrated Manufacturing", Production Engineering, Vol. 33, No. 2, 1983
- [Pessen and Hubl] Pessen, D. & Hubl, W. "The Design and Application of Programmable Sequence Controller for Automation Systems", Lonman Group Limited, 1979
- [Pfeiffer 1990] Pfeiffer, D. "The case for the embedded imaging approach", Advanced Imaging, pp 36-40, Oct. 1990
- [Pound 1988a] Pound, R. "Inspection equipment", in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 393-5, 1988
- [Pound 1988b] Pound, R., "Image processing and non-destructive testing", in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 395-401, 1988
- [Powers Jr. 1987] Powers Jr., J.H. "Computer-Automated Manufacturing", McGraw-Hill Book Co. 1987
- [Powell and Carignan 1989] Powell, J. & Carignan, B. "CAD reference for AOI", PC FAB, pp. 77-93, Dec 1989
- [Prasad 1989] Prasad, R. P. "Surface mount technology: Principles and practice", Van Nostrand Reinhold, 1989
- [Pratt 1978] Pratt, W.K. "Digital Image Processing", John Wiley & Sons, Inc. 1978
- [Prince 1989] Prince, A. "ATE -- The pre-wired probe approach", Electronic Production, Vol. 18, No.4, pp 19-26, Apr. 1989

- [Pugh 1983] Pugh, A. "Second generation robotics", in Pugh, A. (ed.) "Robot Vision", pp 3-10, IFS Publications Ltd. 1983
- [Purll 1985], Purll, D. J. "Solid-state image sensors", in Batchelor, B. G. et al (eds) "Automated Visual Inspection", IFS publications, pp 255-93, 1985
- [Racal-Redac 1987] Racal-Redac Workshop: "Using EDIF 200 for PCB Data", Racal-Redax Ltd. Tewkesbury, Glos. U.K. 1987
- [Rajagopalan and Cheng 1991] Rajagopalan, R. & Cheng, R. M. H., "Binary camera vision for AGV navigation", Proceedings of 4th World Conference On Robotics Research, pp (2-23)-(2-39), 1991
- [Ranky 1991] Ranky, P. G. "Total quality information system design model within a CIM architecture", Proc. of Int. Conf. on Computer Integrated Manufacturing, ICCIM '91, pp71-4, October 1991, Singapore
- [Rieley 1990] Rieley, R.v. "Inspect for success", Circuits Manufacturing, pp 84-8, Mar. 1990
- [Riley 1988] --- "Electronics CAE/CAD/CAM", in Riely, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 491-500, 1988
- [Rittichier 1989] Rittichier, J. "The changing rules of AOI", Electronic Production, Vol. 18, No. 10, pp 28-30, Oct. 1989
- [Rosenfeld 1969] Rosenfeld, A. "Picture processing by computer", Academic Press, 1969
- [Rosenfeld and Kak 1976] Rosenfeld, A. & Kak, A.C. "Digital Picture Processing", Academic Press, N.Y. 1976
- [Rosenfeld 1983] Rosenfeld, A. "Segmentation: Pixel-based methods", in Faugeras, O. D. (ed.) "Fundamentals in computer vision: An advanced course", pp 225-37, Cambridge University Press, 1983
- [Rosenfeld 1987] Rosenfeld, A. "Robot vision", in Wong, A. K. C. and Pugh, A. (eds.) Machine Intelligence and Knowledge Engineering for Robotic Applications", Springer-Verlag, pp 1-19, 1987
- [Rossol 1983] Rossol, L. "Computer Vision in industry-The next decade", in Pugh, A. (ed.) Robot Vision, pp 11-18, IFS (Pubs.), Ltd. 1983
- [Ruocco 1987] Ruocco, S. R. "Robot sensors and transducers", Open University Press, 1987
- [Rummel 1989] Rummel, P. "Applied robot vision: combining workpiece recognition and inspection", in Freeman, H. (ed.) Machine Vision for

- Inspection and Measurement", pp 203-221, Academic Press 1989
- [Samet 1980] Samet, H. "Region representation: Quadtrees from boundary codes", Comm. of ACM, Vol. 23, No. 3, pp 163-70, March 1980
- [Sanz 1988] Sanz, G. L. C. "Introduction to the special PAMI issues on industrial machine vision and computer vision technology", IEEE Trans. Pattern Anal. Mach. Intelligence, Vol. PAMI-10, No.1 pp 1-3, 1988
- [Schalkoff 1989] Schalkoff, R. J. "Digital image processing and computer vision", John Willey & Sons, Inc. 1989
- [Scheer and Hars 1991] Scheer, A. W. & Hars, A. "From CIM to enterprise-wide data modelling", Proc. of Int. Conf. on Computer Integrated Manufacturing, ICCIM '91, pp 89-92, October 1991, Singapore
- [Seames 1990] Seames, W. S. "Computer numerical control: Concepts and programming", 2nd edition, Delmer Publishers, Inc., 1990
- [Serra 1982] Serra, J. "Image analysis and mathematical morphorlogy", Academic Press, 1982
- [Shorter 1990] Shorter, D. "Progress towards standards for CIM architectural frameworks", Proceedings of CIMCON '90, pp 216-31, 1990
- [Silven et al 1984] Silven, O., Piironen, T., Elsilä, M. & Pietikäinen, M. "Performance evaluation of algorithms for visual inspection of printed circuit boards", 7th IEEE Int. Conf. on Pattern REcognition, pp1355-7, 1984
- [Solberg and Heim 1989] Solberg, J. J. & Heim, J. A. "Managing information complexity in material flow systems", in Nof, S. Y. & Moodie, C. L. (ed.) "Advanced Information Technologies for Industrial Material Flow Systems", pp 3-20, Springer-Verlag, 1989
- [Spitz 1988] Spitz, S. L. "Computerized soldering systems", in Riley, F. (ed.) "The Electronics Assembly Handbook", IFS publications, pp 259-63, 1988
- [Sprague et al 1991] Sprague, A. P., Donahue, M.J. & Rokhlin, S. I. "A method for automatic inspection of printed circuit boards", CVGIP: Image Understanding, Vol. 54, No. 3, pp 401-15, November 1991
- [Stanton 1989] Stanton, C. "SPC--an attitude of mind", SPC, pp 37-8, Mar./Apr. 1989
- [Stix 1990] Stix, G. "Data communications", IEEE Spectrum, Vol. 27, No. 1, pp 35-7, January 1990

- [Stoll 1990] Stoll, J. "CAM systems", PC Fab, pp 93-6, July 1990
- [Sumpter et al 1987] Sumpter, C., Weston, R. H. & Gascoigne, J. D. "Computer integration in flexible assembly systems", Int. J. of Robotics Research, Vol. 10, pp43-9, 1987
- [Sun 1986a] Sun Microsystems, Inc. "Chapter 7: Lex -- A lexical analyzer generator", in Programming Utilities for the Sun Workstations, pp 119-40, Revision F of 15 February 1986
- [Sun 1986b] Sun Microsystems, Inc. "Chapter 8: Yacc -- Yet another compiler-compiler", in Programming Utilities for the Sun Workstations, pp 143-82, Revision F of 15 February 1986
- [Svetkoff et al 1987] Svetkoff, D.J., Smith, D. N. & Doss, B. L. "Automatic inspection of component boards using 3-D & greyscale vision", Hybrid Circuit, No. 13, pp 5-9, May 1987
- [Takagi et al 1991] Takagi, Y., Hata, S., Hibi, S. and Beutel, W., "Visual inspection of solder joint using tiered illumination", in Batchelor, B.G. & Walts, F. M. (eds) Machine Vision System Integration in Industry, SPIE 1386, pp 21-9, 1991
- [Tanimoto 1980] Tanimoto, S. L. "Image data structures", in Tanimoto, S. L. & Klinger, A. (eds) Structured Computer Vision: Machine Perception through Hierarchical computation Structures, pp 31-55, Academic Press, 1980
- [Tillman and Yen 1990] Tillman, M. A. & Yen, D. C. C. "SNA and OSI: Three strategies for interconnection", Comm. of ACM, Vol. 33, No. 2, pp 214-24, Feb. 1990
- [Vernon 1991] Vernon, D. "Machine vision: Automated visual inspection and robot vision", Prentice-Hall, 1991
- [Voelcker 1986] Voelcker, J. "Helping computers communicate", IEEE Spectrum, Vol. 23, No. 3, pp 61-70, March 1986
- [Welch Jr. 1986] Welch Jr., F. K. "MAP/TOP: Linking factory and office", Production Engineering, Vol. 33, No. 6, pp 36-8, June 1986
- [Walsh 1991] Walsh, P.M., "Rapid system integration with symbolic programming", in Batchelor, B.G. & Waltz, F. M. (eds) Machine Vision System Integration in Industry, SPIE 1386, pp84-9, 1991
- [Wearden 1990] Wearden, T. "Subcontractor computersises manufacturing", Electronic Production, Vol. 19, No. 7, pp 37-40, July 1990
- [West 1984] West, G. A. W. "A system for the automatic inspection of bare-printed

- circuit boards", IEEE Trans. on Systems, Man, and Cybernetics, Vol. SMC-14, No. 5, pp 767-773, 1984
- [Weston et al 1988] Weston, R.H., Gascoigne, J. D., Rui, A., Hodgson, Sumpter, C. M. & Countts, I. "Steps towards information integration in manufacturing", Int. J. Computer Integrated manufacturing, Vol. 1, No. 3, pp 140-53, 1988
- [Weston et al 1989a] Weston, R.H. et al, "Integration tools based on OSI networks", AUTOFACT'89, pp (17-1) -(17-15), 1989
- [Weston et al 1989b] Weston, R.H. Gascoigne, J.D., Sumpter, C.M. & Hodgson, A. "Robot integration within computer-integrated manufacture" Int. J. of Production Research, Vol. 27, No. 3, pp 515-28, Mar 1989
- [Weston et al 1989c] Weston, R.H., Hodgson, A. Gascoigne, J.D., Sumpter, C.M., Rui, A. & Couttes, I. "Configuration methods and tools for manufacturing systems integration", Int. J. of CIM, Vol.2, No. 2, pp 77-85, 1989
- [Weston et al 1989d] Weston, R. H., Gascoigne, J. D, Sumpter, C. M. & Hodgson, A. "The need for a generic framework for systems integration", in Nof, S. Y. & Moodie, C. L. (ed.) "Advanced Information Technologies for Industrial Material Flow Systems", pp 278-306, Springer-Verlag, 1989
- [Weston et al 1990] Weston, R. H., Hodgson, A., Coutts, I., Mugatroid, I. & Gascoigne, J. D. "Highly extendable CIM systems based on an integrated platform", Proceedings of CIMCON '90, pp 80-94, 1990
- [Weston 1991a] Weston, R. H. "21st century CIM enterprises", Proc. of Int. Conf. on Computer Integrated Manufacturing, ICCIM '91, pp 3-6, October 1991, Singapore
- [Weston 1991b] Weston, R. H. "Integration toolboxes -- A tutorial", presented at the International Computer Integrated Manufacturing, ICCIM '91, October 1991, Singapore
- [Weszka 1978] Weszka, J. S., "A survey of threshold selection techniques", Computer Graphics and Image Processing, Vol. 7, pp 259-65, 1978
- [Willis 1989] Willis H. "Toward image understanding, next stage for imaging", Advanced Imaging, pp 42-4, Sept. 1989
- [Witkin 1981] Witkin, A.P. "Recovering surface from shape and orientation", Artificial Intelligence, 17, pp17-45, 1989
- [Wright 1990] Wright, R.S. "The 'Syntegration' of the PCB fab shop", PC Fab., pp 136-140, Feb 1990
- [Ye and Danielsson 1988] Ye, Q. Z. & Danielsson, P. E. "Inspection of printed

- circuit boards by connectivity preserving shrinking", IEEE Trans. Pattern Anal. Mach. Intelligence, Vol. PAMI-10, No.5 pp 439-451, 1988
- [Zuern 1990] Zuern, A. "CIMplementation: AOI-based Process Control", PC Fab., Vol. 13, No. 9, pp 88, 90-1, Sept. 1990
- [Zuech 1988] Zuech, N. "Applying machine vision", pp 149-62, John Willey & Sons, Inc., 1988
- [Zuech and Miller 1989] Zuech, N. & Miller, R. K. "Machine Vision", Van Nostrand Reinhold, 1989

Appendix A

List of Implemented Routines

I The Primitive Routines (PRs)

Control and Configuration

VIDEO	selects video as output to monitor
IMAGE	selects frame buffer image as output to monitor
INITMAT	initializes the Matrox vision processor
WINDOW(x1, y1, x2, y2)	sets processing and I/O window to (x1, y1, x2, y2)
WINFULL	sets processing and I/O window to (0, 0, 512, 480)
WINTL	sets processing and I/O window to (0, 0, 256, 240)
WINTR	sets processing and I/O window to (256, 0, 512, 480)
WINBL	sets processing and I/O window to (0, 240, 256, 480)
WINBR	sets processing and I/O window to (256, 240, 512, 480)
WOWGEN(wtype)	generates window file of the type specified by wtype

Image I/O and Frame Buffer Manipulation

ICOPY(sbuf, dbuf)	copies an image from frame buffer sbuf to dbuf
ILOAD(ImageFile)	loads an image from a disc file to current_buffer_in_display
ISAVE(ImageFile)	stores the image from current_buffer_in_display to a disk file
ISNAP	takes a snapshot to current_buffer_in_display
CHGBUF(buf)	changes to frame buffer buf
CLRBUF(buf)	clears frame buffer buf

Preprocessing

ENHANCE(v1, v2, u1, u2)	maps grey levels in the range (v1,v2) to (u1, u2); levels below v1 to u1, and levels above v2 to u2
AVERAGE(sbuf, dbuf)	applies a low_pass_filter on image in sbuf and stores result in dbuf
SHARP(sbuf, dbuf)	applies a high_pass_filter on image in sbuf and stores result in dbuf

II. The Element Attribute Extraction (EAE) Algorithms

Statistical Operations

PROFLX	calculates and displays an intensity profile along X_direction of the image on display
PROFLY	calculates and displays an intensity profile along Y_direction of the image on display
HSTOGRAM	calculates and display intensity histogram of the image on display

Image Segmentation

THRESHLDG (t1, t2)	binarizes the image on display with threshold values of (t1, t2)
THRESHMDFY(t1, t2)	binarizes the image on display with initial threshold values of (t1, t2), which are modifiable interactively.
CHAINCODE (sbuf, CodeFile, minilgth)	generates a chain code representation of the object boundary found in the binary image (sbuf), discards objects with boundary length less than minilgth, stores valid chain code in <CodeFile>

Local Feature Extraction

LFEXTRA(CodeFile)	extracts local features from chain code boundary representation; e.g. features like AREA, PERIMETER, CENTROID, ROUNDNESS, MAXIMUM X/Y DIMENSION, and various MOMENTS
CNRPOINT(CodeFile)	extracts corner information from chain code boundary representation
QUIKEXTRA(CodeFile)	extracts simple local features immediately upon finishing reading the chain code stored in <CodeFile>

Geometry Fitting

CIRMATCH(CodeFile)	performs circle match on boundary chain code in <CodeFile>
LMATCH(CodeFile)	performs line match on boundary chain code in <CodeFile>

III. The Transitional Feature Generation (TFG) Algorithms

IDFYSHAPE(<i>FeatureFile</i>)	identifies object shape based on a set of local features specified in < <i>FeatureFile</i> >
CHEKCONN (<i>BE_id</i>)	checks if a particular board element (e.g. a pad) is connected to a track; locates joining points if connected
INSPECONN(<i>x1,y1, x2,y2</i>)	traces a track from one end (<i>x1, y1</i>) to the other (<i>x2, y2</i>); verifies track width and spacing clearance
TRFgen(<i>BE_id, TRFFile</i>)	combine various global BE features and generates a vector of transitional region features (TRFs);

IV. The Global Board Analysis (GBA) Algorithms

Geometrical Relationship Analysis

BETable(<i>Board_id, Source</i>)	extracts a table of expected board elements of a particular board (<i>Board_id</i>), either from a CAD file (i.e. <i>Source</i> = "DESIGN") or from a measured feature list (i.e. <i>Source</i> = "INSPECT")
GEANALYS(<i>Table1, Table2</i>)	examines geometrical relationships between individual board elements of a particular board.

Electrical Interconnection Verification

ERONET	erodes the image currently displayed
DILTNET	dilates the image currently displayed
COMPANET(<i>nlist1, nlist2</i>)	compares to netlists

Global Board Feature Generation

GBFform(<i>GBFFile</i>)	generates a file of global board features in a predefined neutral format, stores output in file <i>GBFFile</i> ;
---------------------------	--

V. The Application Oriented Algorithms (AOAs)

ERREPORT(GBFFile, RepFile)	processes the global board features file <GBFFile> and extracts a report <RepFile> of defects detected.
ERRTREND(DefectType)	generates a bar graph of the specified type of defects (DefectType) detected (If DefectType=0, generates a histogram of all defect-types).
ERRDISP(DefectType)	highlights and superimposes on original image a certain type of defects on video monitor.

VI. Miscellaneous

VISCAL	calibrates X- and Y-scale factors of the optical sub-system of an AOI system.
---------------	---

Appendix B.1

An Example of PDIF Files

```
%*****
% Program : PDIF-OUT VERSION 4.00
% Date : Jan 11 1991
% Time : 03:40:26 PM
% File In : PROJECT.PCB
% File Out : PROJECT.PDF
% Format : P-CAD DATABASE INTERCHANGE FORMAT
%*****
```

```
{COMPONENT PROJECT.PCB
  {ENVIRONMENT
    {PDIFrev 4.00}
    {Program "PDIF-OUT Version 4.00"}
    {DBtype "PC-Board"}
    {DBvrev 1.03}
    {DBtime "Jan. 11, 1997 3:35 p.m. "}
    {DBunit "MIL"}
    {DBgrid 1}
    {Lyrstr "PADCOM" 1 "FLCOMP" 7 "PADSLD" 1 "FLSOLD" 8 "PADINT" 9 "FLINT" 9
      "GNDCON" 10 "FLGCON" 10 "CLEAR" 12 "FLCLER" 12 "PWRCON" 13 "FLPCON" 13
      "SLDMSK" 14 "FLSMSK" 14 "DRILL" 15 "FLDRLL" 15 "PIN" 4 "BRDOUT" 6
      "FLTARG" 4 "SLKSCR" 6 "DEVICE" 5 "ATTR" 6 "REFDES" 6 "COMP" 1
      "SOLDER" 2 "INT1" 14 "INT2" 6 "DRLGIN" 5 "DRLFIN" 6 "PINTOP" 4
      "PINBOT" 3 "MSKGTP" 13 "MSKGBT" 14 "MSKFTP" 8 "MSKFBT" 9 "PSTGTP" 1
      "PSTGBT" 2 "PSTFTP" 12 "PSTFBT" 13 "SLKTOP" 6 "SLKBOT" 5 "DVCTOP" 1
      "DVCBOT" 2 "REFDTP" 3 "REFDBT" 6 "NOTES" 13}
    {Ssymtbl 0 -1 126 126}
    {Lyrphid 23 23 23 24 24 24 25 25 25 26 26 26 29 30 126 30 29 126 31 32 126 32 31 126 33 34
      126 34 33 126 35 36 126 36 35 126 37 38 126 38 37 126 39 40 126 40 39 126 41 42
      126 42 41 126 43 44 126 44 43 126}
  }

  {USER
    {VIEW
      {Mode DETL}
      {Vw 1125 575 5}
      {Lv 24 1 0 1 0 0 0 0 0 0 0 0 0 0 0 0 1 2 0 0 0 0 0 2 1 0
        0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0}
      {Gs 50 50}
    }
  }
}
```

```

(DISPLAY
  [Ly "COMP"] [Ls "SOLID"] [Wd 13] [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
)

(SYMBOL
  (PIN_DEF)
  (PIC)
  (ATR
    (IN
      (Org -32767 -32767) (Ty 255)
    )
  )
)

(DETAIL
  (ANNOTATE
    [Ly "BRDOUT"] [Ls "SOLID"] [Wd 13] [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
    {L 2250 1300 2350 1300} {L 2300 1350 2300 1250} {R -250 -250 2500 1500}
    {L 0 50 0 -50} {L -50 0 50 0}
  )

  (NET_DEF
    (N XN00000
      (DG
        [Ly "COMP"] [Ls "SOLID"] [Wd 13]
        [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
        {W 300 600 500 600 500 650 700 650 700 900 600 900}
      )
      (ATR
        (IN
          {Rats "OFF"}
        )
      )
    )
    (N XN00001
      (DG
        [Ly "COMP"] [Ls "SOLID"] [Wd 13]
        [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
        {W 1250 500 1050 500} {W 200 700 0 700}
        {W 1050 500 1050 450 200 450 200 700 200 1000 0 1000}
      )
      (ATR
        (IN
          {Rats "OFF"}
        )
      )
    )
    (N XN00002
      (DG
        [Ly "COMP"] [Ls "SOLID"] [Wd 13]
        [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
        {W 2050 550 1750 550 1750 300 1550 300 1400 300
          1400 350 900 350 900 400}
      )
      (ATR
        (IN
          {Rats "OFF"}
        )
      )
    )
  )
)

```



```

    }
  }
}
{N +5V
  {DG
    [Ly "COMP"] [Ls "SOLID"] [Wd 40]
    [Ts 80][Tj "LB"][Tr 0][Tm "N"]
    {W 0 900 -100 900} {W -100 900 -100 1150 300 1150}
    {W 2050 1150 1550 1150 900 1150 300 1150 300 1000}
    {W 1550 1000 1550 1150} {W 900 1000 900 1150}
    {W 2050 750 2050 1150 2200 1150}
  }
  {ATR
    {IN
      {Rats "OFF"}
    }
  }
}
}
{N XN00003
  {DG
    [Ly "COMP"] [Ls "SOLID"] [Wd 13]
    [Ts 80][Tj "LB"][Tr 0][Tm "N"] {W 900 800 900 900 1250 900}
  }
  {ATR
    {IN
      {Rats "OFF"}
    }
  }
}
}
{N XN00004
  {DG
    [Ly "COMP"] [Ls "SOLID"] [Wd 13]
    [Ts 80][Tj "LB"][Tr 0][Tm "N"]
    {W 1400 650 1400 1000 1250 1000}
    {W 900 700 1050 700 1050 650 1400 650 2050 650}
  }
  {ATR
    {IN
      {Rats "OFF"}
    }
  }
}
}
{N GND
  {DG
    [Ly "COMP"] [Ls "SOLID"] [Wd 40]
    [Ts 80][Tj "LB"][Tr 0][Tm "N"]
    {W 600 150 0 150 0 400} {W 1250 150 600 150 600 400}
    {W 2050 150 1250 150 1250 300}
    {W 2200 950 2200 150 2050 150 2050 450}
  }
  {ATR
    {IN
      {Rats "OFF"}
    }
  }
}
}
{N XN00005
  {DG
    [Ly "COMP"] [Ls "SOLID"] [Wd 13]

```

```

                                [Ts 80][Tj "LB"][Tr 0][Tm "N"]
                                {W 1550 500 1450 500 1450 400 1250 400}
                                }
                                {ATR
                                {IN
                                {Rats "OFF"}
                                }
                                }
                                }
                                {N XN00006
                                {ATR
                                {IN
                                {Rats "OFF"}
                                }
                                }
                                }
                                {N XN00007
                                {DG
                                [Ly "COMP"] [Ls "SOLID"][Wd 13]
                                [Ts 80][Tj "LB"] [Tr 0][Tm "N"]
                                {W 300 900 300 800 400 800 400 700 600 700}
                                }
                                {ATR
                                {IN
                                {Rats "OFF"}
                                }
                                }
                                }
                                }
                                )

```

```

(PAD_STACK
  {Pad 0 "50R30C.PS"}
  {Pad 1 "60S30C.PS" "60S30N.PS"}
  {Pad 2 "60R30C.PS" "60R30N.PS"}
  {Pad 3 "60R30G.PS"}
  {Pad 4 "60R30P.PS"}
  {Pad 7 "60S38C.PS" "60S38N.PS"}
  {Pad 8 "60R38C.PS" "60R38N.PS"}
  {Pad 19 "40R30C.PS" "40R30N.PS"}
  {Pad 23 "40S30P.PS"}
  {Pad 24 "PHANTOM.PS"}
  {PAD_DEF 50R30C.PS
    {ATR
      {IN
        {Org 0 0}
        {Ty 256}
      }
    }
  }
  {PIC
    [Ly "DRILL"] [Ls "SOLID"][Wd 0]
    [Ts 80][Tj "LB"][Tr 0][Tm "N"]
    {L -25 0 25 0} {L 0 25 0 -25} {C 0 0 25}
    [Ly "PADSLD"] [Wd 20] {C 0 0 15}
    [Ly "PADCOM"] {C 0 0 15}
    [Ly "FLCOMP"] {F1 0 0 3}
    [Ly "FLSOLD"] {F1 0 0 3}
    [Ly "PIN"] [Wd 0] {C 0 0 25} {C 0 0 15}
    [Ly "PADINT"] [Wd 20] {C 0 0 15}
  }

```

```

[Ly "FLINT"] {F1 0 0 3}
[Ly "FLCLER"] {F1 0 0 6}
[Ly "FLSMSK"] {F1 0 0 5}
[Ly "SLDMSK"] [Wd 30] {C 0 0 15}
[Ly "CLEAR"] [Wd 40] {C 0 0 20}
[Ly "FLDRLL"] [Wd 0]
{L 0 30 -30 0 0 -30 30 0 0 30 0 -30} {L -30 0 30 0}
}
}
(PAD_DEF 60S30C.PS
  (ATR
    (IN
      (Org 0 0)
      (Ty 256)
    )
  )
)
(PIC
  [Ly "PIN"] [Ls "SOLID"] [Wd 0] [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
  {R -30 -30 30 30} {C 0 0 15}
  [Ly "PADSLD"] [Wd 24] {R -18 -18 18 18}
  [Ly "FLSOLD"] {F1 0 0 4}
  [Ly "FLCOMP"] {F1 0 0 4}
  [Ly "DRILL"] [Wd 0] {L -30 0 30 0} {L 0 30 0 -30} {C 0 0 30}
  [Ly "PADCOM"] [Wd 24] {R -18 -18 18 18}
  [Ly "PADINT"] [Wd 20] {C 0 0 15}
  [Ly "FLINT"] {F1 0 0 3}
  [Ly "SLDMSK"] [Wd 36] {R -17 -17 17 17}
  [Ly "FLCLER"] {F1 0 0 6}
  [Ly "FLSMSK"] {F1 0 0 18}
  [Ly "CLEAR"] [Wd 40] {C 0 0 20}
  [Ly "FLDRLL"] [Wd 0]
  {L 0 30 -30 0 0 -30 30 0 0 30 0 -30} {L -30 0 30 0}
}
}
(PAD_DEF 60S30N.PS
  (ATR
    (IN
      (Org 0 0)
      (Ty 256)
    )
  )
)
(PIC
  [Ly "PIN"] [Ls "SOLID"] [Wd 0] [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
  {R -30 -30 30 30}
  [Ly "PADSLD"] [Wd 24] {R -18 -18 18 18}
  [Ly "FLSOLD"] {F1 0 0 4} [Ly "FLCOMP"] {F1 0 0 4}
  [Ly "DRILL"] [Wd 0] {L -30 0 30 0} {L 0 30 0 -30} {C 0 0 30}
  [Ly "PADCOM"] [Wd 24] {R -18 -18 18 18}
  [Ly "SLDMSK"] [Wd 36] {R -17 -17 17 17}
  [Ly "FLCLER"] {F1 0 0 6} [Ly "FLSMSK"] {F1 0 0 18}
  [Ly "CLEAR"] [Wd 40] {C 0 0 20}
  [Ly "FLDRLL"] [Wd 0]
  {L 0 30 -30 0 0 -30 30 0 0 30 0 -30} {L -30 0 30 0} {L 79 -49}
}
}
(PAD_DEF 60R30C.PS
  (ATR

```

```

        {IN
            {Org 0 0}
            {Ty 256}
        }
    }

    {PIC
        [Ly "DRILL"] [Ls "SOLID"] [Wd 0]
        [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
        {L -30 0 30 0} {L 0 29 0 -29} {C 0 0 30}
        [Ly "FLSOLD"] {F1 0 0 5} [Ly "FLCOMP"] {F1 0 0 5}
        [Ly "SLDMSK"] [Wd 35] {C 0 0 17}
        [Ly "PADSLD"] [Wd 24] {C 0 0 18}
        [Ly "PADCOM"] {C 0 0 18}
        [Ly "FLSMSK"] {F1 0 0 19} [Ly "PIN"] [Wd 0]
        {C 0 0 15} {C 0 0 30}
        [Ly "FLINT"] {F1 0 0 3} [Ly "PADINT"] [Wd 30] {C 0 0 15}
        [Ly "FLCLER"] {F1 0 0 6} [Ly "CLEAR"] [Wd 46] {C 0 0 22}
        [Ly "FLDRLL"] [Wd 0]
        {L 0 30 -30 0 0 -30 30 0 0 30 0 -30} {L -30 0 30 0}
    }
}

{PAD_DEF 60R30N.PS
    {ATR
        {IN
            {Org 0 0}
            {Ty 256}
        }
    }

    {PIC
        [Ly "DRILL"] [Ls "SOLID"] [Wd 0]
        [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
        {L -30 0 30 0} {L 0 29 0 -29} {C 0 0 30}
        [Ly "FLSOLD"] {F1 0 0 5} "FLCOMP" {F1 0 0 5}
        [Ly "SLDMSK"] [Wd 35] {C 0 0 17}
        [Ly "PADSLD"] [Wd 24] {C 0 0 18}
        [Ly "PADCOM"] {C 0 0 18} [Ly "FLSMSK"] {F1 0 0 19}
        [Ly "PIN"] [Wd 0] {C 0 0 30}
        [Ly "FLCLER"] {F1 0 0 5} [Ly "CLEAR"] [Wd 46] {C 0 0 22}
        [Ly "FLDRLL"] [Wd 0]
        {L 0 30 -30 0 0 -30 30 0 0 30 0 -30} {L -30 0 30 0}
    }
}

{PAD_DEF 60R30G.PS
    {ATR
        {IN
            {Org 0 0} {Ty 256}
        }
    }

    {PIC
        [Ly "DRILL"] [Ls "SOLID"] [Wd 0]
        [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
        {L -30 0 30 0} {L 0 29 0 -29} {C 0 0 30}
        [Ly "FLSOLD"] {F1 0 0 5}
        [Ly "FLCOMP"] {F1 0 0 5} [Ly "SLDMSK"] [Wd 35] {C 0 0 17}
        [Ly "PADSLD"] [Wd 24] {C 0 0 18}
        [Ly "PADCOM"] {C 0 0 18} [Ly "FLSMSK"] {F1 0 0 19}
    }
}

```

```

[Ly "PIN"] [Wd 0] {C 0 0 15} {C 0 0 30}
[Ly "FLINT"] {F1 0 0 3} [Ly "PADINT"] [Wd 30] {C 0 0 15}
[Ly "FLCLER"] {F1 0 0 6}
[Ly "FLGCON"] [Wd 12] {L 35 35 -35 -35} {L 35 -35 -35 35}
[Ly "GNDCON"] {L 35 35 -35 -35} {L 35 -35 -35 35}
[Ly "CLEAR"] [Wd 46] {C 0 0 22}
[Ly "FLDRLL"] [Wd 0] {L 30 0 0 30 0 -30}
{L -30 0 30 0} {L 0 30 -30 0 0 -30 30 0}
}
}
{PAD_DEF 60R30P.PS
  {ATR
    {IN
      {Org 0 0} {Ty 256}
    }
  }
  {PIC
    [Ly "DRILL"] [Ls "SOLID"] [Wd 0]
    [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"]
    {L -30 0 30 0} {L 0 29 0 -29} {C 0 0 30}
    [Ly "FLSOLD"] {F1 0 0 5}
    [Ly "FLCOMP"] {F1 0 0 5} [Ly "SLDMSK"] [Wd 35] {C 0 0 17}
    [Ly "PADSLD"] [Wd 24] {C 0 0 18}
    [Ly "PADCOM"] {C 0 0 18} [Ly "FLSMSK"] {F1 0 0 19}
    [Ly "PIN"] [Wd 0] {C 0 0 15} {C 0 0 30}
    [Ly "FLINT"] {F1 0 0 3} [Ly "PADINT"] [Wd 30] {C 0 0 15}
    [Ly "FLCLER"] {F1 0 0 6} [Ly "CLEAR"] [Wd 46] {C 0 0 22}
    [Ly "PWRCON"] [Wd 12] {L 35 35 -35 -35} {L 35 -35 -35 35}
    [Ly "FLPCON"] {L 35 35 -35 -35} {L 35 -35 -35 35}
    [Ly "FLDRLL"] [Wd 0] {L 0 30 -30 0 0 -30 30 0 0 30 0 -30}
    {L -30 0 30 0}
  }
}
{PAD_DEF 60S38C.PS
  {ATR
    {IN
      {Org 0 0} {Ty 256}
    }
  }
  {PIC
    [Ly "PIN"] [Ls "SOLID"] [Wd 0]
    [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"] {C 0 0 19} {R -30 -30 30 30}
    [Ly "PADSLD"] [Wd 24] {R -18 -18 18 18}
    [Ly "FLSOLD"] {F1 0 0 4} [Ly "FLCOMP"] {F1 0 0 4}
    [Ly "DRILL"] [Wd 0] {L -30 0 30 0} {L 0 30 0 -30} {C 0 0 30}
    [Ly "PADCOM"] [Wd 24] {R -18 -18 18 18}
    [Ly "PADINT"] [Wd 20] {C 0 0 15} [Ly "FLINT"] {F1 0 0 5}
    [Ly "SLDMSK"] [Wd 36] {R -17 -17 17 17}
    [Ly "FLCLER"] {F1 0 0 22} [Ly "FLSMSK"] {F1 0 0 18}
    [Ly "CLEAR"] [Wd 40] {C 0 0 20}
    [Ly "FLDRLL"] [Wd 0] {L -30 30 -30 -30 30 -30 30 30 -30 30}
    {L 0 30 0 -30} {L -30 0 30 0} {L 74 -49}
  }
}
{PAD_DEF 60S38N.PS
  {ATR
    {IN

```

```

        {Org 0 0} {Ty 256}
    }
}
{PIC
    [Ly "PIN"] [Ls "SOLID"] [Wd 0]
    [Ts 80] [Tj "LB"] [Tr 0] [Tm "N"] {R -30 -30 30 30}
    [Ly "PADSLD"] [Wd 24] {R -18 -18 18 18}
    [Ly "FLSOLD"] {F1 0 0 4} [Ly "FLCOMP"] {F1 0 0 4}
    [Ly "DRILL"] [Wd 0] {L -30 0 30 0} {L 0 30 0 -30} {C 0 0 30}
    [Ly "PADCOM"] [Wd 24] {R -18 -18 18 18}
    [Ly "SLDMSK"] [Wd 36] {R -17 -17 17 17}
    [Ly "FLCLER"] {F1 0 0 5} [Ly "FLMSK"] {F1 0 0 18}
    [Ly "CLEAR"] [Wd 40] {C 0 0 20}
    [Ly "FLDRLL"] [Wd 0] {L -30 30 -30 -30 30 -30 30 30 -30 30}
    {L 0 30 0 -30} {L -30 0 30 0}
}
}

{PAD_DEF 60R38C.PS
    {ATR
        {IN
            {Org 0 0} {Ty 256}
        }
    }
    {PIC
        [Ly "DRILL"] [Ls "SOLID"] [Wd 0] [Ts 80] [Tj "LB"]
        [Tr 0] [Tm "N"] {L -30 0 30 0} {L 0 29 0 -29} {C 0 0 30}
        [Ly "FLSOLD"] {F1 0 0 5} [Ly "FLCOMP"] {F1 0 0 5}
        [Ly "SLDMSK"] [Wd 35] {C 0 0 17}
        [Ly "PADSLD"] [Wd 24] {C 0 0 18} [Ly "PADCOM"] {C 0 0 18}
        [Ly "FLMSK"] {F1 0 0 19}
        [Ly "PIN"] [Wd 0] {C 0 0 19} {C 0 0 30} [Ly "FLINT"] {F1 0 0 5}
        [Ly "PADINT"] [Wd 30] {C 0 0 15}
        [Ly "FLCLER"] {F1 0 0 22} [Ly "CLEAR"] [Wd 46] {C 0 0 22}
        [Ly "FLDRLL"] [Wd 0] {L 30 30 -30 30 -30 -30 30 -30 30 30}
        {L 0 30 0 -30} {L -30 0 30 0}
    }
}

{PAD_DEF 60R38N.PS
    {ATR
        {IN
            {Org 0 0} {Ty 256}
        }
    }
    {PIC
        [Ly "DRILL"] [Ls "SOLID"] [Wd 0] [Ts 80] [Tj "LB"] [Tr 0]
        [Tm "N"] {L -30 0 30 0} {L 0 29 0 -29} {C 0 0 30}
        [Ly "FLSOLD"] {F1 0 0 5} [Ly "FLCOMP"] {F1 0 0 5}
        [Ly "SLDMSK"] [Wd 35] {C 0 0 17}
        [Ly "PADSLD"] [Wd 24] {C 0 0 18}
        [Ly "PADCOM"] {C 0 0 18} [Ly "FLMSK"] {F1 0 0 19}
        [Ly "PIN"] [Wd 0] {C 0 0 30} [Ly "FLCLER"] {F1 0 0 19}
        [Ly "CLEAR"] [Wd 46] {C 0 0 22}
        [Ly "FLDRLL"] [Wd 0] {L 0 30 0 -30} {L -30 0 30 0}
        {L -30 30 -30 -30 30 -30 30 30 -30 30}
    }
}

```

```

(PAD_DEF 40R30C.PS
  (ATR
    (IN
      (Org 0 0) (Ty 256)
    )
  )
  (PIC
    [Ly "DRILL"] [Ls "SOLID"] [Wd 0] [Ts 80] [Tj "LB"] [Tr 0]
    [Tm "N"] [L -20 0 20 0] [L 0 20 0 -20] [C 0 0 20]
    [Ly "PADSLD"] [Wd 16] [C 0 0 12] [Ly "PIN"] [Wd 0]
    [C 0 0 20] [C 0 0 15] [Ly "FLSMSK"] [Fi 0 0 3]
    [Ly "PADCOM"] [Wd 16] [C 0 0 12] [Ly "SLDMSK"] [Wd 26]
    [C 0 0 12] [Ly "FLCOMP"] [Fi 0 0 2]
    [Ly "FLSOLD"] [Fi 0 0 2] [Ly "PADINT"] [Wd 16] [C 0 0 12]
    [Ly "FLINT"] [Fi 0 0 3]
    [Ly "FLCLER"] [Fi 0 0 6] [Ly "CLEAR"] [Wd 36] [C 0 0 18]
    [Ly "FLDRLL"] [Wd 0]
    [L 0 15 0 -15] [L -15 0 15 0 0 15 -15 0 0 -15 15 0]
  )
)

```

```

(PAD_DEF 40R30N.PS
  (ATR
    (IN
      (Org 0 0) (Ty 256)
    )
  )
  (PIC
    [Ly "DRILL"] [Ls "SOLID"] [Wd 0] [Ts 80] [Tj "LB"] [Tr 0]
    [Tm "N"] [L -20 0 20 0] [L 0 20 0 -20] [C 0 0 20]
    [Ly "PADSLD"] [Wd 16] [C 0 0 12] [Ly "PIN"] [Wd 0]
    [C 0 0 20] [Ly "FLSMSK"] [Fi 0 0 3]
    [Ly "PADCOM"] [Wd 16] [C 0 0 12] [Ly "SLDMSK"] [Wd 26]
    [C 0 0 12] [Ly "FLCOMP"] [Fi 0 0 2]
    [Ly "FLSOLD"] [Fi 0 0 2] [Ly "FLCLER"] [Fi 0 0 5]
    [Ly "CLEAR"] [Wd 36] [C 0 0 18] [Ly "FLDRLL"] [Wd 0]
    [L 0 15 0 -15] [L -15 0 15 0 0 -15 -15 0 0 15 15 0]
  )
)

```

```

(PAD_DEF 40S30P.PS
  (ATR
    (IN
      (Org 0 0)
      (Ty 256)
    )
  )
  (PIC
    [Ly "DRILL"] [Ls "SOLID"] [Wd 0] [Ts 80] [Tj "LB"] [Tr 0]
    [Tm "N"] [L -20 0 20 0] [L 0 20 0 -20] [C 0 0 20]
    [Ly "PIN"] [L -20 20 20 20 20 -20 -20 -20 -20 20] [C 0 0 13]
    [Ly "PADCOM"] [Wd 16] [R -12 -12 12 12]
    [Ly "PADSLD"] [R -12 -12 12 12] [Ly "FLCOMP"] [Fi 0 0 1]
    [Ly "FLSOLD"] [Fi 0 0 1]
    [Ly "FLSMSK"] [Fi 0 0 21] [Ly "SLDMSK"] [Wd 26]
    [R -12 -12 12 12]
    [Ly "FLCLER"] [Fi 0 0 6] [Ly "FLINT"] [Fi 0 0 3]
    [Ly "PADINT"] [Wd 20] [C 0 0 15]
    [Ly "CLEAR"] [Wd 40] [C 0 0 20] [Ly "PWRCON"] [Wd 12]
  )
)

```

```

        {L 30 30 -30 -30}{L -30 30 30 -30}
        [Ly "FLPCON"]{L 30 30 -30 -30}{L -30 30 30 -30}
        [Ly "FLDRLL"]{Wd 0}{L 0 15 0 -15}
        {L -15 0 15 0 0 -15 -15 0 0 15 15 0}
    }
}

{PAD_DEF PHANTOM.PS
    {ATR
        {IN
            {Org 0 0} {Ty 255}
        }
    }
    {PIC
        [Ly "NOTES"]{Ls "DOTTED"}{Wd 0}{Ts 80}{Tj "LB"}
        [Tr 0]{Tm "N"}{C 0 0 30}
    }
}

{SUBCOMP
    {COMP_DEF 74LS00.PRT
        {PIN_DEF
            {Ly "PIN"}
            {P 1 {Pt 1}{Lq 1}{Ploc 0 0}}
            {P 2 {Pt 2}{Lq 1}{Ploc 0 -100}}
            {P 3 {Pt 2}{Lq 0}{Ploc 0 -200}}
            {P 4 {Pt 2}{Lq 2}{Ploc 0 -300}}
            {P 5 {Pt 2}{Lq 2}{Ploc 0 -400}}
            {P 6 {Pt 2}{Lq 0}{Ploc 0 -500}}
            {P 7 {Pt 3}{Lq 0}{Ploc 0 -600}}
            {P 8 {Pt 2}{Lq 0}{Ploc 300 -600}}
            {P 9 {Pt 2}{Lq 3}{Ploc 300 -500}}
            {P 10 {Pt 2}{Lq 3}{Ploc 300 -400}}
            {P 11 {Pt 2}{Lq 0}{Ploc 300 -300}}
            {P 12 {Pt 2}{Lq 4}{Ploc 300 -200}}
            {P 13 {Pt 2}{Lq 4}{Ploc 300 -100}}
            {P 14 {Pt 4}{Lq 0}{Ploc 300 0}}
        }
        {SPKG
            {Sna A B C D}{Sp OUTY 3 6 8 11}{Sp INA 1 4 9 12}
            {Sp INB 2 5 10 13}
        }
        {PIC
            [Ly "SLKSCR"]{Ls "SOLID"}{Wd 0}
            [Ts 80]{Tj "LB"}{Tr 0}{Tm "N"}
            {L 100 50 150 0 200 50}{L 50 50 50 -650 250 -650 250 50 50 50}
            [Ly "DEVICE"]{Ts 125}{Tj "CC"}{Tr 3}
            {T "74LS00" 150 -300}
        }
    }
    {ATR
        {IN
            {Ty 10000}
        }
    }
}

{I 74LS00.PRT XC00000
    {CN XN00001 +5V ? XN00001 XN00006 ? GND ? ?
        XN00000 ? XN00007 XN00007 +5V}
}

```



```

{ATR
    {IN
        {PI 0 1000}
    }
    {EX
        [Ly "ATTR"] [Ts 40][Tj "CB"] [Tr 0][Tm "N"]
        {At FP DIP14 150 -650}
    }
}

[I 74LS00.PRT XC00001
    {CN ? XN00000 ? XN00007 ? ? GND XN00002 ? ?
        XN00004 XN00003 XN00003 +5V}
    {ATR
        {IN
            {PI 600 1000}
        }
        {EX
            [Ly "ATTR"]
            [Ts 40][Tj "CB"] [Tr 0][Tm "N"]
            {At FP DIP14 150 -650}
        }
    }
}

[COMP_DEF 4164.PRT
    {PIN_DEF
        [Ly "PIN"]
        {P 1 {Pt 1} {Lq 0} {Ploc 0 0}}
        {P 2 {Pt 2} {Lq 0} {Ploc 0 -100}}
        {P 3 {Pt 2} {Lq 0} {Ploc 0 -200}}
        {P 4 {Pt 2} {Lq 0} {Ploc 0 -300}}
        {P 5 {Pt 2} {Lq 0} {Ploc 0 -400}}
        {P 6 {Pt 2} {Lq 0} {Ploc 0 -500}}
        {P 7 {Pt 2} {Lq 0} {Ploc 0 -600}}
        {P 8 {Pt 4} {Lq 0} {Ploc 0 -700}}
        {P 9 {Pt 2} {Lq 0} {Ploc 300 -700}}
        {P 10 {Pt 2} {Lq 0} {Ploc 300 -600}}
        {P 11 {Pt 2} {Lq 0} {Ploc 300 -500}}
        {P 12 {Pt 2} {Lq 0} {Ploc 300 -400}}
        {P 13 {Pt 2} {Lq 0} {Ploc 300 -300}}
        {P 14 {Pt 2} {Lq 0} {Ploc 300 -200}}
        {P 15 {Pt 2} {Lq 0} {Ploc 300 -100}}
        {P 16 {Pt 3} {Lq 0} {Ploc 300 0}}
    }
    {SPKG
        {Sna A} {Sp CAS' 15} {Sp O 14} {Sp A6 13} {Sp A3 12}
        {Sp A4 11} {Sp A5 10} {Sp A7 9} {Sp A1 7} {Sp A2 6}
        {Sp A0 5} {Sp RAS' 4} {Sp R/W' 3} {Sp D 2}
    }
    {PIC
        [Ly "SLKSCR"] [Ls "SOLID"] [Wd 0]
        [Ts 40][Tj "CB"] [Tr 0][Tm "N"]
        {L 50 -750 250 -750 250 50 50 50 50 -750}
        {L 100 50 150 0 200 50}
        [Ly "DEVICE"] [Ts 125][Tj "CC"] [Tr 3] [T "4164" 150 -350]
    }
    {ATR

```

```

        {IN
          {Ty 10000}
        }
      }
    }

(I 4164.PRT 256
  {CN XN00002 ? XN00005 ? ? ? ? +5V XN00004 XN00003 ? ? ?
    XN00001 XN00005 GND}
  {ATR
    {IN
      {Pl 1550 300}
      {Ro 2}
    }
    {EX
      [Ly "ATTR"] [Ts 45] [Tj "CB"] [Tr 0] [Tm "N"]
      {At FP DIP16 -150 750}
    }
  }
}

{COMP_DEF CK06.PRT
  {PIN_DEF
    [Ly "PIN"] {P 1 {Pt 2} {Lq 1} {Ploc 0 0}} {P 2 {Pt 2} {Lq 1}
      {Ploc 0 -200}}
  }
  {SPKG
    {Sna A} {Sp 1 1} {Sp 2 2}
  }
  {PIC
    [Ly "SLKSCR"] [Ls "SOLID"] [Wd 0] [Ts 45] [Tj "CB"] [Tr 0]
    [Tm "N"] {L -1 50 0 50} {L -1 -250 0 -250}
    {Arc 0 -200 -50 -200 50 -200} {Arc 0 0 50 0 -50 0}
    {L -50 -200 -50 0} {L 50 0 50 -200}
    [Ly "DEVICE"] [Ts 60] [Tj "CC"] [Tr 3] {T "CK06" -100 -100}
  }
  {ATR
    {IN
      {Ty 11200}
    }
  }
}

(I CK06.PRT XC00002
  {CN +5V GND} {IPT 19 19}
  {ATR
    {IN
      {Pl 2200 1150}
    }
    {EX
      [Ly "ATTR"] [Ts 25] [Tj "CC"] [Tr 3] [Tm "N"]
      {At FP CK06 -30 -100}
    }
  }
}

{COMP_DEF PIN4A.PRT
  {PIN_DEF
    [Ly "PIN"]
  }
}

```

```

(P 1 {Pt 8}{Lq 0}{Ploc 0 0})
(P 2 {Pt 8}{Lq 0}{Ploc 0 -100})
(P 3 {Pt 8}{Lq 0}{Ploc 0 -200})
(P 4 {Pt 8}{Lq 0}{Ploc 0 -300})
}
{SPKG
  {Sna A B C D}{Sp PIN 1 2 3 4}
}
{PIC
  [Ly "SLKSCR"][Ls "SOLID"][Wd 0][Ts 60][Tj "CC"][Tr 0]
  [Tm "N"] {T "1" -100 50}{R -50 -350 50 50}
  [Ly "DEVICE"][T "PIN4A" 0 -400]
}
{ATR
  {IN
    {Ty 12000}
  }
}
}

(I PIN4A.PRT XC00003
  {CN +5V XN00004 XN00002 GND}
  {ATR
    {IN
      {PI 2050 750}
    }
    {EX
      [Ly "ATTR"][Ts 25][Tj "CC"][Tr 0][Tm "N"]
      {At FP PIN4A 0 -250}
    }
  }
}
}
}
}
}
```

Appendix B.2

YACC Code for Software Information Generators

```

/*****
*****
This is the Yacc code written by the author for the purpose of information extraction from
a PDIF file. The compiled Lex code (source code is given in Figure 5.7 of Chapter 5)
needs to be included to perform lexical analysis of the input PDIF file and return "tokens"
for the Yacc code to perform "grammar rule match".
The Yacc code listed here is composed of two parts. The first part is the "grammar rules"
specified according to the PDIF data structure and representation convention. A complete
list of this part is given below. The second part is a collection of subroutines which are
called when a certain specified "rule" is matched.
*****/

```

```

%{
char    str[128], buffer[128], current_lyr[20], idcard[20], idbuf[20], insname[20];
int      current_width, ii=0, xxo, yyo, locx, locy, pintype, placex, placey, rotation;
int      outdim1, outdim2, inndim1, inndim2, orient, origin_x, origin_y;
char     shape[2], netbuf[128];
char     padcom[10]={"", 'P', 'A', 'D', 'C', 'O', 'M', ''}, brdout[10]={"", 'B', 'R', 'D', 'O', 'U', 'T', ''};

struct coord    {      int x;
                      int y;
                    };

struct cd_list   {      struct cd_list *next;
                      struct coord point;
                    } coord_list = {0};

struct wire      {      struct coord start;
                      struct coord end;
                      int width;
                    };

struct wire_list {      struct wire_list *next;
                      struct wire *track;
                    } network = {0};

struct fdmark    {      struct fdmark *next;
                      struct coord onepoint;
                    } FmkSegLst = {0};

struct pfile     {      int type, orient;
                      char name1[20], name2[20];
                      char shape[5];
                      struct coord outdim, inndim, center;
                    };

struct pf_list   {      struct pf_list *next;
                      struct pfile padfile;
                    } pfile_list = {0};

struct pin       {      char owner[20];

```

```

        int pin;
        int type;
        struct coord locatn;
    };

    struct pinlist    (        struct pinlist *next;
                            struct pin apin;
    )parts={0};

    struct instance    (        char name[20];
                            struct coord place;
                            int rotation;
    );

    struct inst_list    (        struct inst_list *next;
                            struct instance apart;
    ) comps={0};

%}

%start comp
/*:::::::::::: The Following Is a List of Tokens Expected to be Returned from the Lexical Analyser ::::::::::*/
%token  COMP ENVIRTOK USERTOK DISPTOK SYMBLTOK DETALTOK ANNOTTOK
%token  ATRTOK PICTOK PKGTOK SPKGTOK SUBTOK VIEWTOK
%token  ASGTOK CNTOK DGTOK EXTOK INTOK IPTTOK
%token  COMP_DEF NET_DEF PAD_DEF PAD_STAK PIN_DEF
%token  PDIFVREV PROGRAM DBTYPE DBVREV DBTIME DBUNIT DBGRID
%token  LYRSTR LYRPHID SSYMTBL APR
%token  CIRCLE INSTANCE LINE NET PIN RECTANGL TEXT VIA WIRE
%token  APN ARC AT CV FL FR GS IAT JMP LQ LS LV LY  MODE MR NL NN NS OL ORG
%token  PA PAD PID PL PLOC PN PNL POLY PS PT PV RATS RD RDL RO SC SD SMD SNA
%token  SP TJ TM TR TS TY UN VW WD
%token  STRING CHAR DIGIT
%token  LCURL RCURL LSQR RSQR NEG IDCARD
%%

/*:::::::::::: The Following is a List of Grammar Rule Specifications Based on PDIF Convention :::::::::: */

comp      :      LCURL COMP IDCARD enviro user display symbol detail RCURL
          ;

/*:::::::::::: ENVIRONMENT Section Follows ::::::::::*/

enviro    :      LCURL ENVIRTOK envlst RCURL
          ;
envlst    :      /* null */
          |      envlst pdfvrev
          |      envlst program
          |      envlst dbtype
          |      envlst dbvrev
          |      envlst dbtime
          |      envlst dbunit
          |      envlst dbgrid
          |      envlst lyrstr
          |      envlst ssymtbl
          |      envlst lyrphid
          ;

```

```

pdfvrev :      LCURL PDIFVREV number ':' number RCURL
               { printf("\nversion \t%d.%d\n", $3, $5);}
;

program :      LCURL PROGRAM STRING RCURL
               { printf("program \t%s\n", str);}
;

dbtype :      LCURL DBTYPE STRING RCURL
               { printf("dbtype \t\t%s\n", str);}
;

dbvrev :      LCURL DBVREV number ':' number RCURL
;

dbtime :      LCURL DBTIME STRING RCURL
               { printf("dbtime \t\t%s\n", str);}
;

dbunit :      LCURL DBUNIT STRING RCURL
               { printf("dbunit \t\t%s\n", str);}
;

dbgrid :      LCURL DBGRID number RCURL
;

lyrstr :      LCURL LYRSTR lyrst RCURL
;

lyrphid :      LCURL LYRPHID numset RCURL
;

ssymtbl :      LCURL SSYMTBL numset RCURL
;

lyrst :      /* null */
              | lyrst STRING
              | lyrst number
;

/*..... USER Section Follows .....*/

user :      LCURL USERTOK viewlst RCURL
;

viewlst :      /* null */
              | viewlst view
;

view :      LCURL VIEWTOK mvlgs RCURL
;

mvlgs :      /* null */
              | mvlgs mode
              | mvlgs vw
              | mvlgs lv
              | mvlgs gs
;

mode :      LCURL MODE IDCARD RCURL

```

```

vw      :      LCURL VW numset RCURL
;
lv      :      LCURL LV numset RCURL
;
gs      :      LCURL GS numset RCURL
;

/*..... DISPLAY Section Follows .....*/

display :      LCURL DISPTOK piclist RCURL
;

/*..... SYMBOL Section Follows .....*/

symbol  :      LCURL SYMBLTOK symlist RCURL
;
symlist :      /* null */
|              symlist pindef
|              symlist pic
|              symlist atr
;

/*..... DETAIL Section Follows .....*/

detail  :      LCURL DETALTOK detlist RCURL
|              { print_pad(&comps, &parts, &file_list); }
;

detlist :      annot
|              detlist annot
|              detlist net_def
|              detlist pad_stk
|              detlist subcomp
;

/*..... DETAIL / ANNOTATE .....*/

annot   :      LCURL ANNOTTOK piclist RCURL
|              { print_fdmk(&FmkSegLst); }

/*..... DETAIL / NET_DEF .....*/

net_def :      LCURL NET_DEF netlist RCURL
|              { pr_net(&network); kill_net(&network); }
;
netlist :      /* null */
|              netlist net
;
net      :      LCURL NET IDCARD dgatr RCURL
;
dgatr    :      /* null */
|              dgatr dg
|              dgatr atr
;
dg       :      LCURL DGTOK ly ls wd ts tj tr tm wlist RCURL
;
wlist    :      w

```

```

      |      wlist w
      ;
w      :      LCURL WIRE co_set RCURL
      |      { add_wires(&network, &coord_list);}
      ;
co_set :      co_ord
      |      co_set co_ord
      ;
co_ord :      number number
      |      { add_coord(&coord_list, $1, $2);}
      ;

/* .....DETAIL / PAD_STACK .....*/
pad_stk :      LCURL PAD_STAK padlist RCURL
      ;
padlist :      /* null */
      |      padlist padtype
      |      padlist def
      ;
padtype :      LCURL PAD number STRING RCURL
      |      {strcpy(buffer, str); strcpy(str, "\0"); add_pfile1(&pfile_list, $3);}
      |      LCURL PAD number STRING STRING RCURL
      |      { add_pfile1(&pfile_list, $3);}
      ;
def      :      LCURL PAD_DEF IDCARD atr pic RCURL
      |      {add_pfile2(&pfile_list); }
      ;
atr      :      LCURL ATRTOK in ex RCURL
      |      LCURL ATRTOK in RCURL
      ;
in      :      LCURL INTOK inlist RCURL
      ;
inlist   :      /* null */
      |      inlist org
      |      inlist ty
      |      inlist smd
      |      inlist jmp
      |      inlist rats
      ;
org      :      LCURL ORG number number RCURL
      |      {origin_x=$3; origin_y=$4;}
      ;
ty      :      LCURL TY number RCURL
      ;
smd      :      LCURL SMD STRING RCURL
      ;
jmp      :      LCURL JMP STRING RCURL
      ;
rats      :      LCURL RATS STRING RCURL
      ;
pic      :      LCURL PICTOK piclist RCURL
      ;
piclist  :      /* null */
      |      piclist ly
      |      piclist ls
      |      piclist wd
      |      piclist ts
      |      piclist tj
      |      piclist tr

```



```

|      piclist tm
|      piclist circle
|      piclist line
|      piclist flash
|      piclist rect
|      piclist text
|      piclist frect
|      piclist arc
;
line   :      LCURL LINE pt_set RCURL
;
circle :      LCURL CIRCLE number number number RCURL
              { if(strcmp(current_lyr, padcom)==0) add_circle($3, $4, $5); }
;
flash  :      LCURL FL numset RCURL
;
frect  :      LCURL FR numset RCURL
;
rect   :      LCURL RECTANGL number number number number RCURL
              { if(strcmp(current_lyr, padcom)==0) add_rect($3, $4, $5, $6); }
;
text   :      LCURL TEXT STRING numset RCURL
;
arc     :      LCURL ARC numset RCURL
;
pt_set  :      apoint
|              pt_set apoint
;
apoint  :      number number
              { if(strcmp(current_lyr, brdout)==0) add_a_mkpnt(&FmkSegLst, $1, $2); }
;

/*..... DETAIL / SUBCOMP .....*/
subcomp :      LCURL SUBTOK cilist RCURL
;
cilist  :      /* null */
|              cilist compdef
|              cilist instans
;

/*..... DETAIL / SUBCOMP / COMP_DEF .....*/
compdef :      LCURL COMP_DEF IDCARD pindef spkg pic atr RCURL
;
pindef  :      LCURL PIN_DEF pdef RCURL
;
pdef    :      /* null */
|              pdef ly pinlst
;
pinlst  :      /* null */
|              pinlst pins
;
pins    :      LCURL PIN number pt lq ploc RCURL
              { add_pin(&(parts), $3); }
;
pt       :      LCURL PT number RCURL
              { pintype=$3; }
;
lq       :      LCURL LQ number RCURL
;
ploc    :      LCURL PLOC number number RCURL

```

```

                                {locx=$3; locy=$4;}

spkg      :      LCURL SPKGTOK snasp RCURL
;
snasp     :      /* null */
              |      snasp sna
              |      snasp spp
;
sna       :      LCURL SNA {skipper();} RCURL
;
spp       :      LCURL SP spname numset RCURL
;
spname    :      IDCARD
              |      DIGIT
;
/*.....DETAIL / SUBCOMP / INSTANCES .....*/
instans   :      LCURL INSTANCE IDCARD invname instlst RCURL
              |      {add_inst(&comps);}
;
invname    :      /* null */
              |      {strcpy(invname, idcard);}
              |      DIGIT
              |      {strcpy(invname, idcard);}
              |      IDCARD
              |      {strcpy(invname, idbuf);}
;
instlst    :      cn
              |      instlst cn
              |      instlst ipt
              |      instlst asg
              |      instlst iatr
;
cn         :      LCURL CNTOK { skipper();} RCURL
;
ipt        :      LCURL IPTTOK numset RCURL
;
asg        :      LCURL ASGTOK asglist RCURL
;
asglist    :      /* null */
              |      asglist rd
              |      asglist pn
;
rd         :      LCURL RD STRING numset RCURL
;
pn         :      LCURL PN STRING numset RCURL
;
;
iatr       :      LCURL ATRTOK iin ex RCURL
;
iin        :      LCURL INTOK iinlst RCURL
;
iinlst     :      /* null */
              |      iinlst pl
              |      iinlst sc
              |      iinlst ro
              |      iinlst mr
              |      iinlst ps
              |      iinlst pa
              |      iinlst nl

```

```

      |      iinlst un
      |      iinlst iat
      ;
pl      :      LCURL PL number number RCURL
           { placex=$3; placey=$4; }
      ;
sc      :      LCURL SC numset RCURL
      ;
ro      :      LCURL RO number RCURL
           { rotation=$3; }
      ;
mr      :      LCURL MR STRING RCURL
      ;
ps      :      LCURL PS STRING RCURL
      ;
pa      :      LCURL PA number RCURL
      ;
nl      :      LCURL NL numset RCURL
      ;
un      :      LCURL UN STRING RCURL
      ;
iat     :      LCURL IAT IDCARD IDCARD RCURL
      ;
ex      :      LCURL EXTOK piclist at RCURL
      ;
at      :      LCURL AT IDCARD IDCARD numset RCURL
      ;
ly      :      LSQR LY STRING RSQR
           { strcpy(current_lyr,str); }
      ;
ls      :      LSQR LS STRING RSQR
      ;
wd      :      LSQR WD number RSQR
           { current_width = $3; }
      ;
ts      :      LSQR TS number RSQR
      ;
tj      :      LSQR TJ STRING RSQR
      ;
tr      :      LSQR TR number RSQR
      ;
tm      :      LSQR TM STRING RSQR
      ;

/*..... General Numbers.....*/
numset  :      number
      |      numset number
      ;
number  :      DIGIT
           { $$ = $1; }
      |      NEG DIGIT
           { $$ = -$2; }
      ;

/*..... End of Grammar Rule Specifications .....*/

/*..... Subroutines to be Called When a Certain Gramar Rule is Matched .....*/
%%
#include<string.h>
```

```

#include<stdio.h>
#include<math.h>
#include "lex.yy.c"          /* to include the compiled Lex code for returning Tokens */

/*..... Special Routines Used by YACC .....*/
yyerror(s) char *s;
{ fprintf(stderr,"yacc - %s\n",s); }

yywrap()
{ return 1; }

main()
{ return(yyparse()); }

/*..... Information Extraction & Representation Formatting Routines .....*/

/*..... Extract Fiducial Mark Information .....*/
add_a_mkpoint(fmp, x, y) struct fdmark *fmp; int x, y;
{
    while (fmp->next) fmp=fmp->next;
    fmp->next=(struct fdmark *)malloc(sizeof(struct fdmark)); fmp->next->next = 0;
    fmp->onepoint.x = x; fmp->onepoint.y = y;
}

print_fdmk(fmp) struct fdmark *fmp;          /* print fiducial marks */
{
    int xo, yo, lx, ly, xs, ys, xe, ye, nf=1;
    printf("\n\nFDMK NO.\tLocation\tX_dimension\tY_Dimension\tLine_width\n");
    while (fmp->next)
    {
        xs = fmp->onepoint.x;    ys = fmp->onepoint.y;
        fmp=fmp->next;
        xe = fmp->onepoint.x; ye = fmp->onepoint.y;
        if(xs == xe)    {xo=xs; ly = abs(ys-ye);}
        else           {yo=ys; lx=abs(xs-xe);}

        fmp=fmp->next; xs = fmp->onepoint.x; ys = fmp->onepoint.y;
        fmp=fmp->next;
        xe = fmp->onepoint.x; ye = fmp->onepoint.y;
        if(xs == xe)    {xo=xs; ly = abs(ys-ye);}
        else           {yo=ys; lx=abs(xs-xe);}
        printf("FDMK%d\t\t%d\t\t%d\t\t%d\t\t%d\n", nf, xo, yo, lx, ly, current_width);
        fmp=fmp->next; nf++;
    }
}

/*..... Extract Track Information .....*/
add_coord(cp,x,y) struct cd_list *cp; int x,y;
{
    while(cp->next)cp = cp->next;
    cp->next = (struct cd_list *)malloc(sizeof(struct cd_list)); cp->next->next = 0;
    cp->point.x = x; cp->point.y = y;
}

add_wires(net,c_lst) struct wire_list *net; struct cd_list *c_lst;
{
    int i;    struct cd_list *cp;
    cp = c_lst;
    if(!cp->next) return;
    while(cp->next->next)
    {
        while(net->next)net = net->next;
        net->next=(struct wire_list *)malloc(sizeof(struct wire_list)); net->next->next = 0;
        net->track = (struct wire *)malloc(sizeof(struct wire));
    }
}

```

```

        net->track->start.x = cp->point.x; net->track->start.y = cp->point.y;
        net->track->end.x = cp->next->point.x; net->track->end.y = cp->next->point.y;
        net->track->width = current_width;
        cp = cp->next;
        kill_cd_list(c_lst);
    }

kill_cd_list(tp) struct cd_list *tp;
{
    if(!tp->next) return;
    while(tp->next->next) kill_cd_list(tp->next);
    free((char *)tp->next); tp->next = 0;
}

pr_net(net) struct wire_list *net;      /* print tracks */
{
    int continued, trackend, xe, ye;
    trackend=-1;
    while(net->next)
    {
        printf("\nTRACK %d ", net->track->width);
        continued=1;
        printf("%d %d ", net->track->start.x, net->track->start.y);
        xe=net->track->end.x; ye=net->track->end.y;
        while(continued==1)
        {
            printf("%d %d ", xe, ye);
            if(net->next->next)
            {
                net=net->next;
                if((xe == net->track->start.x) && (ye == net->track->start.y))
                    { xe = net->track->end.x; ye=net->track->end.y; continued=1;}
                else {printf("%d\n", trackend); continued=0;}
            }
            else {printf("%d\n", trackend); return; }
        }
    }
}

kill_net(net) struct wire_list *net;
{
    if(!net->next) return;
    while(net->next->next) kill_net(net->next); free((char *)net->next);
    net->next = 0; free((char *)net->track);
}

/*.....Extract pad Information .....*/
add_rect(aa, bb, cc, dd) int aa, bb, cc, dd;
{
    shape[0]='S'; shape[1]='Q'; shape[2]='R'; shape[3]='\0';
    outdim1=abs(cc-aa)+current_width; outdim2=abs(dd-bb)+current_width;
    inndim1=outdim1-2*current_width; inndim2=outdim2-2*current_width;
    orient=(outdim1>=outdim2) ? ((inndim1>=inndim2) ? 0 : 1) : ((inndim1>=inndim2) ? 2 : 3);
}

add_circle(aa, bb, cc) int aa, bb, cc;
{
    shape[0]='R'; shape[1]='N'; shape[2]='D'; shape[3]='\0';
    origin_x=aa; origin_y=bb;
    outdim1=cc+current_width/2; inndim1=outdim1-current_width;
    outdim2 = -1; inndim2 = -1; orient=0;
}

add_pfile1(pfp, padtype) struct pf_list *pfp; int padtype;
{
    while(pfp->next) pfp=pfp->next;
    pfp->next = (struct pf_list *)malloc(sizeof(struct pf_list)); pfp->next->next = 0;
    pfp->padfile.type=padtype; strcpy((pfp->padfile.name1), buffer); strcpy((pfp->padfile.name2), str);
}

```

```

}

add_pfile2(pfp) struct pf_list *pfp;
{
    while( (strcmp(pfp->padfile.name1,idcard)!=0) && (strcmp(pfp->padfile.name2,idcard)!=0)
           && (pfp!=NULL) ) pfp=pfp->next;

    if ((strcmp(pfp->padfile.name2,idcard)==0) || (pfp==NULL)) return;
    pfp->padfile.orient=orient; strcpy((pfp->padfile.shape), shape);
    pfp->padfile.outdim.x=outdim1; pfp->padfile.outdim.y=outdim2;
    pfp->padfile.inndim.x=inndim1; pfp->padfile.inndim.y=inndim2;
    pfp->padfile.center.x=origin_x; pfp->padfile.center.y=origin_y;

    orient= 0; outdim1= 0; outdim2= 0;
    inndim1= 0; inndim2= 0; origin_x= 0; origin_y= 0;
    shape[0]='*'; shape[1]='\0';
}

add_pin(pp, pnum) struct pinlist *pp; int pnum;
{
    while(pp->next)pp=pp->next;
    pp->next=(struct pinlist *)malloc(sizeof(struct pinlist)); pp->next->next=0;
    strcpy(pp->apin.owner, idcard);
    pp->apin.pin = pnum; pp->apin.type = pintype;
    pp->apin.locatn.x = locx; pp->apin.locatn.y = locy;
}

add_inst(ip) struct inst_list *ip;
{
    while (ip->next)ip=ip->next;
    ip->next=(struct inst_list *)malloc(sizeof(struct inst_list)); ip->next->next=0;
    strcpy(ip->apart.name, insname);
    ip->apart.place.x=placex; ip->apart.place.y=placey; ip->apart.rotation=rotation; rotation=0;
}

skipper()
{
    do {      ii++; netbuff[ii]=getchar(); }
    while (netbuff[ii]!=''); ungetc(' ',stdin); ii=0;
}

print_pad(instp,pinp,pfp) /* print pad information */
struct inst_list *instp; struct pinlist *pinp; struct pf_list *pfp;
{
    int xo, yo, xx, xy, lx, ly, alpha, orit, kk=1;
    struct pinlist *pip; struct pf_list *pp;
    pip=pinp; pp=pfp;
    printf("Pad_No\\Location(x,y)\\Shape_id\\Dimension\\n\\n");
    while(instp->next)
    { pip=pip;
      while(strcmp(instp->apart.name,pinp->apin.owner)!=0)pinp=pinp->next;
      while(strcmp(instp->apart.name,pinp->apin.owner)==0)
      { pfp=pp;
        while(pfp->padfile.type!=pinp->apin.type)pfp=pfp->next;
        { alpha=90*(instp->apart.rotation);
          xo=instp->apart.place.x; yo=instp->apart.place.y;
          lx=pinp->apin.locatn.x; ly=pinp->apin.locatn.y;
          switch(instp->apart.rotation)
          { case 0:  xx=xo+lx; xy=yo+ly; break;
            case 1:  xx=xo-ly; xy=yo+lx; break;
            case 2:  xx=xo-lx; xy=yo-ly; break;
            case 3:  xx=xo+ly; xy=yo-lx; break;
          } /*xx=xo+cos(alpha)*lx-sin(alpha)*ly;
             xy=yo+cos(alpha)*ly+sin(alpha)*lx;*/
        }
      }
    }
}

```

```

        orit=pfp->padfile.orient;
    if((instp->apart.rotation==1)||(instp->apart.rotation==3))
    { switch(pfp->padfile.orient)
        {case 0: orit=3;break;
         case 1: orit=2;break;
         case 2: orit=1;break;
         case 3: orit=0;break;
        }
    }
    printf("PAD%d\t\t%4d\t%4d\t\t %s\t\t%4d\n", kk++, xx, xy,
        pfp->padfile.shape, pfp->padfile.outdim.x);
    pinp=pinp->next;
}
instp=instp->next;
}
}

```

