TECHNICAL UNIVERSITY OF CRETE

DEPARTMENT OF ELECTRONIC & COMPUTER ENGINEERING

MULTI-CLASSIFIER APPROACHES FOR POST-PLACEMENT SURFACE - MOUNT DEVICES QUALITY ASSESSMENT

STEFANOS GOUMAS

Ph. D. THESIS

CHANIA 2008

Acknowledgments

First of all, I would like to thank Professor Michalis Zervakis, supervisor of my Ph.D. thesis, for his support, encouragement, valuable suggestions, novel ideas, and enthusiastic discussions. I was very fortunate to have an opportunity to work with him.

Many thanks are also due to Associate Professor Euripidis Petrakis, member of the advisory committee of my Ph.D. thesis, for valuable suggestions at the beginning of this research.

I would like to thank Professor George Stavrakakis, for serving as the advisory committee member of my thesis.

In particularly, I would like to thank Assistant Professor of the Department of Electrical & Computer Enginnering at Aristotle University of Thessaloniki and friend George Rovithakis, for his valuable help and contribution in my research effort.

On the personal side, I would like to thank Katerina, my wife, for making this all possible through her constant encouragement and support for years.

This research was supported by project HIPER (BRPR-CT98-0660) of the EC.

STEFANOS K. GOUMAS

Department of Electronic & Computer Enginnering
Technical University of Crete
July 2008

Abstract

This PhD thesis considers the problem of *Multi-lead Surface Mount Devices (SMD) Printed Circuit Boards (PCB) Post-Placement Quality Inspection* in the context of pattern recognition tasks. A Bayesian novel framework is proposed to visually inspect the placement quality of SMD, immediately after they have been placed in wet solder paste on a PCB. Our approach exploits the fact that individual leads encode the same information regarding relative positioning of the rigid body of the component on the pad area. Positioning measurements on each lead can be viewed as individual (inaccurate) measurements of the same quantity regarding the component displacement and rotation. Three measures of quality placement from individual lead images are of general interest, namely overlap, insulation distance and slump gap. More specifically, the quantification of positioning measures is viewed as a classification problem, where the lead displacement is inferred from characteristic features associated with image analysis for optical inspection. The features for classification are extracted from each segmented lead image and encode optical characteristics (i.e. *optical features*), by means of simple area measures that sustain the most desirable image attributes. Instead of concentrating in one and every (poorly imaged) lead, we fuse complementary information from all leads into a Bayesian estimation framework. The proposed estimation approach operates in two levels. The first level considers a crude computation of quantized displacement of each lead. This is done through classification. The second level operates in a Bayesian framework and aims to accurately model the estimation of component displacement based on quantized lead displacements. The developed methodology is tested with success on real industrial PCB images and has better performance than previous related methods reported in the literature of the PCB inspection field.

Motivated by the need for reducing time requirements and overcoming inaccuracies due to “microscopic” pixel-based consideration of individual lead images (such as segmentation process), we also study in this PhD thesis “macroscopic” techniques that do not consider pixel processing but rather define in an abstract way the characteristic features of individual lead images. More specifically, we consider one approach that only analyzes the edge structure of patterns in the image (i.e.

topological features) and a second approach that processes only the projection profile of patterns at a single relevant orientation (i.e. *projection features*). Both approaches use features that encode “reduced content” of the lead images. In this way we attempt to efficiently balance the amount of relevant information exploited and the computational load of the algorithm. Both methodologies are tested on real industrial PCB images. The quality of inspection slightly deteriorates while the computational time is significantly reduced, when compared to classical visual inspection techniques.

Finally, we also present in this PhD thesis a variety of multiple classifiers fusion strategies based on statistical and soft computing methods to improve the performance of the classification task on individual leads. To our knowledge this is the first time higher level (classifier) fusion is applied to the problem of quality inspection of SMD. Both fusion schemes, using *identical* and *distinct pattern representations* are considered. In the former case, we use only optical features for classification purposes. The latter scheme uses topological and projection features. We elaborate on two schemes for distinct pattern representations. In the former scheme we use only reduced dimensionality features (i.e., topological and projection features), whereas the latter enriches the topological and projection features with optical ones, in order to improve the classification rates and robustness across all lead-displacement classes. Comparing the classification results of the proposed combined classifiers, we can derive that all combiners have better performance than any primary classifier alone. Overall, classifier fusion can contribute to the visual quality inspection of SMD domain by improving accuracy and speed.

MULTI-CLASSIFIER APPROACHES FOR POST-PLACEMENT SURFACE - MOUNT DEVICES QUALITY ASSESSMENT

MULTI-CLASSIFIER APPROACHES FOR POST-PLACEMENT SURFACE - MOUNT DEVICES QUALITY ASSESSMENT

Stefanos Goumas

Ph. D. Thesis, Department of Electronic & Computer Engineering

Technical University of Crete, July 2008

MULTI-CLASSIFIER APPROACHES FOR POST-PLACEMENT SURFACE - MOUNT DEVICES QUALITY ASSESSMENT

Stefanos Goumas

A thesis submitted in partial fulfillment
of the requirements for the degree of
Doctor of Philosophy in Electronic & Computer Engineering

Doctoral Committee

Professor Michalis Zervakis, Thesis Supervisor
Professor George Stavrakakis, Member
Associate Professor Euripidis Petrakis, Member

Technical University of Crete
2008

ΜΕΘΟΔΟΙ ΠΟΛΛΑΠΛΩΝ ΤΑΞΙΝΟΜΗΤΩΝ ΓΙΑ ΤΗΝ ΕΚΤΙΜΗΣΗ ΠΟΙΟΤΗΤΑΣ ΤΟΠΟΘΕΤΗΣΗΣ ΟΛΟΚΛΗΡΩΜΕΝΩΝ ΚΥΚΛΩΜΑΤΩΝ ΣΕ ΠΛΑΚΕΤΕΣ ΤΥΠΩΜΕΝΩΝ ΚΥΚΛΩΜΑΤΩΝ

Στέφανος Γκούμας

Διατριβή για την μερική εκπλήρωση των απαιτήσεων
για το πτυχίο του Διδάκτορος στην Επιστήμη του Ηλεκτρονικού
Μηχανικού & Μηχανικού Υπολογιστών

Τριμελής Επιτροπή Διδακτορικού

Καθηγητής Μιχάλης Ζερβάκης, Επιβλέπων
Καθηγητής Γιώργος Σταυρακάκης, Μέλος
Αναπληρωτής Καθηγητής Ευριπίδης Πετράκης, Μέλος

Πολυτεχνείο Κρήτης
2008

Contents

Chapter 1

Introduction

1.1 Industrial Machine Vision Inspection Systems

Vision has long faschinated researches from disciplines such as psychology, neural science, computer science, and engineering. Machine vision can be described as an automatic deduction of structures or properties of the three-dimensional world from either single or multiple two-dimensional images of the world and recognition of objects with the help of these properties [1]. The images can be monochromatic or colored and can be captured from a single or multiple cameras. The structural properties we seek to deduce may not be only geometric properties, but also material properties. Geometric properties include the shape, size, and location of objects, whereas material properties include lightness or darkness of surfaces, their colors, and their structures. The purpose of a machine vision system is to infer the state of the physical world from noisy or ambiguous images of the world. Machine vision is difficult to realize because image formation is a many-to-one mapping. A variety of objects with different geometric and material properties can lead to identical images. Again, the obtained images may be noisy or distorted. Machine vision systems are complex and often are implemented with several modules. A modular approach makes it easier to control and monitor the performance of a system. Various stages or modules of a vision system can be implemented using conventional statistical methods, neural networks, fuzzy logic techniques, and genetic algorithms. Often the number of stages in a vision system and their complexity depend on the application for which the system is being designed. Applications of machine vision systems include *automation on the assemply line*, remote sensing, robotics, human computer communication, aids for the visually impaired, etc. Machine vision plays an important role of a large number of industrial processes [2]. In recent years, considerable efforts have been directed towards the development of *automated visual inspection systems*

[2], [3]. This is particularly true with specular surface inspection, because it is laborious for a human to perform.

Machine vision systems for industry first received serious attention in the mid-1970s. Throughout the early 1980s, the subject developed slowly, with a steady contribution being made by the academic research community, but with only limited industrial interest being shown. It seemed in the mid-1980s that there would be a major boost to progress, with serious interest being shown in vision systems by the major American automobile manufacturers. Then came a period of disillusionment in the USA, with a large number of small vision companies failing to survive. In the late 1980s and early 1990s, interest grew again, due largely to significant progress being made in making fast, dedicated image processing hardware. For many applications, it is possible now to provide sufficiently fast processing speed on a standard computing platform. Throughout the last 30 years, academic workers have demonstrated feasibility in a very wide range of products, representing all of the major branches of manufacturing industry.

Currently the main application areas for industrial vision systems occur in automated inspection and measurement and, to a lesser extent, robot vision. Automated visual inspection and measurement devices have, in the past, tended to develop in advance of robot vision systems [3]. In fact, quality control related applications, such as inspection, gauging, and recognition, currently account for well over half of the industrial machine vision market. This has been achieved, in many cases, by retrofitting inspection systems onto existing production lines. There is a large capital investment involved in developing a completely new robotic work cell. Moreover, the extra uncertainty and risks involved in integrating two relatively new and complex technologies makes the development of robot vision system seem a daunting task for many companies and development has lagged behind that of inspection devices. The technical difficulties involved in controlling flexible visually-guided robots have also limited the development. On the other hand, automated visual inspection systems now appear in every major industrial sector, including such areas as consumer goods, electronics, automobile, aerospace, food, mining, agriculture, pharmaceuticals, etc.

1.2 Inspection of Solder Joints on Printed Circuit Boards

In recent years the electronic and computer industry has grown rapidly. Due to commercial competition, the products have became lighter, smaller, and more precise. Integrated circuit (IC) is the crucial component in these products. Quality inspection plays a very important role in the IC industry for promoting high quality and high throughput. A popular and demanding real-time application is the automated visual inspection and classification of solder joints on Printed Circuit Boards (PCBs). Solder joint inspection [4], which has been a critical issue for quality control in the PCB assembly process, is a typical specular surface inspection task in machine vision. Imaging of a solder joint surface is a difficult task, since a solder joint forms a tiny, specular, curved and smooth surface. Reflections of the specular solder joint surface may appear, disappear or change their shapes abruptly, even with small changes in viewing direction. Furthermore, a distant point illumination cannot produce smooth shading on the specular surface, because light is reflected in a single direction.

Ever since ***surface-mounting technology devices*** (SMDs) for printed circuit board assemply processes has been developed, electrical products continuouasly tend toward the miniaturization of components, with denser packing its boards [5], [6]. With the increasing necessity for PCB product reliability, there has been a considerable demand for the development of an automatic visual inspection system.

A typical inspection system for the solder joints on PCB consists of a camera with appropriate illumination placed on top of the PCB conveyor system. Processing PCB images consists of two major stages: First, a pre-pocessing is performed in order to remove noise and make the tracking of solder joints on the image of the PCB easy. Then, the solder joints are classified according to the types of defects. The usual classification is concerned with the quantity of the solder paste placed on a joint. Four classes are defined, namely good, excess solder, insufficient and no solder. Simulation results on geometric models of joints have shown that efficient classification can be achieved only by an optimal feature selection, so that the classes do not overlap.

For the purpose of this thesis the problem of *solder joint inspection* is viewed as a pattern recognition problem.

1.3 The Pattern Recognition Problem

The term ***pattern recognition*** [7], [8], [9] encompasses a wide range of techniques for analyzing and interpreting complex data. The aim of pattern recognition is to find patterns in data which can be used to discriminate between subgroups of the data and to identify important distinguishing factors. Many computer-based pattern recognition applications are directed at finding ways of automating processes that humans do naturally, such as understanding language, or interpreting visual scenes. However pattern recognition techniques also provide a means of extracting relevant information from complex data that humans find difficult to interpret. In this case, the emphasis is on helping the human analysts rather than replacing them. Pattern recognition techniques are used in a variety of information processing problems of great practical significance, from speech recognition and the classification of handwritten characters, to machine vision inspection systems and medical diagnosis.

The problem of pattern recognition can be seen as of classifying a group of objects on the basis of certain subjective similarity measures. Those objects classified into the same ***pattern class*** usually have some common properties. The classification requirements are subjective, since different classification occurs under different properties of the features.

The design of an automatic pattern recognition system generally involves several major problem areas. The first one is concerned with the representation of input data which can be measured from the objects to be recognized. This is the ***sensing problem***. Each measured quantity describes a characteristic of the sample pattern or object. The measurements for each sample pattern can be arranged in the form a ***measurement vector*** or ***pattern vector***. The pattern vectors contain all the measured information available about the patterns. When the measurements yield information in the form of real numbers, it is often useful to think of a pattern vector as a point in an *N*-dimensional Euclidean space. The set of patterns belonging to the same class corresponds to an ensemble of points scattered within some region of the ***measurement space*** or ***pattern space***.

The second problem in pattern recognition concerns the extraction of characteristic features or attributes from the received input data and the reduction of the dimensionality of pattern vectors. This is often referred to as the ***preprocessing*** and ***feature extraction problem*** [10], [11], and results in a set of samples, known as ***feature vectors***, by mapping the measurement space into a ***feature space***. The extraction of features has been recognized as an important problem in the design of pattern recognition systems. If a complete set of discriminatory features for each pattern class can be determined from the measured data, the recognition and classification of patterns will present little difficulty. Automatic recognition may be reduced to a simple matching process. However, in most pattern recognition problems which arise in practice, the determination of a complete set of discriminatory features is extremely difficult, if not impossible.

The third problem in pattern recognition system design involves the determination of an optimum decision procedure, which is needed in the identification and classification process, namely the construction of a pattern classifier. The concept of ***pattern classification*** [9] may be expressed in terms of the partition of the feature space (a mapping from feature space to ***decision space***). The work of a pattern classifier is to assign each possible vector or point in the feature space to a proper ***pattern class***. In other words, pattern classification is the act of assigning a class label to an object. The assignment is always based on measurements that are obtained from that object.

Parameter estimation [9], [12], [13] is the process of attributing a parametric description to an object based on measurements that are obtained from that object. Parameter estimation and pattern classification are similar processes because they both aim to describe an object using measurements. However, in parameter estimation the description is in terms of a real-valued scalar or vector, whereas in classification the discription is in terms of just one class selected from a finite number of classes.

The discipline of pattern recognition has seen enormous progress since its beginnings more than four decades ago. Over the years, various approaches have emergend, based on ***statistical decision theory*** [9], structural matching and parsing, ***soft computing*** (i.e. *neural networks*, *fuzzy logic*, and *evolutionary computing* and *genetic algorithms*) [14], [15], artificial intelligence, and others. Obviously, these approaches are characterized by a high degree of diversity. In order to combine their strenghts and avoid their weaknesses, ***hybrid pattern recognition systems*** [15], [16] have been proposed, combining several techniques into a single pattern recognition system. Hybrid methods have been known about for a long time, but they have gained new interest only recently. Examples of hybrid pattern recognition systems are *neuro-fuzzy* and *fuzzy-neural classifiers* [17], [18], evolutionary computing for neural network architecture and parameters optimization, neural networks for structural pattern recognition, combining neural networks and hidden Markov models for statistical pattern recognition [16], ***combined*** (or ***simultaneous***) ***classification-estimation systems*** [12], [19], ***multiple classifier systems*** [20], [21] and others.

1.4 Image Processing and Analysis, Approximate Processing and Reduced Dimensionality

Visual inspection techniques require extensive image processing and image analysis for improving the image quality and deriving characteristic features. ***Image processing*** [22], [23] involves changing the nature of an image in order to either improve the pictorial information for human interpretation, or render it more suitable for autonomus machine perception. Image processing is usually performed within rectangles, circles or along lines and arcs. Image prossecing operations include geometric and radiometric correction, enhancement, restoration, filtering (e.g., smoothing, sharpening), ***segmentation*** (e.g., thresholding, edge detection, Hough transform), morphological operations, etc. Such operations can be used to improve image quality (e.g., remove noise, improve contrast) and to enhance or separate certain image features (e.g., regions, edges) from the background [22], [24]. Image processing operations transform an input image to another image having the desired characteristics.

One of the most difficult and important problems in automating machine vision is to understand what kind of information is required and how is translated into measurements or *features* extracted from images. A descriptive set of uncorrelated features can drastically boost the classification success rate. ***Image analysis*** [22] – [28] transforms images to measurements. In particular, image analysis is related to the extraction and measurement of certain image features (e.g., lines, and corners) and transforms these image features to numbers, vectors, character strings etc. The ultimate goal of image analysis is geared towards the extraction of features that can be used by classifiers to classify objects.

The limitation of computer-based tools related to computer time and working space poses a high priority on the objective choice of a limited number of essential characteristics (state-space or feature-space reduction) but also on the exclusion of redundant observations (sample-space or data-space reduction). Thus, the concept of ***approximate processing*** [29] has been considered in real-time applications, where there is a necessity for approximating a given algorithm with another that has reduced computational cost.

A wide-spread approach related to approximate processing deals with ***feature-space reduction*** [9], [30] and attempts to preserve the most important information conveyed by features extracted from the input data, while simplifying the required computations by reducing the dimensionality of the feature space. Principal components analysis (PCA) [9], [31], [32] is a well-established feature-space reduction technique employed in different forms including Factor Analysis [31], Karhunen-Loeve Transform (KLT) [7], [9], [11], [32], and Hotelling Transform [31], [32], depending on the application.

Another approach to information reduction, referred to as ***data-space reduction*** [30], exploits the fact that the underlying dimensionality of the data (intrinsic dimensionality) may be small, even though the input dimensionality is quite large expressing high correlation among input data. Unsupervised linear-mapping approaches in the form of PCA and orthogonal subspace projections are designed to decorrelate the data and maximize the information content in a reduced dimensionality space.

1.5 Statistical and Soft Computing Approaches for Pattern Recognition

Statistical pattern recognition is a relatively mature discipline and a number of commercial recognition systems have been designed based on this approach. In the ***statistical approach*** [9], [33] – [35], a set of features is extracted from the input pattern, and the classification is carried out by partitioning the feature space. The most general and most natural framework to formulate solutions to pattern recognition problems is a statistical one, which recognizes the *probabilistic nature* of both the information we seek to process and the form in which we express results. In the statistical approach, we are not concerned with whether the classifier *actually* makes a wrong decision, but we are concerned with the *probability* of a wrong decision.

Soft computing (SC), which has emerged in the last decade as an efficient tool, consists of several computing paradigms, including fuzzy logic, neural networks, and genetic algorithms. Its aim is to exploit the tolerance for imprecision, uncertainty, approximate reasoning, and partial truth in order to achieve tractability, robustness, low cost solution and close resemblance with human like decision-making. Traditionally, pattern recognition problems have been solved by using classical statistical methods and models, which lack, in some cases, the accuracy and efficiency needed in real-world applications. Traditional methods include the use of statistical models and simple information systems. We instead, consider more general modeling methods, which include fuzzy logic and neural networks. We also use genetic algorithms for the optimization of the fuzzy systems and neural networks. Combining SC methodologies, we can build a powerful hybrid intelligent system that will solve efficiently and accurately a specific pattern recognition problem.

Fuzzy logic [36] – [38], [41] is an area of soft computing that enables a computer system to reason with uncertainty. A fuzzy inference system consists of a set of if-then rules defined over fuzzy sets. Fuzzy sets generalize the concept of a traditional set (i.e., crisp set) by allowing the membership degree to be any value between 0 and 1. This corresponds, in the real world, to many situations where it is difficult to decide in an unambiguous manner if something belongs or not to a specific class. In other words the fuzzy set concept provides us with an intuitive method of representing one form of uncertainty by eliminating the sharp boundary that divides members of the class from nonmembers. However, in some decision-making situations measurements of length, area, and weight, classes are defined with sharp boundaries. Since the evidence for measurement error is unavoidable in most measurements, some uncertainty usually prevails. To represent this kind of uncertainty, known as ambiguity, we assign a value in the unit interval [0,1] to each possible crisp set to which the element in question might belong. This value represents the degree of evidence or belief or certainty of the elements's membership in the set. Such a representation of uncertainty is known as a ***fuzzy measure*** [39] – [41].

Neural networks [8], [10], [17], [18], [38], [41] – [43] are computational models with learning (or adaptive) characteristics that model the human brain. Generally speaking, biological natural neural networks consist of neurons and connections between them, and this is modeled by a graph with nodes and arcs to form the computational neural network. This graph along with a computational algorithm to specify the learning capabilities of the system is what makes the neural network a powerful methodology to simulate intelligent or expert behavior.

Genetic algorithms and ***evolutionary methods*** [44] – [48] are optimization methodologies based on principles of nature. Both methodologies can also be viewed as searching algorithms because they explore a space using heuristics inspired by nature. Genetic algorithms are based on ideas of evolution and the biological process that occur at the DNA level. Basically, a genetic algorithm uses a population of individuals, which are modified by using genetic operators in such a way as to eventually obtain the fittest individual. Any optimization problem has to be represented by using chromosomes, which are a cidified representation of the real values of the variables in the problem. Both genetic algorithms and evolutionary methods can be used to optimize a general objective function. As genetic algorithms are based on the ideas of natural evolution, we can use this methodology to evolve a neural network or a fuzzy system for a particular application.

1.6 Combination of multiple classifiers

The use of ***multiple classifiers*** [20] – [21], [48] – [50] has gained momentum in the recent years and researchers have continuously argued of the benefits of using multiple classifiers to solve complex pattern classification problems. The idea appeared under many names: *decision combination*, *multiple experts*, *mixture of experts*, *classifier ensembles*, *opinion pool*, *classifier fusion*, and more.

The combination of multiple classifiers has been intensively studied with the aim of overcoming the limitations of primary classifiers. Classifiers differing in feature representation, architecture, learning algorithm, or training data exhibit complementary classification behavior and the fusion of their decisions can yield higher performance than the best individual classifier. The performance of a multiple classifier system relies on both the complementariness of the participating classifiers and the combination method.

Multiple classifier systems can be classified in a variety of ways. The basic categorization of multiple classifier systems has been by the method the classifiers are arranged. The two basic categories in this regard are the *serial suite* and the *parallel* *suite*. The parallel expert architecture consists of a set of classifiers that are consulted in parallel. The decisions of the various experts are combined in parallel by the fusion module. The experts, in this case, are capable of independent and simultaneous operation. On the other hand, the serial suite consists of a set of classifiers arranged in series, or in tandem. This architecture is well suited to deal with situations where the different experts have a ternary decision scheme. A scheme in which they can be undecided on the input pattern they are presented with. If the current expert is undecided, information is passed to the next expert in the sequence.

Multiple classifiers can also be categorized based on the method of mapping between the input and output of the fusion module. This mapping may be *linear* or *non-linear*. Linear combination is the simplest approach, in which a weighting factor is assigned to the output of each expert being combined. Weighted average, fuzzy integrals are among the linear combination methods, while the majority voting is a non-linear method.

Combining methods can also be divided into two different classes depending on the *pattern representation* methodology. The primary classifiers can all use the same representation, and hence the classifiers themeselves should be different. In ***distinct pattern representations*** approach [49], [51] the individual classifiers use different representasions of the same inputs. This can be due to the use of different sensors or different features extracted from the same data set.

Another categorization of classifier combining methods are if they encourage specialization in certain areas of the feature space (i.e *classifier selection*). On the other hand, ensemble of classifiers have primary classifiers that do not encourage such specialization and classifiers themselves must have different classification powers (i.e. *classifier fusion*). In other words, in an ensemble each base classifier can be used alone to provide a solution to the input pattern. While a modular approach would need the coordination of all the classifiers to present a complete solution.

Overall, the combination methods can be categorized according to the level of the individual classifiers outputs: *abstract level* (class label), *rank level* (rank order), and *measurement level* (class scores). The abstract level classifiers output only the class label, whereas the rank level classifiers output the rank for each class. The measurement level classifiers assign each class a measurement value to indicate the possibility that the input pattern pertains to the class.

1.7 Objective and approaches

The objective of this PhD thesis is to develop a variety of approaches to build hybrid intelligent systems for ***multi-lead surface mounted devices (SMDs) post-placement quality inspection*** using classical statistical and soft computing pattern recognition methodologies.

In the first approach a Bayesian novel framework is proposed to inspect the placement quality of Surface Mount technology Devices (SMDs), immediately after they have been placed in wet solder paste on a Printed Circuit Board (PCB) [19], [30]. The developed approach involves the indirect measurement of each individual lead displacement with respect to its ideal position, centralized on its pad region. This displacement is inferred from area measurements (***geometric*** or ***optical features***) on the raw image data of the lead region through a classification process. To increase the accuracy in the computation of the lead displacement, we introduce a ***combined classification / estimation process*** [19], [30] in which the individual lead displacement classifications are viewed as measurements (or observations) of the same physical quantity i.e., the displacement of the entire component as a rigid body. Certain geometric relations connecting lead shifts to component displacement are also derived. Employing these relations we can infer a new refined measurement of the shift of each individual lead, a quantity crucial to the calculation of the quality measures.

Motivated by the capabilities of approximate processing and the need for reducing time requirements and overcoming inaccuracies due to “microscopic” pixel-based consideration of images (such as segmentation process), we study in the second approach “macroscopic” techniques that do not consider pixel processing but rather define in an abstract way the characteristic features of individual lead images. More specifically, we adopt two different forms of ***data-space reduction*** [28] directly on the initial image space, affecting: 1) the intensity levels or dynamic range, by transforming the gray-scale image into a binary edge image (referred to as ***reduced dynamic-range processing***) [28], extracting ***topological features***, and 2) the number of independent variables, by utilizing only specific ***image projections*** of the image data (referred to as ***reduced input-dimension processing***) [28]. The classification task of individual leads is executed via two different classifiers, which based on the abovementioned reduced-dimensionality features. The first classifier is a Hamming neural network classifier based on reduced dynamic-range processing. The second classifier is a Bayesian distance classifier based on input – dimension processing.

The goal of the third approach is to test and compare multiple classifier fusion methods for improving the classification of the individual leads in component quality inspection [48]. Instead of using single statistical or neural classifiers as in first approach, we implement multi-modular classification systems that combine decisions from statistical and neural modules. Combining the power of the primary classifiers through multimodular architectures we improve the classification results and enhance the robustness of the overall classification system. We propose four representative schemes for soft fusion of multiple classifiers based on identical pattern representations.

The scope of the last approach is to fuse decisions from primary classifiers, which operate on distinct pattern representations [48]. The motivation for exploring the combination issue is to improve performance of classification task of individual leads based on reduced dimensionality distinct pattern representations [28], [48]. We elaborate on two schemes for distinct pattern representations. In the former scheme we use only reduced dimensionality features (i.e., topological and projection features), whereas the latter enriches the topological and projection features with optical ones, in order to improve the classification rates and robustness across all lead-displacement classes.

1.8 Contribution

This PhD thesis considers the problem of *Multi-lead Surface Mount Devices (SMD) Printed Circuit Boards (PCB) Post-Placement Quality Inspection* in the context of pattern recognition tasks. More specifically, the present work introduces:

- Novel issues in relation to image processing and analysis techniques for the individual lead images of components.
- A Bayesian novel framework for component displacement estimation from individual lead displacements.
- New concepts for data-space reduction for individual lead images:
    - Reduced Dynamic-Range Processing based on topological features,

  - Reduced Input-Dimension Processing based on projection features
- The concept of Multiple Karhunen Loéve Transformation for feature-space reduction.
- The application of ensemble classifiers for improving the classification of the individual lead displacements using two different scenarios:
  - Identical pattern representations
  - Distinct pattern representations

1.9 Publications

Parts of the work presented in this thesis have been already submitted for publication or published in international scientific journals and conference proceedings as follows:

Journals

- M. Zervakis, S.K. Goumas, and G. A. Rovithakis, "A Bayesian Framework for Multi-lead SMD Post-Placement Quality Inspection", *IEEE Transactions on Systems, Man and Cybernetics* -Part B: Cybernetics, vol. 34, no. 1, pp. 440-453, February 2004.
- S. K. Goumas, I. N. Dimou, M. E. Zervakis, "Combination of Multiple Classifiers for Post Placement Quality Inspection of Components: A Comparative Study", *Information Fusion*, Accepted for publication in the March 1, 2008.

Conferences

- S. K. Goumas, G. A. Rovithakis and M. E. Zervakis, "A Bayesian Image Analysis Framework for Post Placement Quality Inspection of Components", *Proceedings of 2002 IEEE International Conference on Image Processing - ICIP 2002*, pp. II-549-552, September 22-25, 2002, Rochester, New York, USA.
- S. K. Goumas, M.E. Zervakis, and G. A. Rovithakis, "Reduced Dimensionality Space for Post Placement Quality Inspection of Components based on Neural Networks", in *Proc. ESSAN' 2004, 12th European*

Symposium on Artificial Neural Networks, pp. 275-280, April 28-30, 2004, Bruges, Belgium.

1.10 Thesis overview

The overview of the thesis is as follows. In the next Chapter the fundamentals concepts of *industrial machine vision inspection systems* (IMVIS) are given. The structure, design issues, components and applications of these systems are presented in enough detail to give necessary background for the particular problem of SMD PCB Post Placement Quality Inspection. The aim of the 3rd Chapter is the formulation of the SMD PCB Post Placement Quality Inspection problem. After a short review of surface mount process and automatic visual inspection of PCBs, the problem formulation and requirements of industrial manufacturer for SMD post placement quality inspection are presented. Finally, the state of the art in SMD defects is referred. In Chapter 4, the algorithmic concepts of pattern recognition and image analysis are described. In the first part of this chapter, the general pattern recognition problem is formulated. Then, a diverse of pattern recognition approaches, such as classical statistical and soft computing ones along with parameter estimation algorithms are presented and explained in enough detail to give necessary theoretical background for the particular problems that will need to be addressed. The second part of 4th Chapter concentrates on the image analysis techniques, such as adaptive thresholding algorithms that were used for pixel-based feature extraction from individual lead images of the components. In Chaptrer 5, a Bayesian novel framework is proposed to inspect the placement quality of SMDs, immediately after they have been placed in wet solder paste on a PCB. We propose in the 6th chapter two neural networks based approaches to extract the characteristic features (reduced-dimensionality features) from individual lead images. The goal of the 7th Chapter is to test and compare multiple classifier fusion methods, operating on identical pattern representations, for improving the classification of the individual leads in component quality inspection. The scope of the 8th Chapter is to fuse decisions from primary classifiers, which operate on distinct pattern representations, for improving the performance of classification task of individual leads of the components. Finally we conclude in Chapter 9 with some discussion on our future research.

Chapter 2

Fundamental Concepts of Industrial Machine Vision Inspection Systems

2.1 Fundamentals of Machine Vision Systems

Machine (or ***computer***) ***vision*** is computer imaging where the application does not involve a human being in the vision loop. In other words, the images are examined and acted upon by a computer. Although people are involved in the developed of the system, the final application requires a computer to use the visual information directly [58].

A ***machine vision system*** (MVS) is the technological integration of a camera and a computer. In a MVS, the camera does the task of an eye and the computer acts as the brain by processing the information perceived by the camera. Signals generated by the camera are stored in the computer as a digital image. Image processing and analysis algorithms are used to extract a set of features, called a ***pattern***, from the image to represent an object. On the basis of the pattern, the object can then be classified into one of the several pre-defined classes using a classification algorithm, called a ***pattern classifier***.

A machine vision system is concerned basically with the deduction of surfaces and properties of three-dimensional objects from their two-dimensional images. It involves edge detection, segmentation, and extracting features using texture, shading, stereo, motion, and recognition [1], [3]. A machine vision system, like other engineering systems, has two components – hardware and software – and consists of stages such as image acquisition, preprocessing, feature extraction, storing objects by association, accessing a knowledge base, and recognition. The software component of the vision system deals with algorithms and the implementation of these stages. A variety of tools, such as conventional statistical methods, neural networks, fuzzy logic techniques, genetic algorithms, or hybrid techniques (such as fuzzy-neural or neuro-fuzzy), can be used in implementing various stages of a machine vision system. The hardware component of a machine vision system deals with imaging devices such as digitizers, scanners, cameras, and display devices, as well as film recorders, storage devices, and a computer. To process images on a digital computer requires that the images first be digitized. Digital images are often obtained with digital cameras.

Humans can easily recognize objects and understand complex scenes with multiple objects, noise, clutter, occlusion, and camouflage. Humans are able to recognize as many as 10,000 distinct objects [63] under varying viewing conditions, while a state-of-the-art machine vision system can recognize relatively few objects. One approach to implementing a machine vision system is to emulate the ***human vision system*** [1], [59]. However, the problem with this approach is that the human vision system is very complex and is not well understood. The human vision system beyond the human eye is disjointed and speculative. Therefore, at this time it is impossible to emulate the human vision system exactly; however, the study of biological systems provides us with clues for developing machine vision systems. Research suggests that the advantages of biological vision over current machine vision are from feedback, flexible control, and the kinds of feature detectors. In [59] an example general purpose machine vision system having some of these biological characteristics is described. The purpose of this system is to find and identity spatial features of luminance in the field of view. The design method is to model the human vision system. The functions of the modules approximate those of the human brain. For convenience, implementation in a test bed uses a mixture of Neural Networks and standard processing algorithms. The system recognizes gray-level images in the field of view, with arbitrary translations and rotations. It does not emulate certain biological characteristics. Not emulated are binocularity, size invariance, motion perception, color sensitivity, and discernment of visual boundaries. Indeed, many applications can omit these properties.

Machine vision is concerned with both low-level and high-level processing issues, such as cognitive issues. Stages in a typical MVS are shown in block diagram in figure 2.1. The first three stages – ***image acquisition***, ***preprocessing***, and ***feature extraction*** – implement early processing or ***low-level processing***, whereas the last stage – ***recognition*** that includes a knowledge base and associative storage – deal with cognitive processing or ***high – level processing***. Low-level processing deals with the retina, while high – level processing deals with the cognitive use of knowledge.

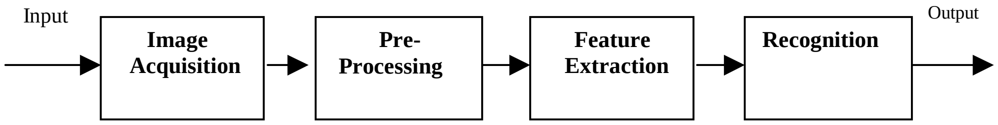


Figure 2.1 Processing stages in a typical machine vision system

The traditional machine vision approach rests upon three basic tenets [61]

1. The goal of a MVS is to create a detailed model and full representation of the visual world.
2. The MVS is hierarchical, with each stage being responsible for performing a specific task until finally only unique features are left than can be acted upon by the later stages of processing.
3. There is a dependency of the higher levels of visual processing on the lower levels, but in general the reverse is not true.

MVS may be applied to a variety of areas having considerable significant commercial importance and academic interest such as:

- Industrial applications (robot vision, automated visual inspection)
- Aerial/satellite image analysis
- Agriculture
- Document processing
- Forensic science, including fingerprint recognition
- Health screening
- Medicine
- Military applications (target identification, missile guidance, vehicle location)
- Publiching
- Research, particularly in physics, biology, astronomy, materials engineering, etc.
- Security and surveillance
- Road traffic control

For the purposes of this PhD thesis we will restrict our discussion to industrial applications of MVS and in particularly to *automated inspection of solder joints on surface mount printed circuit boards*.

2.2 Industrial Machine Vision Inspection Systems

Over the past 30 years, machine vision has taken a vital role in controlled industrial applications. ***Industrial machine vision*** is the use of computer processing of images that arise from manufacturing processes [111]. As such, industrial machine vision is a sub-field of the larger discipline of computer vision, which tackles the problem of computerized image interpretation. Industrial machine vision systems (IMVS) operate in constrained and controlled environments: a fact that can be exploited by algorithms used by the system. However the demands on these systems are high in terms of performance and cost. To meet these demands, industrial machine vision systems have successfully integrated various vision modules, such as optics, image acquisition and image analysis.

Industrial machine vision can be divided into two areas: automated assemply by robots, i.e. ***robot vision***, and ***automated visual inspection*** (AVI) [53], [60] or ***industrial machine vision inspection***. Robot vision is typically concerned with sensing, interpreting, and reasoning about a three dimensional scene so as to make decisions about it. Applications include driving an autonomously guided vehicle, a welding robot or clearing submarins. Applications such as gauging and determining the presence or absence of parts are generally termed AVI. The difficulty of trying to measure a feature accurately is offset by the ability to have more control over the environment in such areas as lighting, and part fixturing. Advances in computer technology, sensing devices, image processing, and pattern recognition have resulted in better and cheaper industrial visual inspection equipment. So, AVI systems has become more widely used in many fields such as delicate electronics component manufacturing, quality textile production, metal product finishing, glass manufacturing, machine parts, printing products and granite quality inspection, integrated circuits manufacturing and many others [2].

The automated visual inspection process involves observing the same type of object repeatedly to detect anomalies [54], [55]. The process starts with *imaging* the object to be inspected by a sensor (or sensors) from which visual data are collected and sent to the processor for analysis. Features representing the object are then extracted and matched to a predefined *model*. This ***feature-to-model*** matching process (or ***model-based***) is the most common technique for ***detecting defects*** [54]. It is realized by finding features in the given object that match the model's definition of defects, or by verifying all extraced features in the given object to be normal, expected features as those defined by the model. Afer the detection, decisions are made to classify the object into a *defect type*.

There are two main areas of application for AVI: as a means of ensuring quality control by rejecting defective products; and as a means of gathering statistical information to provide feedback to the manufacturing process. The former case is often referred to as a feedforward solution, and the latter case is often referred to as a feedback solution [53].

The advantages of AVI stem from the replacement of lasbour with capital, the automation of the manufacturing process, enhancing consistency of production, removing the need to work in hazardous environments, while producing quantitative measurements and facilitating the integration with other aspects of automated manufacturing.

There are, however, several disadvantages to automatic visual inspection. Commercial systems are quite expensive. For most applications, off-the-shelf systems do not exist and special packages must be designed or modified, further increasing the total cost. Special fixturings may be needed for loading and unloading objects and special lighting must be used. More skilled technician and engineering staff may be needed to operate and maintain the systems; integration will probably require more sophisticated software and will increase the complexity of current systems. Despite this, the applications of automated visual inspection are legion [56] and there is a definite trend toward the deployment of AVI systems as a special-purpose real time machine vision systems with control of their own environment.

An AVI system usually requires real-time operation to enable the inspection process to keep up with the manufacturing process. This is the most challenging task when devoloping an AVI system in which products (objects) are transported at high speed. One the most critical timing measurements is the cycle time between the presentations of successive images requiring analysis [53]. The image processing and analysis must take less than the image acquisition cycle time in order to cope with the requirement of the ral-time operation. Therefore it is vital to develop ***computationally efficient vision algorithms*** (such as ***adaptive multilevel thresholding***) [57].

For real-time AVI, there are three important advantages that are used to drive the inspection system design [53]:

- The products under inspection might well be simpler than those found in natural scenes, and it may be possible to use computationally efficient algorithms to inspect them.
- The system will probably know what it is looking for, hence it is possible to create a stored model of the object under inspection. ***Model-based inspection*** introduces the possibility of greater efficiency and robustness.
- The system developer will usually have complete control over the lighting environment, and some inventive lighting schemes can be employed.

2.3 Structure, Requirements and Design of Industrial Machine Vision Inspection Systems

Industrial machine vision inspection systems (IMVIS) are replacing the process of manual inspection of products in different industries. Inspection may include defect detection, dimensional measurement, orientation detection, grading, sorting and counting.

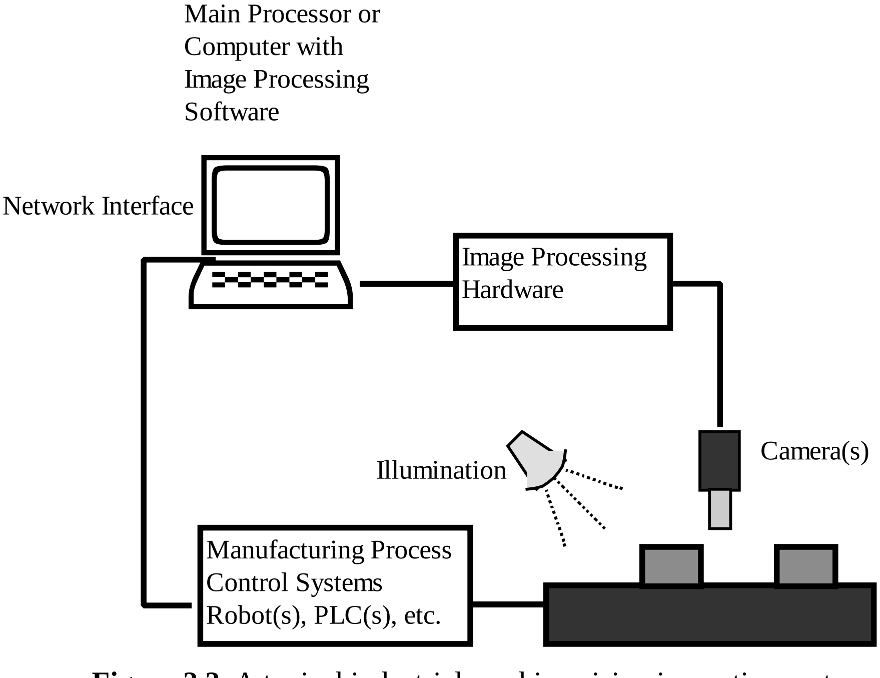


Figure 2.2 A typical industrial machine vision inspection system.

Figure 2.2 illustrates the structure of a typical industrial vision system [2]. First, a computer is employed for processing the acquired images. This is achieved by applying special purpose image processing analysis and classification software. Images are usually acquired by one or more cameras placed at the scene under inspection. The positions of the cameras are usually fixed. In most cases, industrial automation systems are designed to inspect only known objects at fixed positions. The scene is appropriately illuminated and arranged in order to facilitate the reception of the image features necessary for processing and classification. These features are also known in advance. When the process is highly time-constrained or computationally intensive and exceeds the processing capabilities of the main processor, application specific hardware (e.g., DSPs, ASICs, or FPGAs) is employed to alleviate the problem of processing speed. The results of this processing can be used to:

- Control a manufacturing process (e.g., for guiding robot arms placing components on printed circuits, painting surfaces etc.).
- Propagated to other external devices (e.g., through a network or other type of interface like FireWire) for further processing (e.g., classification).
- Characterize defects of faulty items and take actions for reporting and correcting these faults and replacing or removing defective parts from the production line.

The requirements for the design and development of a successful industrial machine vision system vary depending on the application domain and are related to the tasks to be accomplished, environment, speed etc. For example, in machine vision inspection applications, the system must be able to differentiate between acceptable and unacceptable variations or defects in products, while in other applications, the system must enable users to solve guidance and alignment tasks or, measurement and assembly verification tasks.

There exists no industrial vision system capable of handling all tasks in every application field. Only once the requirements of a particular application domain are specified, then appropriate decisions for the design and development of the application can be taken. The first problem to solve in automating a machine vision task is to understand what kind of information the machine vision system is to retrieve and how this is translated into measurements or features extracted from images. For example, it is important to specify in advance what "*defective*" means in terms of measurements and rules and implement these tasks in software or hardware. Then, a decision has to be made on the kind of measurements to be acquired (e.g., position or intensity measurements) and on the exact location for obtaining the measurements.

For the system to be reliable, it must reduce "*escape rates*" (i.e., non-accepted cases reported as accepted) and "*false alarms*" (i.e., accepted cases reported as non-accepted) as much as possible. It is a responsibility of the processing and classification units to maintain system reliability, but the effectiveness of classification depends also on the quality of the acquired images. An industrial vision system must also be robust. Thus, it should adapt itself automatically and achieve consistently high performance despite irregularities in illumination, marking or background conditions and, accommodate uncertainties in angles, positions etc. Robust performance is difficult to achieve. High recognition and classification rates are obtained only under certain conditions of good lighting and low noise. Finally, an industrial vision system must be fast and cost efficient.

As we have emphasized above, the important attributes of an industrial machine vision inspection system are flexibility, efficiency in performance, speed and cost, reliability and robustness. In order to design a system that maintains these attributes it is important to clearly define its required outputs and the available inputs. Typically, an industrial inspection system computes information from raw images according to the following sequence of steps [2], [54] :

1. ***Image acquisition:*** The first step in using a machine vision system is to acquire a digital image. This can be achieved by either using a digital camera or a sensor and a digitizer. A well-designed imaging system would reduce noise, prevent blur, stop object motion, optimize contrast between parts of interest and the background, resolve the desired defect size, and emphasize those features which are relevant to the inspection. The primary objectives are to acquire a quality representation of the object under inspection and, more importantly, to greatly reduce the complexity of subsequent image processing. Image acquisition involves the design of illumination and optics, and the choice of sensors and their placement [54].

2. ***Image processing:*** Once images have been acquired, they are filtered to remove background noise or unwanted reflections from the illumination system. Image restoration may also be applied to improve image quality by correcting geometric distortions introduced by the acquisition system (e.g., the camera).
3. ***Feature extraction:*** A set of known features, characteristic for the application domain, is computed, probably with some consideration for non-overlapping or uncorrelated features, [62] so that better classification can be achieved. Examples of such features include size, position, contour measurement via edge detection and linking, as well as and texture measurements on regions. Such features can be computed and analyzed by statistical or other computing techniques (e.g. neural networks or fuzzy systems). The set of computed features forms the description of the input image.
4. ***Decision-making:*** Combining the feature variables into a smaller set of new feature variables reduces the number of features. While the number of initial features may be large, the underlying dimensionality of the data, or the intrinsic dimensionality, may be quite small. The first step in decision making attempts to reduce the dimensionality of the feature space to the intrinsic dimensionality of the problem. The reduced feature set is processed further as to reach a decision. This decision, as well as the types of features and measurements (the image descriptions) computed, depends on the application. For example, in the case of visual inspection during production the system decides if the produced parts meet some quality standards by matching a computed description with some known model of the image (region or object) to be recognized. The decision (e.g., model matching) may involve processing with thresholds, statistical or soft classification.

At the last level of decision-making and model matching (i.e. model-based) mentioned above, there are two types of image (region or object) models that can be used namely, declarative and procedural [2], [55]. ***Declarative models*** consist of constraints on the properties of pixels, objects or regions and on their relationships. ***Procedural models*** are implicitly defined in terms of processes that recognize the images. Both types of models can be fuzzy or probabilistic, involving probabilistic constraints and probabilistic control of syntactic rules respectively. A special category of models is based on neural networks.

Model-based approaches often require that descriptions (e.g., features) of the image at different levels of specificity or detail be matched with one of possible many models of different classes of images. This task can become very difficult and computationally intensive if the models are complex and a large number of models must be considered. In a top-down approach to model matching, a model might guide the generation of appropriate image descriptions rather than first generating the description and then attempting to match it with a model. Another alternative would be to combine top-down and bottom-up processes. The above control strategies are simplified when one is dealing with two-dimensional images taken under controlled conditions of good lighting and low noise, as it is usually the case in industrial vision applications. Image descriptions and class models are easier to construct in this case and complex model matching can be avoided. Model-based approaches to industrial visual inspection tasks [64] have been applied in a variety of application fields and many of them are reviewed in the following sections.

2.4 Applications of Industrial Machine Vision Inspection Systems

Interesting surveys specializing in a single application field include among others Ref. [65] for automatic PCB inspection, Ref. [66] for wood quality inspection, and Ref. [67] for automatic fruit harvesting. Other important general reviews that cover all the fields of visual inspection have been published in Ref. [68], whereas model-based approaches to visual inspection are considered in [54] and [69] and more recently in [55], [70] and [71]. In Ref. [55], a classification of automated visual inspection applications is presented based on the type of images to be processed. Binary, gray-scale, color, and range image systems are considered, each one showing certain characteristics in the context of the particular application field being used. In Ref. [70] and [71] on the other hand, machine vision systems are classified according to the qualitative characteristics of the objects or processes under inspection. Three classes are presented, namely dimensional verification, surface detection, and inspection of completeness.

Two independent ways of classifying industrial vision applications are proposed in [2]. First, industrial vision applications are classified according to the inspected features of the industrial product of process in four categories, namely: (a) *Dimensional quality*, (b) *Structural quality*, (c) *Surface quality* and (d) *Operational quality*. Industrial vision applications are also classified in terms of flexibility according to the so-called "*Degrees of Freedom*" (DoFs) that form inspection independent features. This classification enables the evaluation of tools intended towards similar industrial vision applications.

2.5 Components of an Industrial Machine Vision Inspection System

Developing a machine vision system that is useful to industry in practice requires a multidischiplinary approach, encompassing aspects of all of the following technologies:

- Spatial sampling
- Illumination (or lighting)
- Imaging Optics (or Camera)
- Image sensors
- Analog signal processing
- Digital information processing
- Digital systems architecture
- Software
- Interfacing vision systems to other machines
- Networking
- Interfacing visual systems to humans
- Existing industrial work and Quality Assurance practice

System development involves integration of software and hardware tools into a complete application. Today's industrial machine vision systems are offering far easier integration of various components originating from various software and hardware vendors. Even conventional programming environments such as C, C++, MATLAB, etc., allow for software components to be embedded into a single system.

2.5.1 Imaging Optics and Illumination Techniques

As for many natural visual systems the process of image formation in industrial machine vision inspection begins with the light rays which enter the camera through an *angular apperture*, and hit a screen or *image plane*, the camera photosensitive device which registers light intensities. Notice that most of these rays are the result of the reflections of the rays emitted by the light sources and hitting object surfaces. There are a variety of physical parameters playing a essential role in image formation such as [74]:

1. Optical parameters of the lens characterize the sensor's optics. They include:
   - lens type,
   - focal length,
   - field of view,
   - angular apertures,
2. Photometric parameters appear in models of the light energy reaching the sensor after being reflected from the objects in the scene. Yhey include:
   - type, intensity, and direction of illumination,
   - reflectance properties of the inspected surfaces,
   - effects of the sensor's structure on the amount of light reaching the photoreceptors.
3. Geometric parameters determine the image position on which a 3-D point is projected. They include:
   - type of projections,
   - position and orientation of camera in space,
   - perspective distortions introduced by the imaging process.

All the above plays an essential role in any intensity imaging device, be it a photographic camera, camcorder, or computer-based system. However, further parameters are needed to characterize digital images and their acquisition systems such as:

- the physical properties of the photosensive matrix of the viewing camera,
- the discrete nature of the photoreceptors,
- the quantization of the intensity scale.

An ***optical system*** can be regard as a device that aims at producing the same image obtained by a traditional camera's *pinhole aperture*, but by means of a much larger apperture, under a wide range illumination conditions and exposure times (the exposure time being controlled by a *shutter*). Modern optical systems are quite sophisticated, composed of lenses, apertures, and other elements, explicity designed to make all rays coming from the same 3-D point converge onto a single image point [74].

To acquire a good image, proper illumination is a basic necessity. Improper illumination can cause glare or non-uniform light variation over the field of view. This can lead to distortion of object features in the image. Determination of an ideal illumination source is not easy and depends on the nature of the visual inspection task.

In order to setup an appropriate *illumination system* we have to consider the radiometric properties of the illumination sources, such as spectral characteristics, intensity distribution, radiant efficiency and luminous efficacy. For practical applications we also have to carefully choose electrical properties, temporal characteristics, and package dimensions of the sources [25].

Single illumination sources alone are not the only way to illuminate a scene. There is a wealth of possibilities to arrange various sources geometrically, and eventually combine them with optical components to form an illumination setup that is suitable for different industrial machine vision applications. In many cases, features of interest can be made visible by a certain geometrical arrangement or spectral characteristics of the illumination, rather than by trying to use expensive machine vision algorithms to solve the same task. There are standard illumination techniques such as ***directional*** (or ***specular***), ***bidirectional***, ***rear***, ***vertical***, ***diffused***, ***telecentric***, light and dark field illumination [25], [72]. Some inventive lighting schemes (the so-called *structured* lighting approaches) can be used to derive object features without requiring any computation[53]. For instance, the projection of light stripes or grids onto the object can be used to generate 3-D information about the object under inspection. A structured lighting approach to inspect PCB solder joints has been used in [73].

Pulsed illumination can be used for a variety of purposes, such as increasing the performance of the illumination system, reducing blurring effects, and measuring time constants and distances. Instead of pulsing the illumination signal, it can be modulated with a certain frequency (*modulated illumination*) [25]. Some illumination sources (e.g., special lasers) can only be fired for a short time with a certain repetition rate. Others, such as Light-emmiting diodes (LEDs), have a much higher light output if operated in pulsed mode [25]. LEDs are increasingly popular in the infrared and visual illumination setups for inspection. LEDs illumination has many advantages including stability, graceful deterioration, long life cycle, and ease of control [72].

A detailed overview of imaging optics and illumination techniques can be found in [25].

2.5.2 Image Sensors

The most commonly used image sensors deal with visible and infrared light. These can further be classified into *vidicon* cameras and *solid-state arrays* [75]. Vidicon cameras are based on the principle of photoconductivity. An image focussed on the tube surface produces a pattern of varying conductivity that matches the distribution of brightness in the optical image. An independent, finely focussed on electron beam scans the rear surface of the photoconductive target and, by charge neutralization, this beam creates a potential difference that produces a signal on a collector proportional to the input brightness pattern. A digital image is obtained by quantizing this signal, as well as the corresponding position of the scanning beam.

Solid-state arrays [12], [25] are composed of discrete silicon imaging elements, called photosites, that have voltage output proportional to the intensity of the incident light. *Line-scan* and *area-scan* sensors are two types of solid-state sensors [12], [75]. A line-scan sensor consists of a row of photosites and produces a 2-D image by relative motion between the scene and the detector. An area-scan sensor is composed of a matrix of photosites and is therefore capable of capturing the image in the same manner as a vidicon tube.

Solid-state technology has allowed the elimination of thermionic technology from the capturing of images, which was inappropriate for such applications due to slow frame rates, increased device volume, increased noise [76] etc. The introduction of solid-state technology in image capturing has led to some breakthroughs in industrial vision, since they offer a number of advantages as opposed to the predecessor technology. Some of these advantages are smaller device sizes, robustness against EM noise, higher resolutions, asynchronous triggering (capturing the image the time it is needed), stop-motion techniques (capturing fast-moving objects) [76], on-chip signal processing [77] – [79], robustness against changes of lighting conditions [80], [81] etc. The most important technologies used in integrated imaging sensors are the *Charge-Coupled Device* (CCD), *Charge-Injection Device* (CID) and *Complementary MOS* (CMOS) [82]. Although CCD is a mature technology that is commonly used in industrial vision applications, the potential of the alternative technologies (CID and CMOS) is very high, considering their on-chip intelligent and autonomous post-processing. High complexity algorithms can be implemented for real time vision inspection and new sensors (e.g., CMOS sensors) offering high dynamic range allow for more reliable, flexible and faster image acquisition than traditional CCD sensors, even under poor lighting conditions.

Several criteria are used to evaluate image sensors, the most important being the following: [82], [83] a) *Responsivity*, which is a measure of signal level per unit of optical energy. CMOS sensors are slightly better than CCD in this category, due to the fact that gain elements are easier to place on their chip. b) *Dynamic Range* defined as the ratio of a pixel's saturation level to its signal threshold. CCD sensors are better because they have less on-chip circuitry, which reduces the noise and increases the sensitivity of the sensor. c) *Uniformity*, indicating the consistency of response for different pixels under identical illumination conditions. Circuitry variations affect the uniformity of pixels on an image sensor. ***CMOS sensors*** are more sensitive to these variations because of the more additional circuitry on sensor. Newer CMOS devices have added feedback to the amplifiers to compensate these variations, but this only works well under illuminated conditions. CCD has better uniformity because the lack of any amplification in the sensor itself. d) *Speed* of operation, with CMOS sensors operating faster because most of the circuitry is on board. Thus, the signals communicate less distance and don't have to be piped to other chips on the printed circuit board. CCD imagers still operate adequately fast for most applications, but anticipated demanding applications will consider CMOS sensors instead. e) *Reliability*, in which respect CMOS sensors are superior to CCDs because of the high level of integration contained on the chip. More integration means less external connections that are susceptible to corrosion and other problems associated with solder joints in harsh environments. Overall, CCDs offer superior image performance and flexibility at the expense of system size. CMOS imagers offer more integration, lower power dissipation, and smaller system size at the expense of image quality and flexibility. For next-generation applications, CMOS evolves in order to get around the low–quality problem. Improvements are incorporated by the use of microlenses, which are small lenses manufactured directly above the pixel to focus the light towards the active portion, and the minimization of the space circuitry in the CMOS pixel.

The images can also be taken outside the visible spectrum. For example, infrared cameras are used for thermal imaging, ultra-violet light has been used for crack detection, and X-rays are being used for defect detection in food grains [75] and solder joints [91].

In cases where the camera is not capable of acquiring the images in digital form, an image acquisition board (***frame grabber***) is required to digitize the analogue signals received from the camera and store them as an image in computer memory [74]. In Figure 2.3 below the essential components of a digital image acquisition system are illustrated.

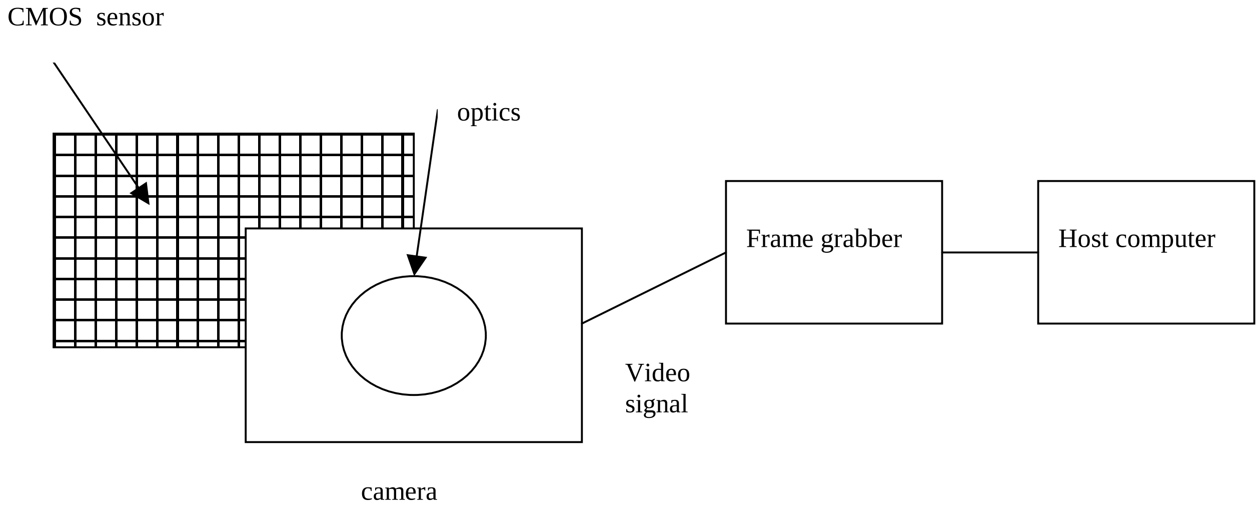


Figure 2.3 Essential components of a digital image acquisition system.

2.5.3 Software Tools

The selection of the appropriate software tools is of crucial importance for development of an industrial vision inspection system. There have been significant advances in the software area especially in user-interfaces and algorithms [2], [72]. The graphical user interface (GUI) has benefited from object-oriented design methodologies, web interfaces and integrated development environments that allow rapid prototyping. Most of these environments support both, visual programming in combination with flexible GUI interfaces and traditional programming. Both programming practices can be combined to facilitate application development. Visual programming can be employed to accelerate application's prototyping whereas the final application can be implemented and optimized using standard programming methods and languages.

Image-processing software has become user friendly and powerful utilizing software libraries implementing some of the most popular image processing and analysis algorithms. An image processing environment to be suitable for industrial inspection, must (at least) contain algorithms for edge and line detection, image enhancement, illumination correction, geometry transforms, Region of Interest (ROI) selection, object recognition, feature selection and classification. Ref [2] provides a review of some of the most popular image processing and analysis software tools offering the desired functionality.

A wide range of computational intelligence approaches like neural networks, fuzzy logic, neuro-fuzzy, and genetic algorithms [38], [41] have been applied to the features generated from the image analysis algorithms for pattern classification. In [2] a review of some of the most popular computational intelligence software packages is provided.

Overall a wide array of the abovementioned algorithms can be prototyped and tested rapidly using the graphical programming blocks available in software packages such as MATLAB and Mathematica.

2.5.4 Hardware Tools

Software implementations are often insufficient to meet the real time requirements of many industrial vision applications. The ever-increasing computational demands of such applications call for hardware tools implementing image processing algorithms. Application Specific Integrated Circuits (ASICs), Digital Signal Processors (DSPs), Field Programmable Gate Arrays (FPGAs) and general-purpose processors are considered as possible alternatives in dealing with the problem of processing speed. The choice among them has to be made taking into account issues such as, size of chip, power dissipation and performance. However, issues such as flexibility of usage, programming environment are now becoming of great importance for the application developers [2].

There are several ways to perform hardware image processing. The first one is to build a circuit dedicated to the application using an ASIC. As algorithms become more complex, the future of ASIC design will use more and more Intellectual Property (IP) blocks available on the market either as a hardware black box units (i.e., layout cells) or software packages (in a hardware description language such as VHDL or Verilog) [2]. The use of DSP boards for the fast execution of image processing algorithms has been extensively used in industrial vision applications with hard real-time constraints. Some popular DSP architectures are the TriMedia Mediaprocessor by Philips Semiconductors [84], IM-PCI by Imaging Technology, MaxPCI by Datacube, Texas Instruments (TI) TMS320Cxx Family, the Genesis Vision processor by Matrox based on the TI's TMS320C80 DSP and on the PCI platform, the Mpact media processor by Chromatic Research Inc. [85] etc. FPGAs are now competitive to ASICs both, in terms of capacity (i.e., number of equivalent gates contained in one chip) and performance. This allows to quickly having prototype of the circuit that has to be designed and able to operate in real conditions. The main advantage compared to ASICs is that FPGAs can be reprogrammed. Complex FPGAs allow to design reconfigurable systems that can efficiently implement real-time image processing algorithms. FPGA-based PCI boards are an attractive alternative to DSP systems [2].

Overall, there exist also platforms that are capable of implementing fuzzy, neural, or hybrid systems. Most of them are based on general-purpose micro-controllers, which are fast enough to execute assembly programs that describe fuzzy or neural systems. On the other hand, there are dedicated processors, such as the SGS-Thomson WARP family of fuzzy controllers, for the acceleration of fuzzy-oriented applications [2].

2.6 Information Support to Industrial Machine Vision Inspection Systems using CAD/CAM Data

To support more advanced, and ***intelligent***, applications of machine vision inspection, it is necessary to achieve a level of integration between inspection systems and other manufacturing facilities and process control functions, such as design, fabrication, assemply, quality control, production management, planning and control. Ultimate objectives of systems integration are to achive a higher level of information sharing and supporting amongst those systems. In this sense, inspection systems should therefore be treated as an integral element of this broader integrated manufacturing environment so as to assume a pro-active role of important on-line generators of product quality information. So, a fundamental issue is the providing information to the inspection system so as to support inspection operations. Such information may be derived ***computer aided design*** (CAD), ***computer aided manufacturing*** (CAM) or any other relevant manufacturing systems/processes [89].

In essence,"inspection" is the process of comparing ***detected features*** with ***expected features*** of a product. Detected features are features extracted from a product under inspection. (i.e., through image acquisition, processing and feature extraction), whereas expected features are product specifications with tolerance. The ultimate utilization of the supporting information is to establish a local model of the products to be inspected. This model will contain specifications of all the expected features (e.g., in the form of ***inspection specifications***, or ***generic rules***) to which an acceptable product should confirm. However, it needs not be a complete representation of the product, but rather contains only specifications used for the purpose of inspection. In other words, it is a local view of a complete product model, being taken from the perspective of inspection.

Among the potential information sources, CAD is one of the most important for such purposes. The ***CAD data*** generated during design can be considered as representing the original product specifications, and are thus defect-free in nature. Use of such data to establish inspection criteria will be much "safer" and more reliable than the conventional "known good product" approach. Where CAD data are not available for such purpose, ***CAM data*** (e.g., often Gerber files for NC Drills or placement machines) can be an important alternative from which necessary information can be extracted to establish inspection criteria, or used for other reference purposes (e.g., nominal locations of components, etc.). With the availability of this off-line generated information the time required to set up the inspection system will be reduced dramatically. Requirement for manual on-line teaching processes (which are often time consuming and error-prone) will be minimal, if not totally eliminated.

For situations where a broad-ranging system integration is not in place, an ad hoc approach can be adopted to establish the information link and to provide CAD/CAM based information support to inspection systems, as has been proposed in [89]. This approach entails the following logical steps, namely:

1. Analyze the target inspection application and identify information requirements of the visual inspection system.
2. Based on the information requirements identified, determine (a) the content of the information to be extracted from CAD/CAM database, and
3. Design and code the necessary software program to fulfil the requirements.

Implementations of the above software program may differ from each other, depending on the end-systems involved, hardware platform, operating system and programming language used. However, a generic functional model can be defined as guideline for developing such software information links [89]. As illustrated in Figure 2.4 the model consists of three functional layers, viz. :

1. Parsing and interpreting the incoming data from CAD/CAM database
2. Extracting the necessary information items, and
3. Representing the extracted information in a format deemed suitable for the inspection system in question.

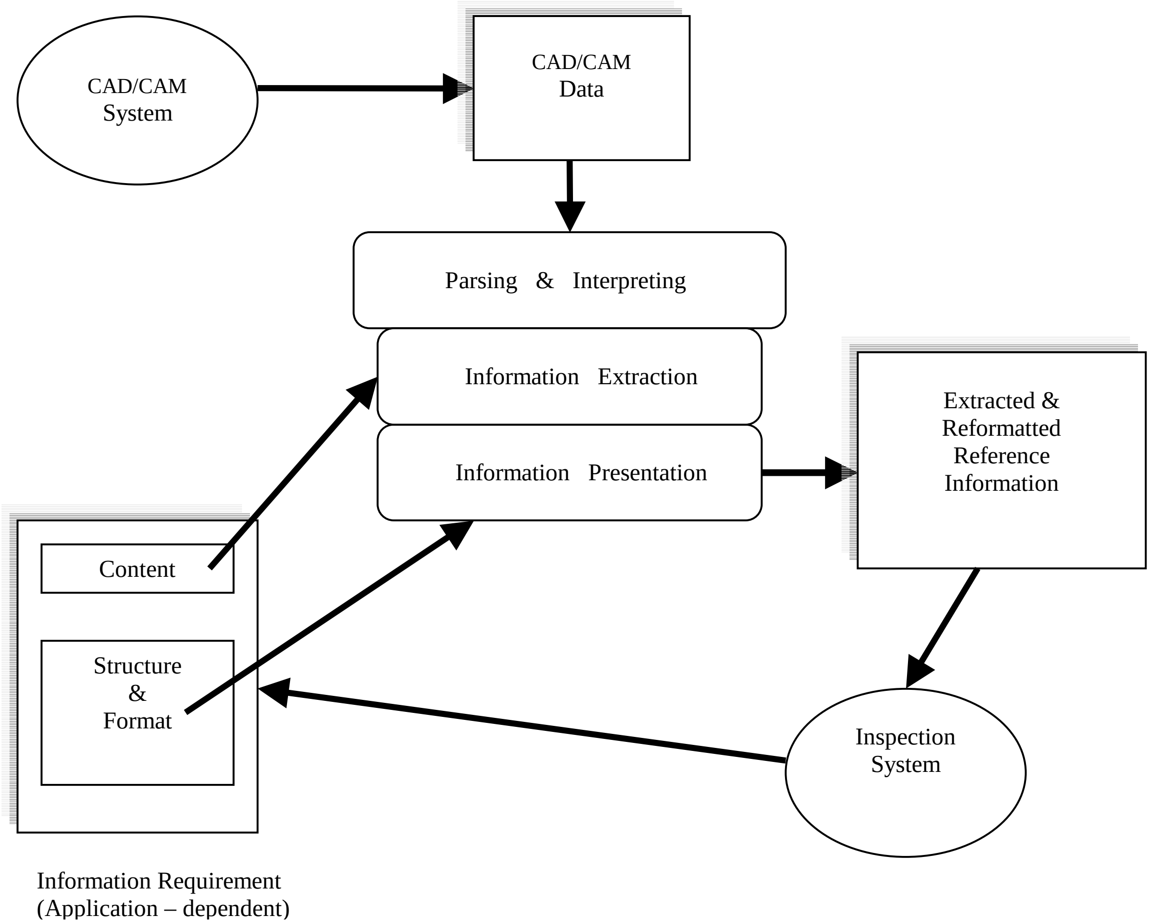


Figure 2.4 Key issues in providing CAD/CAM information to support industrial visual inspection

Chapter 3

The Problem of Sourface-Mount Devices Printed Circuit Boards Post Placement Quality Inspection

3.1 Introduction to Electronics Manufacturing

Electronics manufacturing is the process of design, development, fabrication, assemply, and testing of electronic parts, tools, technologies, components, and systems. In electronics manufacturing, several major stages are involved in the manufacturing of an electronic product. Component fabrication results in a packaged part which performs an electrical or electronic function. These parts are then assembled onto a printed circuit board (PCB), and PCBs are, in turn, assembled into a system. The typical manufacturer performs some, but not all, stages of the production process.

In electronics manufacturing, the majority of components fall into two main categories: through-hole components (THC) and ***surface-mount devices*** (SMD) [6], [56]. THC are components that have wire leads which must be inserted through predrilled hole on a PCB. The wire leads are then clinced, trimmed, and soldered. The leads of these components serve the dual purpose of providing circuit connectivity, by being soldered to the circuit paths, and acting as a secure mounting structure to hold the component in place.

Surface-mount components (SMC) or surface-mount devices (SMD) are components that are mounted directly to the surface of the PCB, so that it is not necessary to have holes drilled through the substrate to mount the components [6], [56]. Almost any type of THC will have a counterpart SMD. The SMDs only differ from their THC counterparts by their packaging. Whereas the leads do not have to pass trrough the PCB, the leads can be much smaller than their THC counterparts. ***Surface – mount technology*** (SMT) requires pickup, centering, and placement of components. The circuit density and fine lead pitches of SMT necessitate automated assembly with integrated industrial machine vision systems. SMDs are packaged in tape and reel, tube, and waffle pack. Dedicated placement machines are then used to place the components on the PCB.

In general, electronic systems are made up of several layers or levels of packaging which must be interconnected. Each level of packaging has distinctive types of interconnection devices associated with it. The hierarchy of interconnection varies from gate-to-gate interconnections on silicon chips (IC chips) to cables used to interconnected subsystems. At the chip-level, vacuum-deposited thin-film metal is used to interconnect individual devices. Chip input/output is accomplished using a variety of techniques such as wire bonding. In recent years, electronics packaging manufacturing has developed low peripheral leaded packages, such as SOP (Small Outline Package), SOJ (Small Outline J-leaded Package), SOIC (Small Outline IC), QFP (Quad Flat Package), COF (Chip-on-Film) and BGA (Ball Grid Array) [86] – [88].

At the PCB level, printed conductor paths connect the device leads of components to PCBs and to the electrical edge connectors for off the board interconnection. For higher levels of integration, electronics systems utilize cables for signal propagation between subsystems. The use of cables necessitates the use of connectors to provide mechanical and electrical linkage.

3.2 The Surface Mount Process and Automatic Visual Inspection of Printed Circuit Boards

Central to this PhD thesis is the research in electronics manufacturing in the area of quality with the development of novel pattern recognition algorithms for the inspection of Surface Mount devices PCBs, which is one of the most important challenges faced by the electronics industry today. Since the emergence of Surface Mount Technology the electronics products are more compact and complex and the added complexity has become more difficult for electronic assemply processes to attain the required quality on the end products [109].

Human inspection is inefficient, time consuming, and costly process that increases the overall production costs without being 100% reliable, and with the application of SMT, it is not feasible to use the naked eye anymore to perform certain visual inspection tasks where very close tolerances are required. Thus, the use of Automated Visual Inspection (AVI), has become a necessity in the electronics industry [61], [65].

The problem of Visual Inspection of Surface Mounted Devices on a PCB has been studied using different algorithms to get information from 2D and 3D images. 2D images are rapidly acquired with respect to 3D ones, but depth information is not directly available from the resulting image and they required additional expensive computational analysis. The computing cost required for 2D images has been reduced significantly by transforming them into a one-dimensional signal keeping enough information to determine the main characteristics of interest of the object under inspection, based on vector classification.

Surface mount devices are designed to be placed on the board automatically and not hand soldered. There are three main steps in the assemply of SMD PCBs: *screen printing*, *device onsertion* (or “onplacement”) and *solder reflow* [56], [60]. The first stage deposits solder paste onto the conductor pads on the PCB. This is done by stencilling on the solder in a paste form. The second stage requires the placement of the devices onto the solder paste deposits. In the third stage the solder must be melted or “reflowed” to make a mechanical and electrical bond.

Proper applications of industrial visual inspection systems can result an increased product inspection throughput, improved inspection reliability and more consistent inspection results. More importantly the earlier in the manufacturing cycle an automatic inspection system is used, the better the PCB quality will be and the less the product scraps will result. As the structure of PCB product is becoming increasingly complicated, a single missed defect in any of the inner layers of a board would cost the entire board to be scraped. The philosophy is therefore to detect defects as early as possible so as to avoid adding additional values to defective boards or layers.

As illustrated in Figure 3.1, numerous opportunities exist for applications of industrial visual inspection systems in various stages of PCB manufacturing (from bare board fabrication to PCB assemply). However, these seemingly diverse applications can be classified into four major categories, these being *panel inspection*, ***solder paste inspection***, ***component placement inspection***, and ***solder joint inspection*** [89].

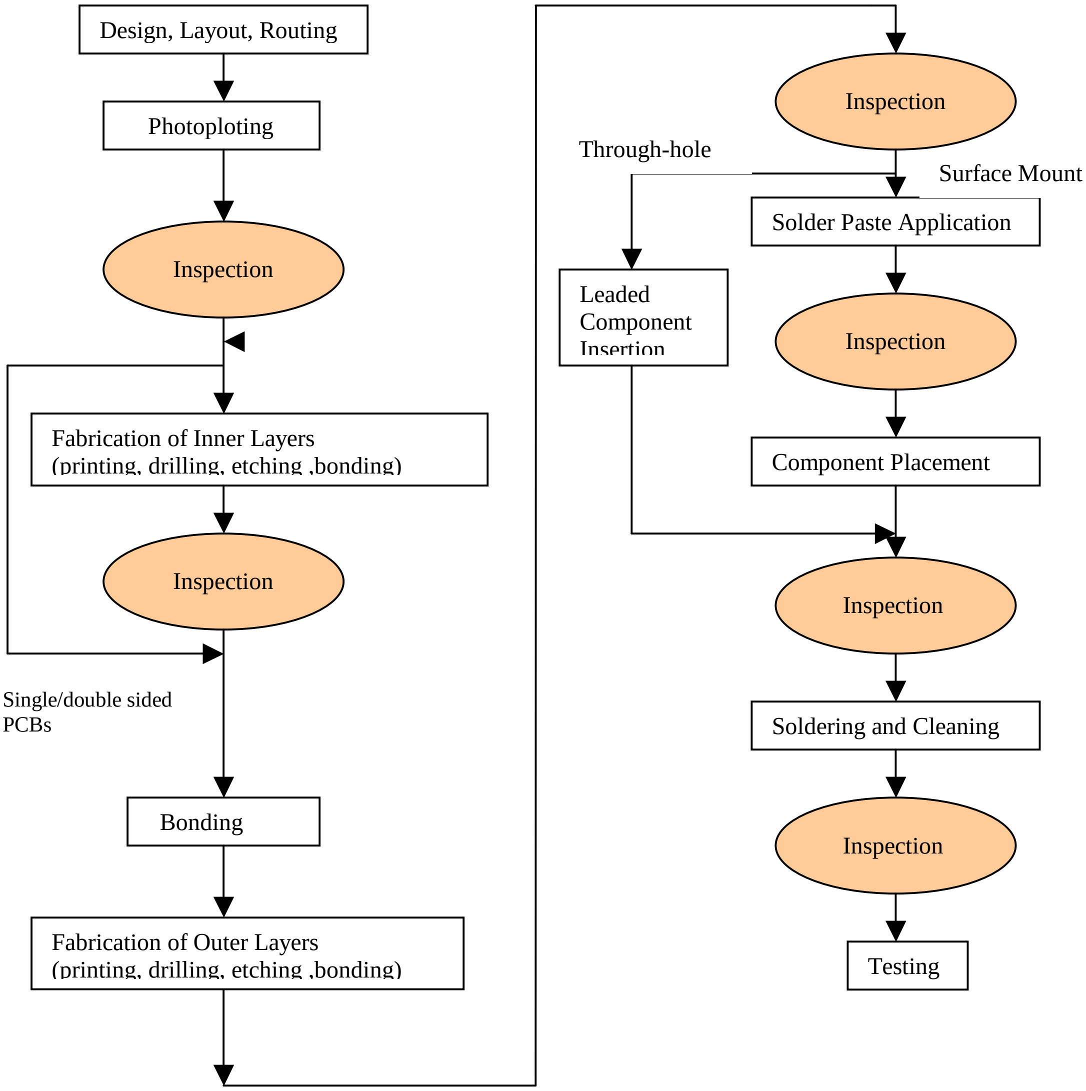


Figure 3.1 Applications of industrial visual inspection

3.2.1 Panel Inspection

The term *panel* refers to a representation of a PCB artwork in the form of artwork films, production master films, individual layers (e.g., signal layers the ground layer, etc.), or bare boards. Panel inspection thus refers to the inspection of these various artwork films, master films, inner layers, bare boards, and so on. Basically panel inspection is concerned with the inspection of features that are essentially of a two-dimensional nature. The requirements are to verify track width, via hole size and shape, soldering pad shape, and size. In addition, it is also required to verify the spacing clearances between track and pad, pad and pad, or track and track.

Conventionally used techniques for panel inspection can be classified as falling into one of the following categories, namely, (i) image comparison based techniques, (ii) rule checking based techniques, (iii) combined approaches.

3.2.2 Solder Paste Inspection

Entering the cycle of PCB assmbly, the first operation (for SMT) is screen printing of solder paste onto the bare board. Solder paste application has a direct and often decisive effect on the final quality of the solder joints (and thus the functionality and reliability of the PCBs) Thus the inspection of solder paste is a must for PCBs involving SMT. The requirements are to check, prior to component placement and soldering, the volume and the three-dimensional distribution of the of the solder paste applied by the screen printing process.This is in effect to make sure that the right amount of solder paste has been supplied at the right place and in good alignment with solder pads.

Depending on the requirements of inspection, two basic classes of inspection techniques can be differentiated, namely, two-dimensional (2D) based and three-dimensional (3D) based methods. If only alignment and area need to be verified, then 2D techniques will be enough. However, if volume and/or spatial distribution of the solder paste need to be checked, then some 3D inspection techniques will be required. In addition, special illumination and filtering techniques may also be required to facilitate the acquisition of an image with a high contrast between the metal solder paste and any other background materials (e.g., the substrate).

3.2.3 Component Placement Inspection

Where there is less confidence in the placement equipment or where complex components are used, it is often necessary to perform component placement inspection both pre-soldering and post soldering. Typically, this application is concerned with (i) checking the presence/absence of components, (ii) verifying that the right component (i.e., its type and value) is placed on the right place and with the right orientation.

Pre-soldering inspection helps isolate misplaced components before they are physically fixed on the board (i.e., by solder joints) and thus helps minimize post soldering rework/repair. On the other hand, post-soldering placement inspection performs a final inspection, in-circuit test and functional test, to verify the presence of certain components as required. This has an additional advantage of limiting the range of possible variables affecting the electrical functionality and thus assists test equipment in making decisions regarding the board testing.

3.2.4 Solder Joints Inspection

Automating visual inspection of solder joints is probably the most difficult and demanding, yet important inspection task to tackle. The objectives are to isolate soldering defects such as insufficient/excess solder, poor wetting, missing lead, bridging/opens, solder balls and so on. Due to the shape and the high reflective nature of the metal joints inspected, proprietary built illumination systems are almost always a prerequisite to obtaining any useable images of solder joints. For instance the use of *structured light* [90], *tiered illumination* [91], and *sufficiently-diffused* illumination [4] have been reported.

The introduction and increasing application of surface mount technology contributed further to the difficulties of solder joints inspection. Not only the number of solder joints to be inspected has increased dramatically, but also image acqisition of solder joints becomes more difficult. In some cases the solder joints may not even be visible at all as a result of using some special surface mount packages such as plastic leadless chip carriers (PLCCs) and leadless ceramic chip carriers (LCCCs). In these cases, a special X-ray lighting and X-ray sensitive image acqisition subsystem will be required to obtain images of solder joints that are underneath components [92].

3.3 Problem Formulation and Requirements of Industrial Manufacturer for SMD Post Placement Quality Inspection

The goal of this PhD thesis is the development of novel intelligent pattern recognition algorithms to inspect the placement quality of a selected group of SMD's immediate after they have been placed in wet solder paste on a printed circuit board (PCB).

SMD post placement quality inspection requires a high dynamic range because a distinction should be made between a shiny footprint and a shiny SMD lead. It also requires fast image intake and fast data processing to sustain the required throughput of the Advanced Component Mounter (ACM). A CMOS imaging sensor with its high dynamic range and area readout together with a custom image processor complies with these requirements.

The SMD post placement quality inspection is done to comply with the zero-defects policy of today’s electronics manufacturing industry. The current achieved yields are already very high, so an inspection system has to be very accurate and robust otherwise it will cause a lower yield than without the system. Post placement inspection has the advantage that the inspection data is available immediately after placement so no more time and components are spend on an already faulty PCB. The later a defect is caught, the more expensive it is to repair, and so catching a defect early in the process is inherently cheaper. Correcting a defect after re-flow produces a more brittle joint and increases risk of field failure, which is very expensive to repair. Therefore, detecting a defect before re-flow will save money and increase reliability. Another advantage of post placement inspection is that it can be used to predict possible future board failure. Electrical and functional tests do not always yield this information.

The following two components are inspected here as a test case:

- an SO Component, and
- a QFP component with pitch of 0.4mm.

Examples of the two components are shown in Figures 3.2 and 3.3 correspondingly.

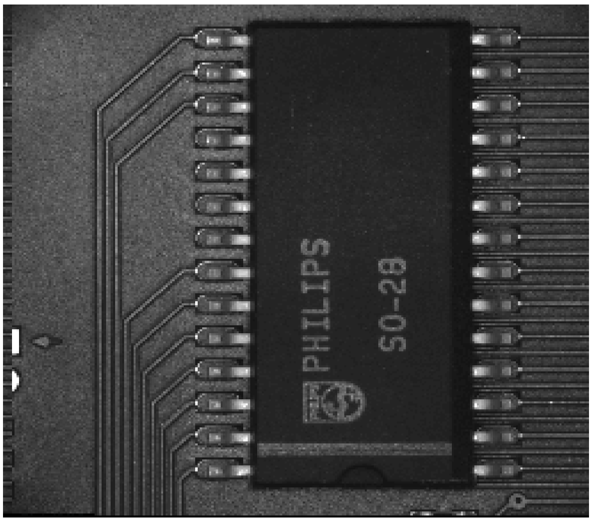


Figure 3.2. The SO28 Component

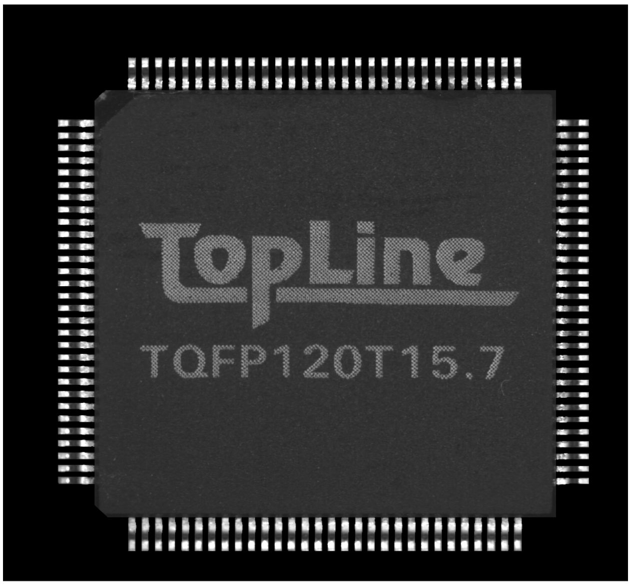


Figure 3.3 The QFP 120 Component

The CMOS camera, including illumination, will be mounted on a placement arm of an ACM. The placement head, which is usually mounted to the arm, will be detached for this. The CMOS camera will be mounted next to the placement head in its final industrial version.

Some assumptions are made to simplify this problem, which are valid for the majority of the cases:

- The components comply with the specifications (they are checked before placement)
- The solder paste distribution complies with the specifications, this is best checked with a 3D-camera or 3D-scanner before the components are placed
- The component’s footprint and artwork comply to the specifications
- The PCB is green, this is true for 99% of all cases
- Re-flow soldering is used
- The metal surface finish is
- Bare copper with OSP (a.o. Enthone OMI, Entek)
- HASL (Hot Air Solder Levelled)
- Electron-less Ni / Immersion Au

The resolution of the camera will be 1024 × 1024 with a field of view of 20 × 20 mm. This results in a resolution of about 20μm × 20μm per pixel. The images are taken with the same focal length of the lens. Moreover, the solder paste is assumed to be located on the pad area.

The pad has precise shape and dimensions and its location relative to the image is assumed known within the tolerance of the placement machine (50μm). The required measurement accuracy is 20μm. The sizes of the components and the pad areas are specified in the following.

For the SO 28:

Overall size of the component: L=17.7-18.1mm, W=10.0-10.65mm

Pitch of component: 1.27mm

Width of each lead: 0.36-0.49mm

Dimensions of the land pattern (pad):

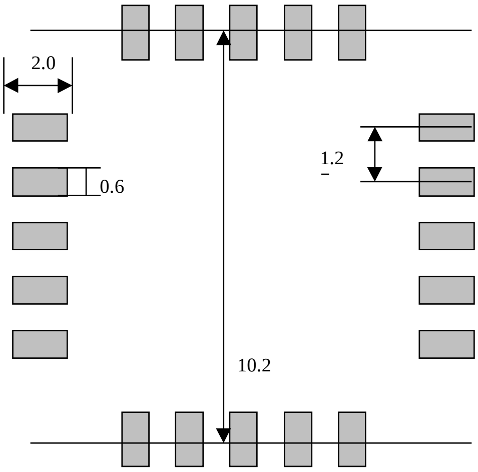


Figure 3.4 Size and pad area of SO28 Component

For the QFP 120:

Overall size of the component: L=W=15,8-16.2mm

Pitch of component: 0.4mm

Width of each lead: 0.13 - 0.23mm

Length of each lead: 1.12 – 1.18 mm

Dimensions of the land pattern (pad):

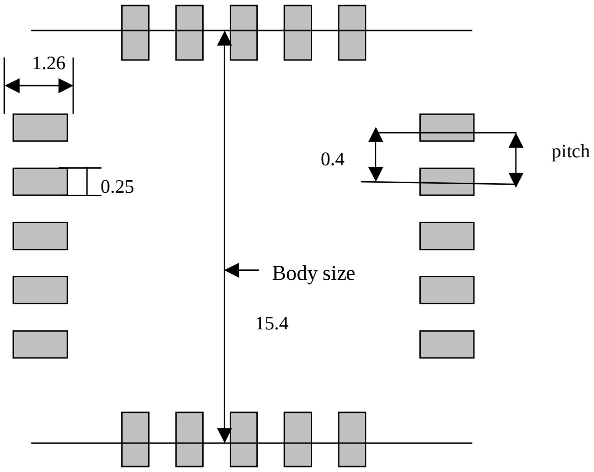


Figure 3.5 Size and pad area of QFP 120 Component

3.4 Simulation Platform Specifications for SMD Post Placement Inspection of Components

The simulation in our application has been based on images that have been taken by industrial manufacturer right from the test-bed. It is the purpose of this simulation to design and test novel algorithms that can be useful in SMD component visual inspection. For this reason, three major sources of error in the SMD placement process will be examined, i.e. ***open circuits***, ***smearing of solder paste***, and ***component miss-alignment*** relative to the placement head co-ordinates.

The following properties of a placed SMD define the placement quality:

- SMD presence (existence of lead).
- SMD positioning (rotation angle $\Delta\varphi$) relative to the position of the Solder Land (See Figure 3.6)
- SMD positioning with respect to solder pads: 3 criteria (See Figure 3.7)
  - Insulation distance: This is the distance between the lead and its neighbouring solder pad. When this distance becomes too small, a short circuit can occur more easily during soldering. An empirical size for a minimum distance is 0.1 mm.
  - Overlap (adhesion width): This is the width of the lead that is on the solder pad. Generally minimal 50% of the lead width should overlap with the belonging solder pad. Preferably 75% should overlap.

- Slump gap: This is the distance between a lead and its neighbouring solder paste deposit. When this distance becomes too small, a short circuit can occur during re-flow soldering. Empirical sizes for a slump gap vary from 0.1 to 0.2 mm.

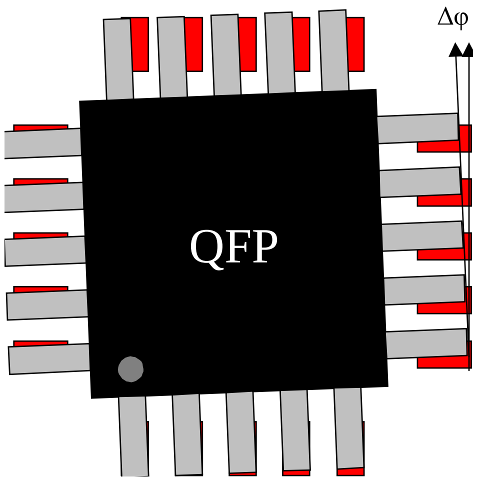


Figure 3.6. The rotation of the SMD, relative to the solder land.

Currently these are the only properties of the placement process that are not checked because it is not possible to make reliable measurements with “common” CCD cameras because of their limited dynamic range. The difficulty here is that a distinction should be made between a shiny copper footprint and a shiny SMD lead. The dynamic range of CCD cameras is insufficient for this.

Solder paste volume has been identified as the single best predictor of finished board quality. The paste volume is best checked before placement because then the view is not blocked by the component. Some components, like BGA’s, don’t even allow inspection after component placement because the paste isn’t visible anymore at that time.

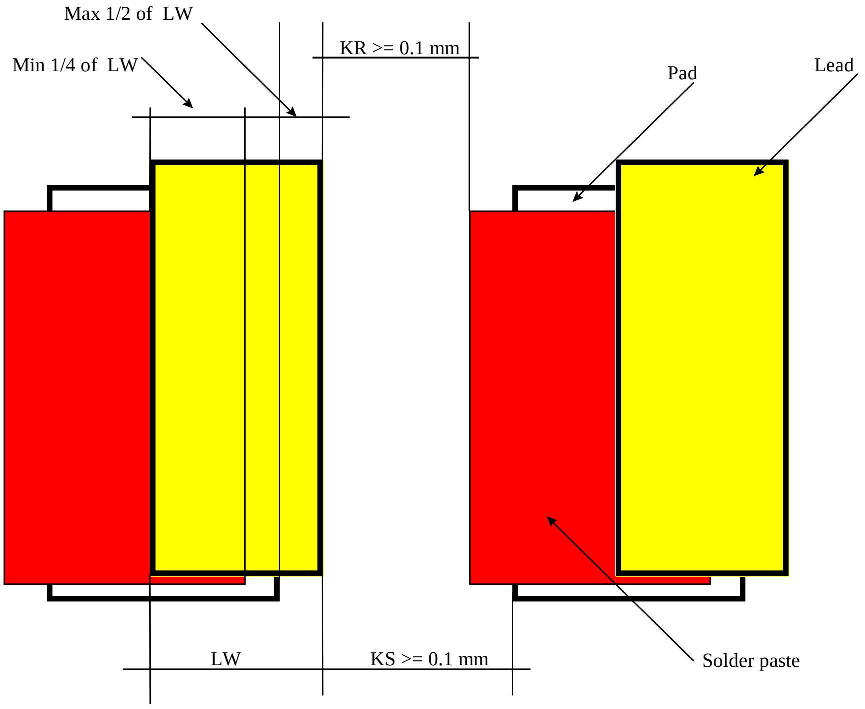


Figure 3.7. The placement quality criteria, LW = Lead width, KS = Insulation distance, KR = Slump gap

3.4.1 Smearing of Solder Paste

This error can be caused by the placement force, which can cause the components to slide over the PCB while they are placed. Smearing of the solder paste can be a local phenomenon, which is very difficult to detect. A 3D-camera is much more suitable to detect this. Some experiments have been done by industrial manufacturer to find out if it is possible to robustly detect smearing of the solder paste next to the leads. Figure 3.8 illustrates this.

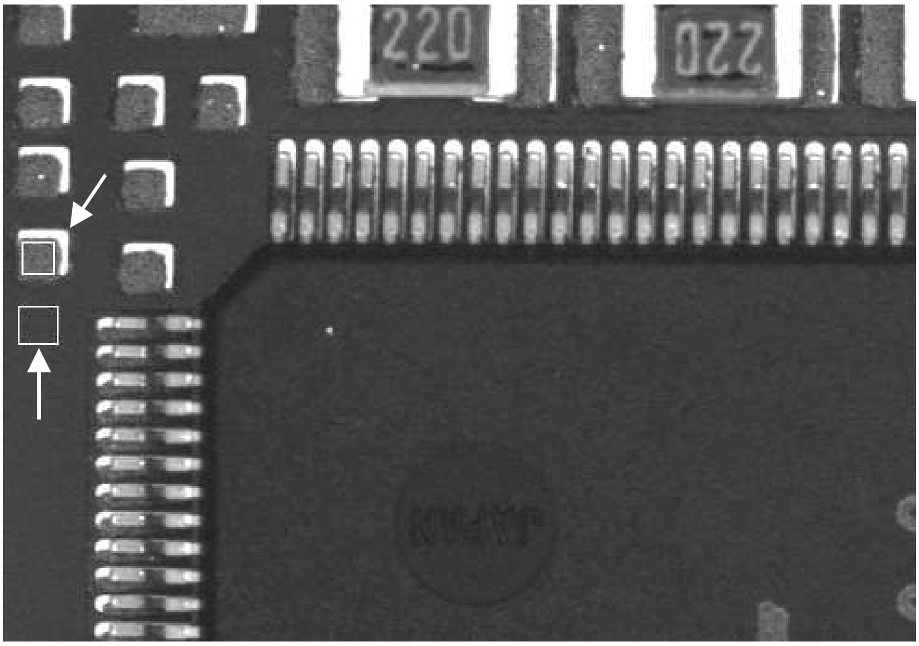


Figure 3.8 A corner of a quad flat pack

The image in Figure 3.8 only shows a small contrast between the board and the solder paste. Two white rectangles are indicated in the left side of the image. The upper rectangle indicates an area that only contains solder paste. The other rectangle only contains circuit board. The average grey level in the paste area is 103, in the PCB area it is 67. The contrast between the solder paste and PCB is even smaller next to the leads because the leads block some light that would fall onto the solder paste. The standard deviation of the paste area is about three times as high as the deviation in the PCB area. This makes accurate segmentation of the solder paste areas between the leads very difficult if not impossible.

The solder paste in Figure 3.9 has an offset to the left. This is very difficult to detect even for the human eye. Another example is shown in Figure 3.10.

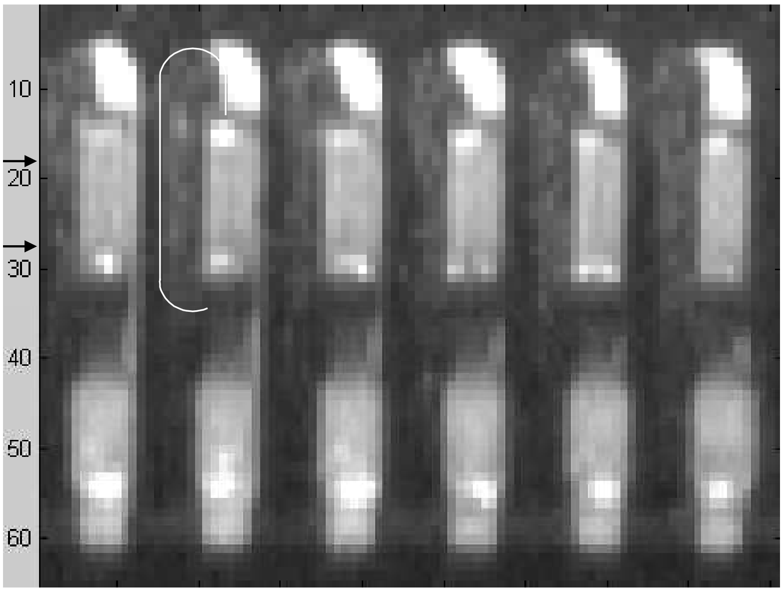


Figure 3.9. Part of Figure 3.8 enlarged (one paste area is outlined by hand)

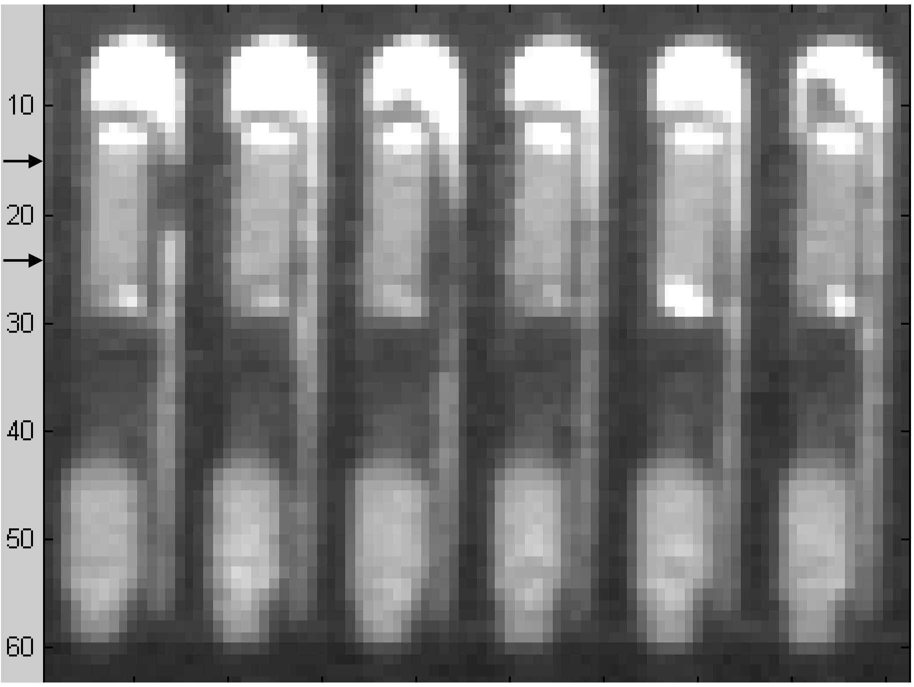


Figure 3.10. Another part of figure 3.8 enlarged

3.4.2 Component Position Relative to Placement-Head Coordinates

The CAD data plus the position data of the placement head give enough information to place a ruler at a suitable column/row over the leads. The actual position of the component, relative to these rulers and, consequently, the placement-head coordinates is not given immediately.

The QFP Component

Figure 3.11 shows a model of a QFP with the contact areas of the leads with the PCB.

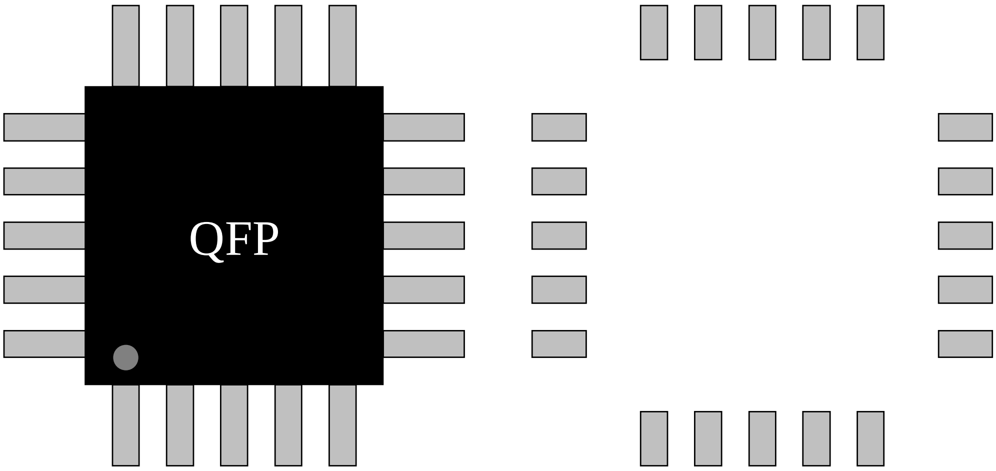


Figure 3.11 A model of a QFP (left) and the contact areas of the leads with the PCB

The size of the contact area's doesn't only depend on the placement, but also on the length of the leads and how the leads are bend. The housing cannot be used to measure the component position because the housing is manufactured to inaccurate for this.

The basis for the position measurement of a QFP is formed by measuring each lead close to package. A ruler can be put over all leads on each side. This part of the lead is very suited because the lead shows a high contrast to its surroundings at that place. A QFP and an edge map of the QFP are shown in Figure 3.12 with a row and column indicated that could be used to place a ruler.

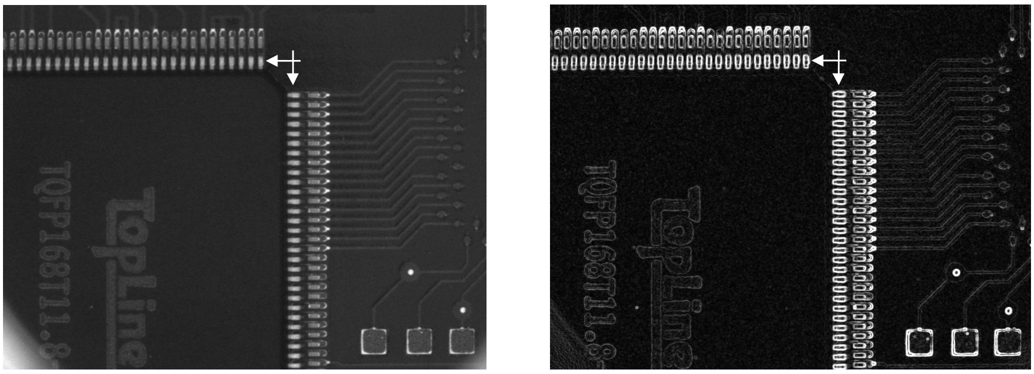


Figure 3.12. A QFP and its edge map

The part of the lead closest to the housing is visible as a very regular pattern in the edge map. The rulers over the columns/rows don’t reveal the component position immediately. The results from this measurement can be used to place rules in the direction of the component leads to measure the actual component position. This can be very tricky because of the different circumstances around the lead ends. The Figure 3.13 shows a side view of a QFP lead.

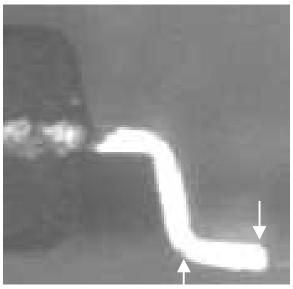


Figure 3.13. A side view of a QFP lead

The points indicated in Figure 3.13 are the points that should be found by the rulers, which are placed in the direction of the leads. These points are clearly visible in the edge map in Figure 3.12.

The part of the lead closest to the housing is located about 1 mm above the substrate surface. The imaging optics should produce sharp images at this height as well as on PCB level. Telecentric lenses can be used for this. A very accurate localisation measurement can be done by averaging all found lead positions.

The SO Component

About the same approach as with the QFP can be used for a SO, but of course a SO only has leads on 2 sides. A model of a SO and the contact areas of the leads with the PCB are shown in Figure 3.14.

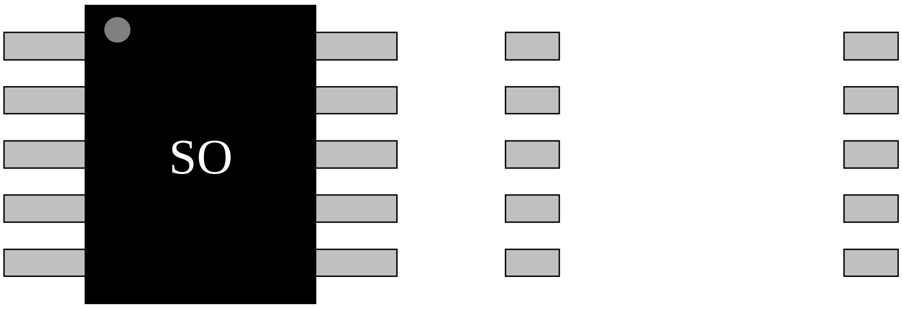


Figure 3.14. A model of a SO (left) and the contact areas of the leads with the PCB

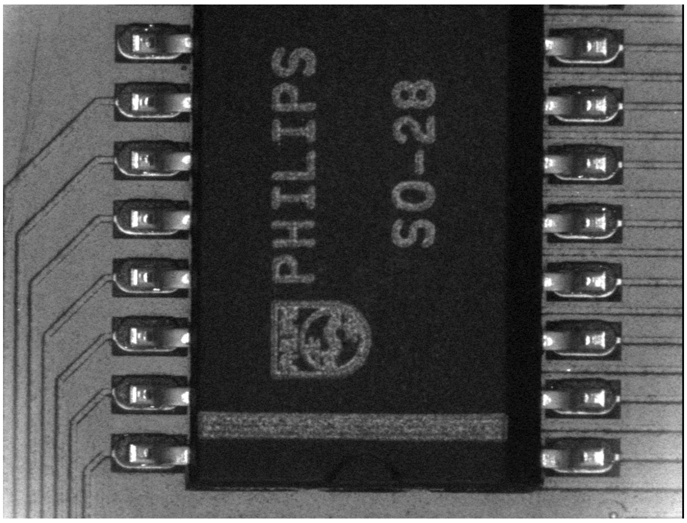


Figure 3.15. A SO28

Figure 3.15 shows that a bright reflection of the lead is visible at the lead endpoint and close to the housing. These positions can be found by thresholding, this is done in Figure 3.16.

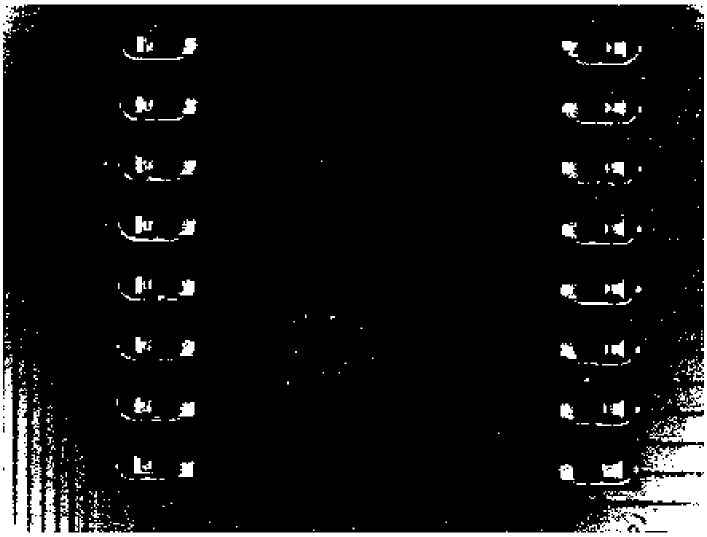


Figure 3.16. The image from figure 3.15 thresholded

Figure 3.16 directly yields the lead positions, one has to take care not to mistake the uncovered solder pads for the leads.

3.5. Previous Related Research

Current trends in the electronics industry are towards miniaturization of components, denser packing of boards and highly automated assembly lines. The technology of Surface Mounted Devices (SMDs) leads to this direction, thus explaining the substantial increase in the use of its various versions. The aforementioned advantages though, make the quality inspection of SMDs more critical and demanding [4]. Various SMD defects have been reported in the literature [5], including component misplacement and absence, component with wrong polarity, solder joint defects and component shifting. Much of the current research efforts are concentrated on detecting solder joint defects. The types of solder joint defects include surplus solder, insufficient solder and no solder. Component shifting has also been reported as a special defect in SMD technology [93].

Commercially available automatic solder joint quality inspection systems are based on laser infrared signatures [94], digital radiography [95], laser Doppler vibrometry [96], or laser acoustic microscopy [97]. These methods however, are destructive, slow and very expensive, so alternative machine vision systems have become quite popular. Several visual inspection systems for solder joints have been reported [4], [98], [99], [100], [101], with different illumination and processing techniques. Conventional visual sources and sensors are not sensitive to dim light diffused by the surface of the solder, saturate quickly for direct light emitted by specular reflection, and often create varying lighting conditions. Thus, processing and analysis are restricted to specific tasks, such as the quality of soldered joints in terms of the distribution of solder paste [102], inspection of IC wafer contamination [103], post-sawing inspection [104], and segmentation of PCBs [105], [106]. Most approaches involving visual sensors attempt to increase the quality of the images acquired through appropriate lighting conditions, in order to make their processing and analysis more efficient. Controlled light sources using LED arrays [5], [99], circular color lamps [107], [108], laser sources with parabolic mirrors or range finders [100], [109], [110], have been used to acquire and analyze the 3D structure of SMDs on the board. Such approaches are also directed toward soldered joint inspection through pattern classification schemes.

Chapter 4

Algorithmic Concepts

The theoretical background of this study based upon two different disciplines, i.e. pattern recognition and image analysis. In the first part of this chapter, the algorithmic concepts of pattern recognition are described whereas the second part concentrates on the image analysis techniques.

4.1 Pattern Recognition Approaches.

The problem of ***pattern recognition*** [7 – 13] can be seen as of classifying a group of objects on the basis of certain subjective similarity measures. Those objects classified into the same ***pattern class*** usually have some common properties. The classification requirements are subjective, since different classification occurs under different properties (features) of the objects.

Given any particular pattern recognition problem, the first task is to choose a discretization method in order to obtain a ***measurement vector*** for each sample pattern. A major difficulty often arises when using these discretization methods, since the dimension of the ***measurement space*** is usually very large. It is therefore common practice to try to reduce this dimension by mapping the measurement space Ω_Z into a ***pattern space*** or ***feature space*** Ω_X, where dim(X) << dim(Z), while retaining as many properties or ***features*** of the original samples as possible. For this reason, this part of the pattern recognition problem is called ***feature extraction*** (or preprocessing) and results in a set of samples from the feature space.

The concept of ***pattern classification*** may be expressed in terms of the partition of the feature space, thus forming a mapping from the feature space to the ***decision space***. Suppose that *N* features are to be measured from each input pattern. Each set of *N* features can be considered as a vector $\mathbf{x} = (x_1, x_2, ..., x_N)^T \in \mathbb{R}^N$ called a feature ***vector*** or a point in the *N*-dimensional feature space Ω_X. The problem of classification is to assign each possible vector or point in the feature space to a proper pattern class. This can be interpreted as a partition of the feature space into mutually exclusive regions (i.e., ***decision regions***) where each region corresponds to a particular ***pattern class.*** A ***classifier*** partitions feature space into class-labelled decision regions. In order to use decision regions for a possible and unique class assignment, these regions must cover $\mathbb{R}^N$ and be disjointed (non-overlapping). The border of each decision region is a ***decision boundary***. Figure 4.1 depicts the recognition/classification process.

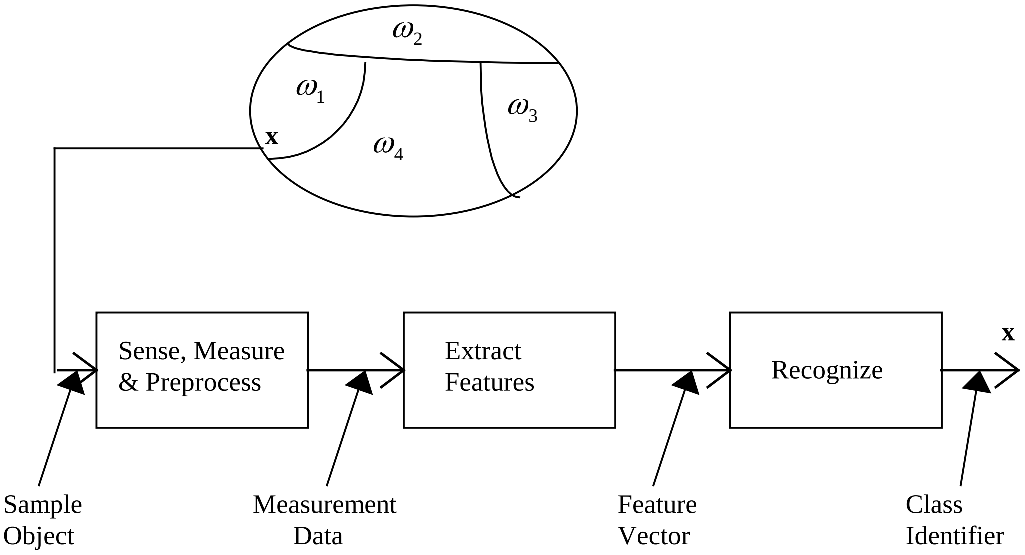


Figure 4.1 The recognition/classification process

Suppose that a given pattern recognition problem has M different pattern classes denoted by $\omega_i,\ 1 \le i \le M$ In several problems we can obtain a number of sample patterns of known classification, say $\mathbf{x}^{i,j}$, where:

$$\mathbf{x}^{i,j} \in \omega_i,\ 1 \le i \le M,\ 1 \le j \le S_i$$

i.e. we have S_i samples pattern class ω_i. Note that each sample vector $\mathbf{x}^{i,j}$ is an element of the pattern space $\mathbb{R}^N$ - we assume that feature selection has already been carried out. Thus, $\mathbf{x}^{i,j}$ has N components, denoted by $x_k^{i,j},\ 1 \le k \le N.$ We have used superscripts to denote pattern samples so as not to conflict with vector components.

A pattern recognition system is, then, a system which takes a new sample **x*** of unknown classification and assigns it to some pattern class ω_i $(1 \le i \le M)$ on the basis of some ***decision*** or ***classification rule***. The decision rule is often obtained by partitioning pattern space into disjoint regions corresponding to the classes ω_i (Figure 4.1). The hyperplanes separating the pattern classes are called ***decision boundaries*** with dimension (*N* -*1*).

The selection of the decision boundaries can be made in a variety of ways. The simplest method is to use all the labeled samples simultaneously and find the 'best' partition of the pattern space which places the samples as far from the decision boundaries as possible. This type of decision boundary selection leads to the ***minimum-distance pattern classification technique*** [7], [9], [11]. One drawback with this kind of method is that once the decision boundaries are placed according to some finite set of samples they are fixed throughout the lifetime of the pattern recognition system.

There exist two distinct phases in the pattern classification process - ***training phase*** and ***testing phase***. In the training phase, the labeled samples are presented to the system sequentially and the decision rule is altered by a 'teacher' which corrects any errors in the classification of the current sample on the basis of the previous decision rule. Once the system has been trained by the labeled samples, it can then be used in the testing phase to classify new samples of unknown classification. In order to test the system it is usual to set aside some of the data whose class is known during the development of the system and then to use this ***testing set*** to evaluate the performance of the ***classification rules***. There are a number of methods for evaluating the classification rules, depending on how the test set is chosen. If we have a large number of samples it is a common tactic to divide them into two groups and use one for ***training*** and the other for ***testing***. If, however there are only a few samples of known class it may be necessary to use what is known as the ***leave one out*** method for assessing the success of the classification rules. This method entails using all the cases except one as the ***training set***, and then using the excluded case as the ***testing set***, repeating this process until the whole set has been tested.

A system which is trained on the basis of labeled samples is said to undergo ***supervised training***.

In problems where one actually wishes to determine a classification scheme for a variety of data vectors (rather than assign new samples to an existing classification) there are no labeled samples, since the classes are not specified *a priori.* In this case we can use *unlabeled* samples to determine a 'natural' classification or clustering for the problem. A typical example of a classification problem is animal taxonomy in which one wishes to classify unknown species on the basis of comparative anatomical or genetic features. The basic method in the classification problem is called ***clustering*** or ***cluster analysis*** [9], [11] and seeks to find subsets of the samples, called ***natural groupings*** or ***clusters***, whose elements are mutually 'close' but far away from members of other clusters. The training of a pattern classifier with unlabelled samples is expressed as ***unsupervised training***.

Parameter estimation [9], [11], [12], [13] is the process of attributing a parametric description to an object based on measurements that are obtained from that object. Parameter estimation and pattern classification are similar processes because they both aim to describe an object using measurements. However, in parameter estimation the description is in terms of a real-valued scalar or vector, whereas in classification the description is in terms of just one class selected from a finite number of classes.

Pattern recognition is a very large subject which draws together methods from various related disciplines. There exist different approaches to pattern recognition, such as ***statistical***, structural, ***neural network***, ***fuzzy logic*** and ***genetic algorithms*** approaches. In this Ph.D. thesis we examine a wide range of PR approaches each focusing on particular characteristics and revealing different relationships of the data.

4.1.1 Statistical Pattern Recognition

In this section we discuss the statistical approach to pattern recognition [7 – 9], [11 – 13], [35]. This approach is based on the statistical study of measurements made on the data to be classified. The problem of assigning a feature vector to a particular class is tackled by estimating density functions in the *N*-dimensional space, and dividing the space into regions of categories or classes. By means of stochastic considerations, a classification rule is optimal in the sense that it results in the lowest average probability of committing classification errors. This statistically optimal classification rule is a generally accepted standard against which the performance of other classification algorithms is often compared.

As in most fields of measuring and interpreting physical events, statistical considerations become important in pattern recognition because of the randomness under which pattern classes are typically generated. For instance, consider the problem of classifying (SMD post placement quality inspection) component displacements on the pad regions into five classes: ***-6 pixels shift***, ***-3 pixels shift***, ***0 pixels shift*** (i.e. without shift, or normal case) ***+3 pixels shift*** and ***+6 pixels shift***. The sample patterns for these five classes would be obtained by gathering numerous PCB with component displacements which have been labeled as ***-6 pixels shift*** or ***-3 pixels shift*** or ***0 pixels shift***, or ***+3 pixels shift*** or ***+6 pixels shift***. Clearly, these samples would form a statistical distribution since, for example, there would be great variability or randomness among the component displacements labeled as normal. This randomness would be due to the varied technical problems during SMD component placement.

4.1.1.1 Bayesian Classification

The decision-making process in pattern recognition may be treated as a statistical game played by the classifier of the pattern recognition system against nature. This process is analogous to a two-person zero-sum game with nature acting as a player *A* and the pattern classifier acting as player *B*. A zero-sum game is a game in which one player's gain is equal in magnitude to the other player's loss. Among the strategies used are the Bayes strategy, the minimax strategy, and the Neyman-Pearson strategy [7], [9], [11]. The job of the classifier is to find an optimal decision which minimizes the average risk or cost.

In the framework of PR problems, we may imagine that nature is player *A* and that the classifier is player *B*. We call the strategies of *A* the states of nature, which will be denoted by ω_i. The states of nature correspond to pattern classes. The strategies of the classifier are decisions concerning the states of nature. In the following discussion it will be assumed that the number of decisions is equal to the number of possible classes.

Each time that the game is played, nature selects a strategy ω_i according to the probability $P(\omega_i)$, which is called the ***a priori*** (or ***prior***) probability of class ω_i. This is simply the probability of occurrence of class ω_i . The outcome of nature's move is a sample pattern **x.** In other words, we do not know which class nature has chosen. All the information that we have is a sample **x**. The job of the classifier is to determine, on the basis of this information, which class **x** came from. The classifier's move, therefore, consists of some decision which indicates what class it "thinks" nature has selected.

Suppose that in a game between nature and the classifier, nature selects class ω_i, $i = 1,...,M$, and produces a pattern **x**. The probability that **x** comes from ω_i is written as $P(\omega_i \mid \mathbf{x})$ and is called the ***a posteriori*** (or ***posterior***) probability. If the classifier decides that **x** came from ω_j when it actually came from ω_i, it incurs a loss equal to L_{ij} . Since pattern **x** may belong to any of the *M* classes under consideration, the expected loss incurred in assigning observation x to class ω_i is given by

$$r_j(\mathbf{x}) = \sum_{i=1}^{M} L_{ij} P(\omega_i \mid \mathbf{x}) \tag{4.1}$$

which is often referred to as the ***conditional average risk*** or *loss* in decision theory.

The classifier has *M* possible categories to choose from for each pattern given by nature. If it computes the quantities $r_1(\mathbf{x}), r_2(\mathbf{x}), \ldots, r_M(\mathbf{x})$, for each **x**, and assigns each pattern to the class with the smallest conditional loss, it is clear that the total expected loss with respect to all decisions will also be minimized. The classifier which minimizes the total expected loss is called the ***Bayes classifier***. From a statistical point of view, the Bayes classifier represents the optimum measure of performance.

Using Bayes's formula,

$$P(\omega_i \mid \mathbf{x}) = \frac{p(\mathbf{x} \mid \omega_i) P(\omega_i)}{p(\mathbf{x})} \tag{4.2}$$

we may express Eq. (4.1) in the form:

$$r_j(\mathbf{x}) = \frac{1}{p(\mathbf{x})} \sum_{i=1}^{M} L_{ij}\, p(\mathbf{x} \mid \omega_i) P(\omega_i) \tag{4.3}$$

where the *class-conditional probability density function* $p(\mathbf{x}\,|\,\omega_i)$ is called the ***likelihood*** of class ω_i with respect to **x,** and the ***probability density function*** (pdf) $p(\mathbf{x})$ is called the ***evidence***, and for which we have

$$p(\mathbf{x}) = \sum_{i=1}^{M} p(\mathbf{x}\,|\,\omega_i)P(\omega_i) \tag{4.4}$$

Since 1/*p*(**x**) is a common factor in the evaluation of $r_j(\mathbf{x}),\ j = 1,2,\ldots, M$, it may be dropped from Eq. (4.3). The expression for the average loss then reduces to

$$r_j(\mathbf{x}) = \sum_{i=1}^{M} L_{ij}\, p(\mathbf{x}\,|\,\omega_i)P(\omega_i) \tag{4.5}$$

In the general multiclass case, a pattern **x** is assigned to class ω_i if $r_i(\mathbf{x}) < r_j(\mathbf{x})$ for $j = 1,\ldots,M;\ j \neq i$; in other words, **x** is assigned to class ω_i

$$\sum_{k=1}^{M} L_{ki}\, p(\mathbf{x}\,|\,\omega_k)P(\omega_k) < \sum_{q=1}^{M} L_{qj}\, p(\mathbf{x}\,|\,\omega_q)P(\omega_q), \quad j{=}1,2,\ldots,M;\ j \neq i \tag{4.6}$$

In most pattern recognition problems, the loss is zero for correct decisions, and it is the same for all erroneous decisions. Under these conditions, the loss function may be expressed as

$$L_{ij} = 1 - \delta_{ij} \tag{4.7}$$

where δ_{ij} =1 when *i* = *j* and δ_{ij} =0 when *i≠j*. This equation indicates a normalized loss of unity for incorrect classifications and no loss for correct classification of a pattern. Substituting Eq. (4.7) into Eq. (4.5) yields

$$r_j(\mathbf{x}) = \sum_{i=1}^{M} (1-\delta_{ij})\, p(\mathbf{x}\,|\,\omega_i)P(\omega_i) = p(\mathbf{x}) - p(\mathbf{x}\,|\,\omega_j)P(\omega_j) \tag{4.8}$$

The ***Bayes classifier*** assigns a particular pattern **x** to class ω_i if

$$p(\mathbf{x}\,|\,\omega_i)P(\omega_i) > p(\mathbf{x}\,|\,\omega_j)P(\omega_j), \qquad j{=}1,2,\ldots,M;\ \ j \neq i \tag{4.9}$$

It is noted that the ***Bayes decision rule*** of Eq. (4.9) is the implementation of the ***decision*** (or ***discriminant***) ***functions*** [7-9], [11]:

$$d_i(\mathbf{x}) = p(\mathbf{x}\,|\,\omega_i)P(\omega_i), \qquad i{=}1,2,\ldots,M \tag{4.10}$$

where a pattern **x** is assigned to class ω_i if for that pattern $d_i(\mathbf{x}) > d_j(\mathbf{x})$ for all $j \neq i$.

An expression that is equivalent to Eq. (4.10) but does note require explicit knowledge of $p(\mathbf{x}\,|\,\omega_i)$ or $\mathrm{P}(\omega_\mathrm{i})$ is obtained upon substitution of Eq. (4.2) into Eq. (4.10). Performing this substitution yields

$$d_i(\mathbf{x}) = P(\omega_i\,|\,\mathbf{x})\,p(\mathbf{x}), \quad i=1,2,...,M \tag{4.11}$$

However, since $p(\mathbf{x})$ does not depend on i it may be dropped, yielding the decision functions:

$$d_i(\mathbf{x}) = P(\omega_i\,|\,\mathbf{x}), \quad i=1,2,...,M \tag{4.12}$$

Equations (4.10) and (4.12) provide two alternative, yet equivalent, approaches to the same problem. Since estimation of the ***a priori probabilities*** $P(\omega_i),\ i=1,2,...,M$, normally presents no difficulties, the basic difference between these two formulations lies in the use of $p(\mathbf{x}\,|\,\omega_i)$ versus $P(\omega_i\,|\,\mathbf{x})$.

4.1.1.2 Bayesian Classification for Normal Distribution of Patterns

When the data (patterns) is assumed to have a normal distribution, i.e., the probability density functions $p(\mathbf{x}\,|\,\omega_i)$ are ***multivariate normal*** (***Gaussian model***) [7-9], [11], [33] the Bayes classifier derived in the preceding section results in some interesting and familiar decision functions. Because of its analytical tractability, the multivariate normal density function has received considerable attention. Furthermore, it represents an appropriate model for many important practical applications.

Using a Gaussian model, the class-boundaries can be characterized by the ***mean vector*** (or ***class reference vector***) and the ***covariance matrix***, having the form of hyper-ellipsoids or hyper-spheres positioned appropriately in the feature space . The above concepts can be developed in a mathematical framework as follows :

Suppose that M mean vectors $\boldsymbol{\mu}_1, \boldsymbol{\mu}_2, ..., \boldsymbol{\mu}_M$ are given with $\boldsymbol{\mu}_i$ associated with the pattern class ω_i. Let us consider M pattern classes governed by the multivariate normal density functions

$$p(\mathbf{x}\,|\,\omega_i) = \frac{1}{(2\pi)^{N/2}\left|\boldsymbol{\Sigma}_i\right|^{1/2}} \exp\left[-\tfrac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T \boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i)\right],\ i=1,2,...,M \tag{4.13}$$

where each density is completely specified by its ***mean vector*** $\boldsymbol{\mu}_i$ and ***covariance matrix*** $\boldsymbol{\Sigma}_i$, which are defined as

$$\boldsymbol{\mu}_i = E_i\{\mathbf{x}\} = \frac{1}{N_i}\sum_{j=1}^{N_i}\mathbf{x}_{ij} \tag{4.14}$$

and

$$\boldsymbol{\Sigma}_i = E_i\{(\mathbf{x}-\boldsymbol{\mu}_i)(\mathbf{x}-\boldsymbol{\mu}_i)^T\} = \frac{1}{N_i}\sum_{j=1}^{N_i}(\mathbf{x}_{ij}-\boldsymbol{\mu}_i)(\mathbf{x}_{ij}-\boldsymbol{\mu}_i)^T \tag{4.15}$$

where $E_i\{\cdot\}$ denotes the expectation operator over the patterns of class ω_i, N_i denotes the number of patterns in class ω_i, and $\mathbf{x}_{ij}$ represents the *jth* pattern in the *ith* class.

In Eq. (4.13), N is the dimensionality (length) of the pattern vectors and $|\boldsymbol{\Sigma}_i|$ indicates the determinant of matrix $\boldsymbol{\Sigma}_i$.

Based on the Equation (4.13) the calculation of the ***Bayesian classification rule***

$$p(\mathbf{x}\,|\,\omega_i)P(\omega_i) > p(\mathbf{x}\,|\,\omega_j)P(\omega_j), \qquad j=1,2,...,M;\ \ j \neq i$$

can be greatly simplified by taking the logarithm of both sides, yielding an expression of the form

$$\ln P(\omega_i)\,p(\mathbf{x}\,|\,\omega_i) = \ln P(\omega_i) - \frac{N}{2}\ln 2\pi - \frac{1}{2}\ln|\boldsymbol{\Sigma}_i| - \frac{1}{2}(\mathbf{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) \tag{4.16}$$

From this, we can define the ***squared-Bayesian "distance"*** by dropping the constant expression of $\ln 2\pi$ and grouping the other terms together as follows:

$$B_i(\mathbf{x}) = \frac{1}{2}h_i(\mathbf{x}) - Q_i \tag{4.17}$$

where the pattern-independent terms are

$$Q_i = \ln P(\omega_i) - \frac{1}{2}\ln|\boldsymbol{\Sigma}_i| \tag{4.18}$$

and the ***squared-Mahalanobis distance*** (i.e., the squared statistical distance from the mean) is given by

$$h_i(\mathbf{x}) = (\mathbf{x}-\boldsymbol{\mu}_i)^T\boldsymbol{\Sigma}_i^{-1}(\mathbf{x}-\boldsymbol{\mu}_i) \tag{4.19}$$

Note that the true likelihood can be reconstructed from the ***Bayesian distance*** as follows:

$$P(\omega_i)\,p(\mathbf{x}\,|\,\omega_i) = \frac{1}{(2\pi)^{N/2}}\exp\left[-D_i(\mathbf{x})\right] \tag{4.20}$$

where $D_i(\mathbf{x})$ is ***Euclidean distance*** between two points given by:

$$D_i(\mathbf{x}) = (\mathbf{x}-\boldsymbol{\mu}_i)^T(\mathbf{x}-\boldsymbol{\mu}_i)$$

The decision rule can then be rewritten as follows: assign $\mathbf{x}$ to the class with the smallest Bayesian distance, i.e.,

$$B_i(\mathbf{x}) < B_j(\mathbf{x})\ ,\ \ j = 1,2,\ldots M;\ \ \ j \neq i \tag{4.21}$$

4.1.1.3 Parameter Estimation

In Subsections 4.1.1.1 and 4.1.1.2 we saw how we could design an optimal classifier if we knew the prior probabilities $P(\omega_i)$ and class-conditional densities $p(\mathbf{x}\,|\,\omega_i)$. Unfortunately, in pattern recognition applications we rarely, if ever, have this kind of complete knowledge about the probabilities structure of the problem. In a typical case we merely have some vague, general knowledge about the situation, together with a number of ***design samples*** or ***training data*** **–** particular representatives of the patterns we want to classify. The problem then, is to find some way to use this information to design or train the classifier.

One approach to this problem is to use the samples to estimate the unknown probabilities and probability densities, and then use the resulting estimates as if they were the true values. In typical supervised pattern classification problems, the estimation of the prior probabilities presents no serious difficulties. However, estimation of the class-conditional densities is quite another matter. The number of available samples always seems too small, and serious problems arise when the dimensionality of the feature vectror **x** is large. If we know the number of parameters in advance and our general knowledge about the problem permits us to parameterize the conditional densities, then the severity of these problems can be reduced significantly. Suppose, for example, that we can reasonably assume that $p(\mathbf{x}\,|\,\omega_i)$ is a normal density with mean $\boldsymbol{\mu}_i$ and covariance matrix $\boldsymbol{\Sigma}_i$, although we do not know the exact values of these quantities. This knowledge simplifies the problem from one of estimating an unknown function $p(\mathbf{x}\,|\,\omega_i)$ to one of estimating the parameters $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$.

The problem of parameter estimation is a classical one in statistics, and it can be approached in several ways [9], [11-13], [33-34]. In this thesis we shall use two common and reasonable procedures, namely, ***maximum likelihood*** (ML) estimation and ***maximum a posteriori probability*** (MAP) estimation [9], [11].

Maximum-Likelihood Parameter Estimation

Maximum-likelihood estimation methods have a number of attractive attributes. First they nearly always have good convergence properties as the number of training samples increases. Furthermore, maximum-likelihood estimation often can be simpler than alternative methods, such as Bayesian techniques [9].

Let us consider an *M* class problem with feature vectors distributing according to $p(\mathbf{x} \mid \omega_i),\ i = 1, 2, ..., M$. We assume that $p(\mathbf{x} \mid \omega_i)$ has a known parametric form, and is therefore determined uniquely by the value of a parameter vector $\boldsymbol{\theta}_i$. For example, we might have $p(\mathbf{x} \mid \omega_i) \sim N(\boldsymbol{\mu}_i, \boldsymbol{\Sigma}_i)$, where $\boldsymbol{\theta}_i$ consists of the components of $\boldsymbol{\mu}_i$ and $\boldsymbol{\Sigma}_i$. To show the dependence of $p(\mathbf{x} \mid \omega_i)$ on $\boldsymbol{\theta}_i$ explicity, we write $p(\mathbf{x} \mid \omega_i)$ as $p(\mathbf{x} \mid \omega_i; \boldsymbol{\theta}_i)$. Our goal is to estimate the unknown parameters $\boldsymbol{\theta}_i,\ i = 1, 2, ..., M$ using a set of known feature vectors in each class. If we further assume that data from one class do not affect the parameter estimation of the others, we can formulate the problem independent of classes and simplify our notation. At the end, one has to solve one such problem for each class independently.

Let $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ be random samples drawn from probability density function (pdf) $p(\mathbf{x} \mid \boldsymbol{\theta})$. We form the joint pdf $p(X \mid \boldsymbol{\theta})$, where $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$ is the set of the samples. Assuming statistical independence between the different samples we have

$$p(X \mid \boldsymbol{\theta}) \equiv p(\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n \mid \boldsymbol{\theta}) = \prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\theta}) \tag{4.22}$$

The above is a function of $\boldsymbol{\theta}$ and it is known as the likelihood function of $\boldsymbol{\theta}$ with respect to *X*. The maximum likelihood (ML) method estimates $\boldsymbol{\theta}$ so that the likelihood function takes its maximum value, i.e.

$$\hat{\boldsymbol{\theta}}_{ML} = \arg\max_{\boldsymbol{\theta}} \prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\theta}) \tag{4.23}$$

A necessary condition that $\hat{\boldsymbol{\theta}}_{ML}$ must satisfy in order to be a maximum is

$$\frac{\partial \prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0 \tag{4.24}$$

For analytical purposes, it is usually easier to work with the logarithm of the likelihood than with the likelihood itself. Because the logarithm is monotonically increasing, the $\hat{\boldsymbol{\theta}}_{ML}$ that maximizes the log-likelihood also maximizes the likelihood. We define the ***log-likelihood function*** as

$$L(\boldsymbol{\theta}) \equiv \ln \prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\theta}) = \sum_{k=1}^{n} \ln p(\mathbf{x}_k \mid \boldsymbol{\theta}) \tag{4.25}$$

and Eq. (4.24) is equivalent to

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{k=1}^{n} \frac{\partial \ln p(\mathbf{x}_k \mid \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \sum_{k=1}^{n} \frac{1}{p(\mathbf{x}_k \mid \boldsymbol{\theta})} \frac{\partial p(\mathbf{x}_k \mid \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = 0 \tag{4.26}$$

Applications of maximum-likelihood parameter estimation in specific cases are given in **Appendix A**.

Maximum Aposteriori Probability Estimation

For the derivation of the ML estimator we considered $\boldsymbol{\theta}$ as an unknown parameter. In this subsection we will consider it is a random vector, and we will estimate its value on the condition that samples $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ have occurred. Let $X = \{\mathbf{x}_1, ..., \mathbf{x}_n\}$. Our starting point is $p(\boldsymbol{\theta} \mid X)$. From our familiar Bayes theorem we have

$$p(\boldsymbol{\theta} \mid X) = \frac{p(\boldsymbol{\theta})\, p(X \mid \boldsymbol{\theta})}{p(X)} \tag{4.27}$$

The maximum aposteriori probability (MAP) estimator $\hat{\boldsymbol{\theta}}_{MAP}$ is defined at the point where $p(\boldsymbol{\theta} \mid X)$ becomes maximum, i.e.,

$$\boldsymbol{\theta}_{MAP}: \ \frac{\partial}{\partial \boldsymbol{\theta}} p(\boldsymbol{\theta} \mid X) = 0 \ \text{ or } \ \frac{\partial}{\partial \boldsymbol{\theta}} \left[p(\boldsymbol{\theta})\, p(X \mid \boldsymbol{\theta}) \right] = 0 \tag{4.28}$$

4.1.2 The Neural Networks Pattern Recognition Approach

This approach uses ***Artificial Neural Networks*** (ANNs or simply NNs) for classifying data [8-10], [32], [43], [112]. ANNs reveal (encode) unknown nonlinear relations of the data through training.

Artificial neural networks originated from idea to model mathematically human intellectual abilities by biologically plausible engineering designs. Meant to be massivelly parallel computational schemes resembling a real brain, NNs evolved to become a valuable classification tool with a significant influence on pattern recognition theory and practice. Neural networks provide a reasonable and powerful alternative to conventional classifiers. Potential benefits of neural networks extend beyond the high computation rates provided by massive parallelism. Neural networks are natural classifiers with significant and desirable characteristics such as resistance to noise, tolerance to distorted images/patterns (ability to generalize) and superior ability to recognize partially occluded or degraded images. Neural networks are often used as base classifiers in multiple classifiers systems [20-21].

Literature on NNs is excessive and continuously growing. Many publications such as textbooks and monographs [10], [32], [34], [41-43], [113-116], paper collections [112], [117-120] and so on, discuss NNs at various theoretical and algorithmic depths. In addition, a unified view of neural and statistical pattern recognition approaches is referred in [10], [34], [116-117].

Consider an *N*-dimensional pattern recognition problem with *M*-classes $\omega_1, \omega_2, \ldots, \omega_M$. A neural network obtains a feature vector $\mathbf{x} = (x_1, x_2, ..., x_N)^T \in \mathbb{R}^N$ at its input, and produces values for the *M* discriminant functions $d_1(\mathbf{x}),...,d_M(\mathbf{x})$ at its output. Typically NNs are trained to minimize the squared error on a labeled training set $\mathbf{Z} = \{\mathbf{z}_1,...,\mathbf{z}_n\}$, $\mathbf{z}_j \in \mathbb{R}^N$, and $l(\mathbf{z}_j) \in \Omega = \{\omega_1, \omega_2, ..., \omega_M\}$

$$E = \frac{1}{2}\sum_{j=1}^{n}\sum_{i=1}^{M}\left[d_i(\mathbf{z}_j) - I(\omega_i, l(\mathbf{z}_j))\right]^2 \tag{4.29}$$

where $I(\omega_i, l(\mathbf{z}_j))$ is an indicator function taking value 1 if the label of $\mathbf{z}_j$ is ω_i and 0 otherwise. It has been shown that the set of discriminant functions obtained by minimizing Eq. (4.29) approach the posterior probabilities for the classes for data size $n \to \infty$ [122-123]; that is,

$$\lim_{n\to\infty} d_i(\mathbf{x}) = P(\omega_i \mid \mathbf{x}), \qquad \mathbf{x} \in \mathbb{R}^{\mathbf{N}} \tag{4.30}$$

This result was brought to light in connection with NNs, but in fact, it holds for any classifier that can approximate an arbritary discriminant function with a specified precision.

Various NN training protocols and algorithms have been developed, and these have been the key to the success of NN classifiers. In this Ph.D. thesis, we are especially interested in multilayer perceptron, ***learning vector quantization*** (LVQ) and ***Hopfield autoassociative memory neural networks*** for classification purposes [10], [32], [41-43], [113], [115-116], [124-125]. In the next three subsections we describe these types of neural nets. In addition to, we have used the ***high order neural networks*** (HONNs) [132-136] for feature extraction purposes [28], [137-138]. The theoretical framework of HONNs is introduced in Chapter 6.

4.1.2.1 Multilayer Perceptron Feed-forward Neural Network

One type of ANN system is based on a unit called a ***perceptron*** [42], [126] illustrated in Figure 4.2. A perceptron takes a vector of real-valued inputs, calculates a linear combination of these inputs, then outputs a 1 if the result is greater than some threshold and –1 otherwise. More precisely, given inputs $x_1,...,x_N$, the output

$$\phi\left(x_1,...,x_N\right)=\begin{cases} 1\ if\ \ w_0+w_1x_1+w_2x_2+...+w_Nx_N>0 \\ -1 \text{ otherwise} \end{cases} \tag{4.31}$$

where each w_i is a real-valued constant, or weight, that determines the contribution of input x_i to the perceptron output. Notice the quantity $\left(-w_0\right)$ is a threshold that the weighted combination of inputs $w_1x_1+...+w_Nx_N$ must surpass in order for the perceptron to output a 1.

By connecting perceptrons we can design an NN structure called the ***multilayer perceptron*** (MLP). This is a feedforward structure because the output of the input layer and all intermediate layers is submited only to the higher layer. The generic model of a feedforward MLP classifier is shown in Figure 4.3. Here “layer” means a layer of perceptrons. There are three distinct types of layers: the input layer, the hidden layer(s) and the output layer. The connections between the neurons (i.e. perceptrons) of adjacent layers relay the output signals from one layer to the next. The input layer receives the input information (i.e. the feature vector $\mathbf{x}=(x_1,x_2,...,x_N)^T\ \in\mathbb{R}^N$) and distributes the information to the next processing layer (the first hidden layer). The number of the neurons in the input layer equals to the dimension of the feature vector. The hidden and output layers process the incoming signals by amplifying or attenuating or inhibiting the signals through weighting factors.

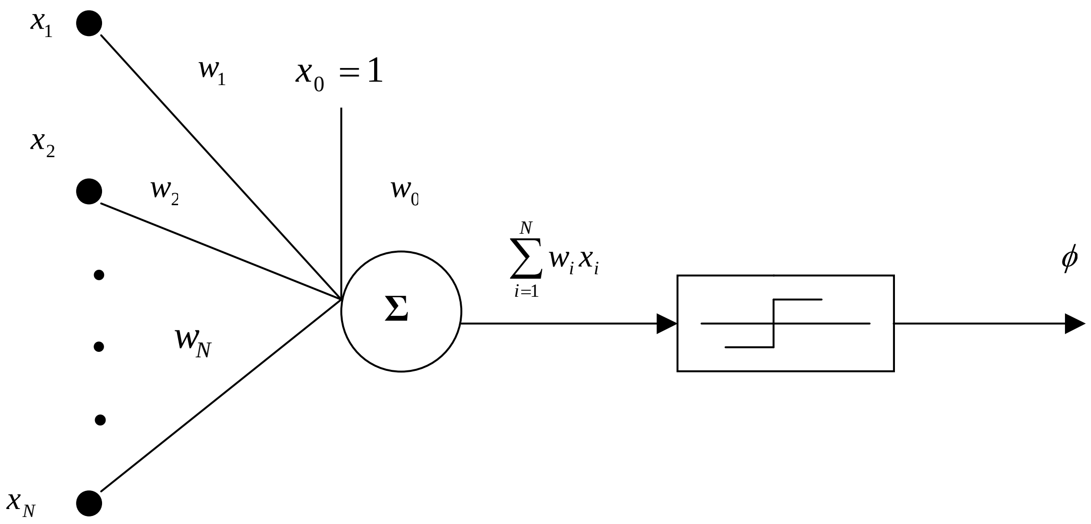


Figure 4.2 A perceptron

Except for the input layer neurons, the network input to each neuron is the sum of the weighted outputs of the neurons in the previous layer. The number of neurons in the output layer is determined by the number of classes under investigation (i.e. M outputs $z_1,...,z_M$ or M discriminant functions $d_1(\mathbf{x}),...,d_M(\mathbf{x})$ for N-dimensional pattern recognition problem with M-classes $\omega_1, \omega_2, ..., \omega_M$). The number of hidden layers and the number of neurons in each hidden layer depend on specific application.

The most critical part of an ANN-based model is to train the network. The most widely studied and used training algorithm is the so-called ***backpropagation*** learning algorithm [41-43], [113, [115-116], which is robust and reliable. The problem of neural network training is to devise a method of updating the representative weights that minimize the error. It is essentially an optimization problem. The behaviour of feedforward network and learning through least square based updating, e.g., backpropagation rules can be followed more easily corresponding to changes of parameters, e.g., number of hidden layers, number of nodes in a hidden layer, change in the type of activation function, learning rate, etc. However, the updating of the weights has been done in this thesis by ***Levenberg – Marquardt algorithm*** [17],

[125]. Levenberg – Marquardt algorithm performs much better with some knowledge of the process so that quick convergence is obtained with a very small error.

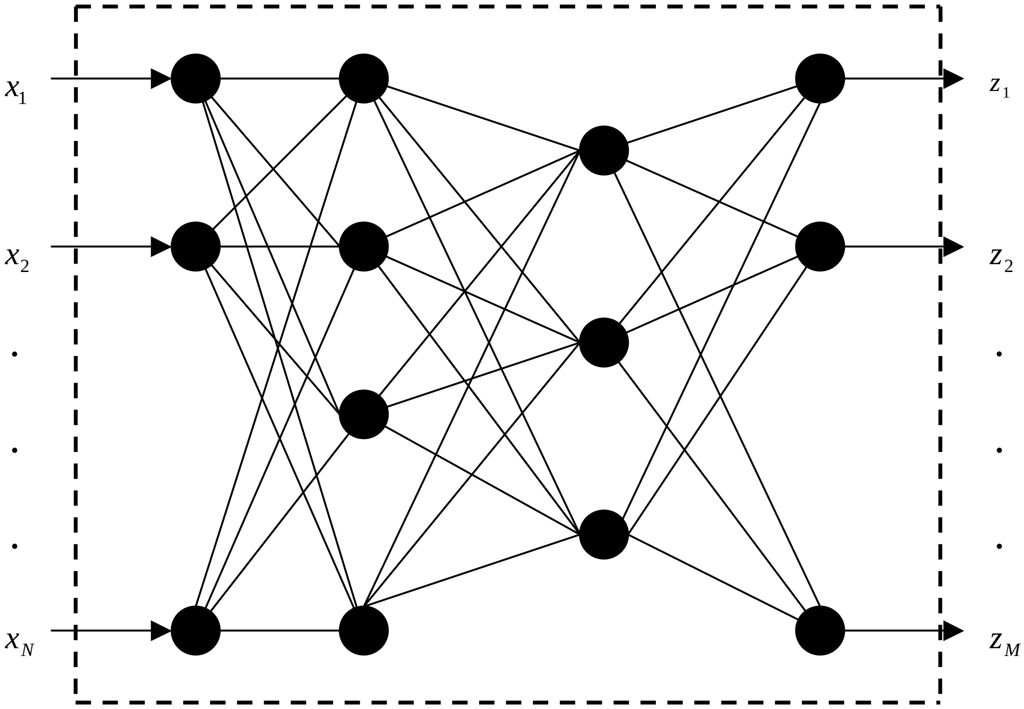


Figure 4.3 A generic model of an MLP classifier

The structure of the network has been given in Figure 4.4. In this figure, $v_{ij},\ i=1,...,N,\ j=1,...,P$ and $w_{jk},\ j=1,...,P,\ k=1,...,M$ denote the weights for the successive layers. The basic purpose of training a network is to optimize v_{ij} and w_{jk} corresponding to a particular set of input – output training pattern. The responses at the hidden nodes $y_j,\ j=1,2,...,P$ are calculated by evaluating the contributions from all the input nodes through a nonlinear mapping function

$$y_j = f\left[\sum_{i=1}^{N} x_i v_{ij} + \theta_j\right] \qquad (4.32)$$

where the function $f(\bullet)$ given by

$$f = \frac{2}{1+\exp(-2N)} - 1 \qquad (4.33)$$

θ_j is the bias at the *j*th hidden layer node and $(x_1, x_2, ..., x_N)^T \in \mathbb{R}^N$ is the input vector (feature vector). Similarly, $z_k,\ k=1,...,M$ is calculated using

$$z_k = f\left[\sum_{j=1}^{P} b_j w_{jk} + \tau_k\right] \tag{4.34}$$

where τ_k is the bias at the kth output layer node.

A mixture of Gauss-Newton method and gradient descent technique [17], [41- 43], [125] has been used for training the network (i.e. for optimization of the weights, v_{ij} and w_{jk}).

The Levenberg – Marquardt weight update rule is

$$\Delta w = \left(\mathbf{J}^T\mathbf{J} + \mu\mathbf{I}\right)^{-1}\mathbf{J}^T e \tag{4.35}$$

where **J** is the Jacobian matrix of derivatives of each error to each weight, μ is a scalar, **I** is the identity matrix and *e* is an error vector. If the scalar μ is very large, the above expression approximates gradient descent, while if it is small the above becomes Gauss – Newton method. The Gauss – Newton method is faster and more accurate near an error minimum. After each succesful step, i.e., if the error continues to decrease, μ is decreased by one-step and vice-versa. Training continues until the error goal is met and the minimum error gradient occurs.

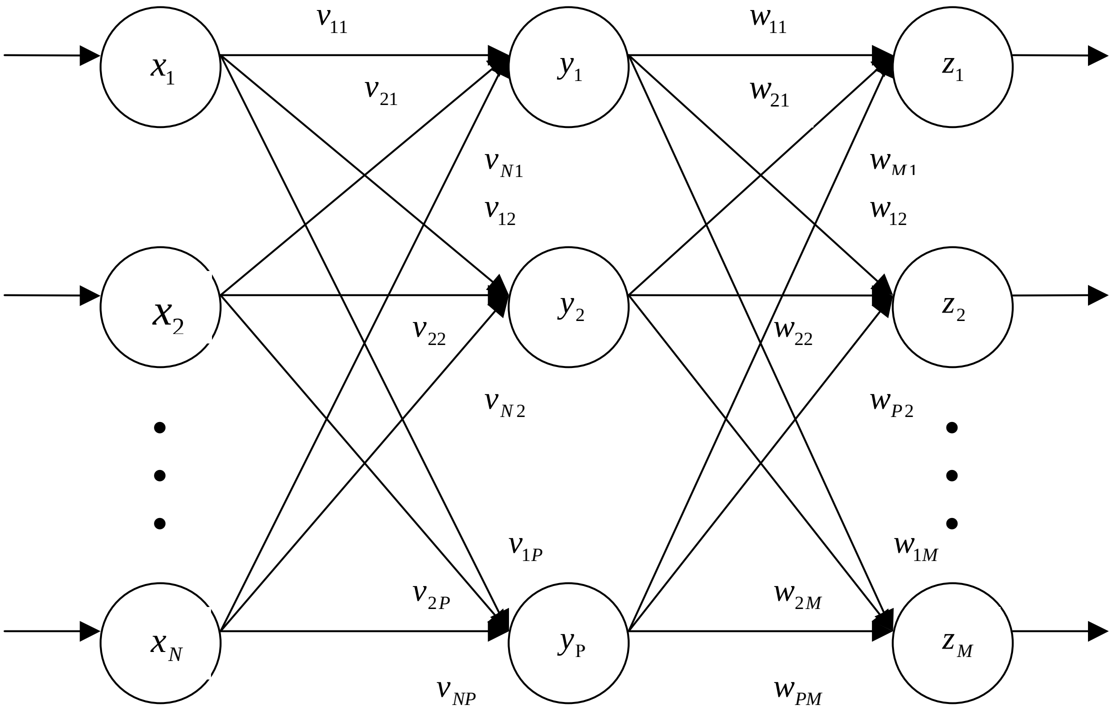


Figure 4.4. Weight vectors in the MLP feed-forward network

4.1.2.2 Learning Vector Quantization

Self-organizing map (SOM) is an unsupervised neural network model developed by Kohonen [124-125]. Unsupervised training is one of the two types of training in Neural Networks theory. In this kind of training, there is no training and testing phase, but all samples are used for training.

The SOM can be used as a pattern classifier. However, the classification accuracy of the SOM can be significantly increased by using the supervised learning vector quantization (LVQ) algorithm [113], [125].

The architecture of a LVQ neural network is shown in Figure 4.5. Each neuron in the first layer (competitive layer) ,S^1, of the LVQ network learns a weight vector $\mathbf{w}_m = (w_{1m}, w_{2m}, \ldots, w_{nm})$ (often referred to as a ***reference*** or ***codebook*** vector), which allows the network to classify a region of the input space. During the training phase, the distance between each input vector and the weight vector of each neuron is calculated. The norm of each distance $\|\mathbf{x} - \mathbf{w}_m\|$ is computed and the output of the first layer of the LVQ is a vector with 1 corresponding to the neuron whose weight vector is closest to the input vector, and 0's everywhere else.

Thus the LVQ network behaves exactly like the competitive network [113], [125]. However, there is a difference in interpretation, because in the competitive network the neuron with nonzero output (winning neuron) indicates which class the input vector belongs to. For the LVQ network, the winning neuron indicates a subclass, rather than a class. There may be several different neurons (subclasses) that make up each class.

The second layer, S^2, of the LVQ network is used to combine subclasses into a single class. This is done with the $\mathbf{W}^2$ matrix . The columns of $\mathbf{W}^2$ represent subclasses , and the rows represent classes. $\mathbf{W}^2$ has a single 1 in each column , with the other neurons set to zero. The row in which the 1 occurs indicates which class the appropriate subclass belongs to.

$$(w_{km}^2 = 1) \Rightarrow \text{subclass } m \text{ is a part of class}$$

During training, the process of combining subclasses to form a class allows the LVQ network to create the decision boundaries of the Bayes classifier. It is assumed that a set of training patterns with known classifications is provided, along with an initial distribution of reference vectors (each of which represents a known classification).

The learning in the LVQ network combines competitive learning with supervision. As with all supervised learning algorithms, it requires a set of examples $\{x_1, t_1\}, \{x_2, t_2\}, \ldots, \{x_n, t_n\}$, where x_i, t_i are input and target values respectively and $i = 1, \ldots, n$. Each target vector must contain only zeros except for a single 1. The row in which the 1 appears indicates the class to which the input vector belongs.

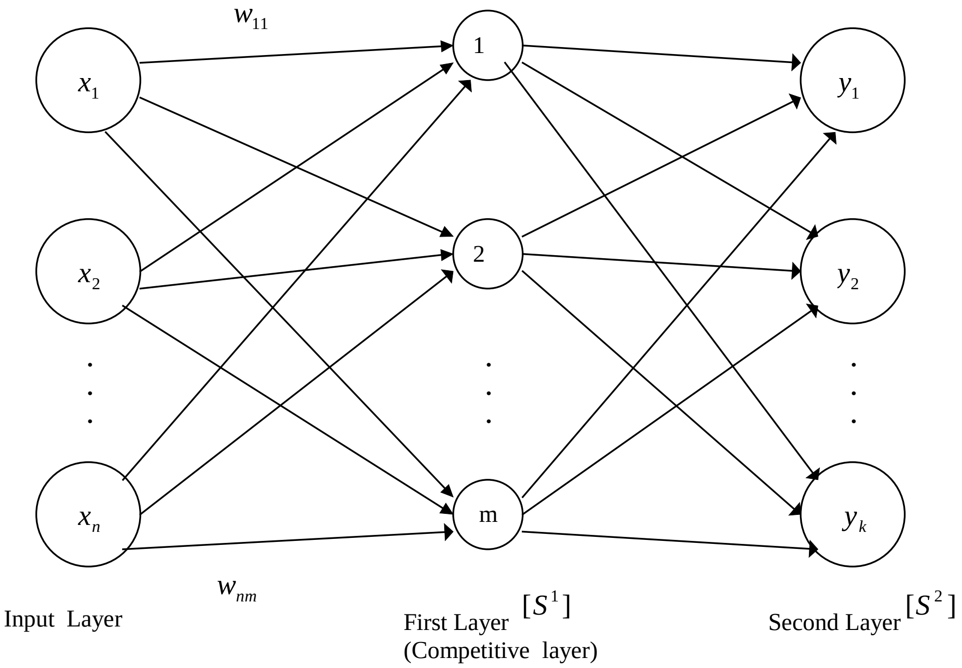


Figure 4.5. The architecture of LVQ neural network.

The LVQ learning rule proceeds as follows [125]. First, the weight matrix $\mathbf{W}^1$ and the learning rate α are initialized randomly. At each iteration, an input vector $\mathbf{x}$ is presented to the network and the distance from $\mathbf{x}$ to each weight vector is computed. The neurons of first layer compete, neuron J wins the competition, and the J th element of the output of the first layer $\mathbf{a}^1$ is set to 1 at the J th entry. Then $\mathbf{a}^1$ is multiplied by $\mathbf{W}^2$ matrix to get the final output $\mathbf{a}^2$, which also has only one nonzero element *k*, indicating that $\mathbf{x}$ is being assigned to class *k*.

The Kohonen rule is used to update the weight vector $\mathbf{w}_J$ of the winning neuron in two ways. First, if $\mathbf{x}$ is classified correctly ($\mathrm{a}_k{}^2 = t_k = 1$), then the weight vector $\mathbf{w}_J$ is moved toward $\mathbf{x}$.

$$\mathbf{w}_J{}^1(new) = \mathbf{w}_J{}^1(old) + \alpha[\mathbf{x} - \mathbf{w}_J{}^1(old)] \qquad (4.36)$$

Second, if $\mathbf{x}$ was classified incorrectly ($\mathrm{a}_k{}^2 = 1 \neq t_k = 0$), then it is known that the wrong competitive neuron won the competition, and therefore the weight vector is moved away from $\mathbf{x}$.

$$\mathbf{w}_J{}^1(new) = \mathbf{w}_J{}^1(old) - \alpha[\mathbf{x} - \mathbf{w}_J{}^1(old)] \qquad (4.37)$$

The result is that each competitive neuron moves toward vectors that fall into the class for which it forms a subclass and away from vectors that fall into other classes.

Afterwards, learning rate α is reduced and is checked the stopping criterion, which may be to make more iterations than a fixed number, or if the learning rate reaching a sufficiently small value.

4.1.2.3 Hopfield Networks and Neural Associative Memories

The publication of Hopfield's seminal papers [127 – 128] started the modern era in neural networks. His proposed networks are known as ***Hopfield networks***. Hopfield networks have found many useful applications, espesially in associative memory and optimization problems.

One of the most useful and most investigated areas of applications of neural networks addresses implementations of associative memories (AMs) [41], [113], [127-128]. The function of an AM is to recall a complete set of previously stored information, called a "memory", when the AM is initialized with a subset of the memory, called a "key". More specifically, the AM is designed to store a set of vectors, say **x**, in such a way that a stimulus, say **y=x+***d***x**, evokes the output **x** for sufficiently small *d***x**. If *d***x** is considered to constitute either noise or perturbations, then the AM is performing the functions of noise suppression or error correction, respectively.

Hopfield has presented continuous time and discrete time systems that are capable of implementing AMs [127 – 128]. In addition, several other investigators have addressed the *analysis* of various types of continuous time and discrete time neural networks [130]. Effective and general *synthesis* procedures for such systems have been presented [129 – 130].

Discrete Hopfield Networks

The Hopfield network is a single-layer feedback network; its detailed network configuration is shown in Figure 4.6.

When operated in discrete-time fashion, it is called a discrete Hopfield network and its structures as a single-layer feedback network can also be termed ***recurrent***. When a single-layer recurrent network performs a sequential updating process, an input pattern is first applied to the network, and the network’s output is initialized accordingly. Then, the initializing pattern is removed and the initialized output becomes the new, updated input through the feedback connections. The first update input forces the first update output; this in turn acts as the second updated input through the feedback links and produces the second update output. The transition process continues until no new, updated responses are produced and the network has reached its equilibrium.

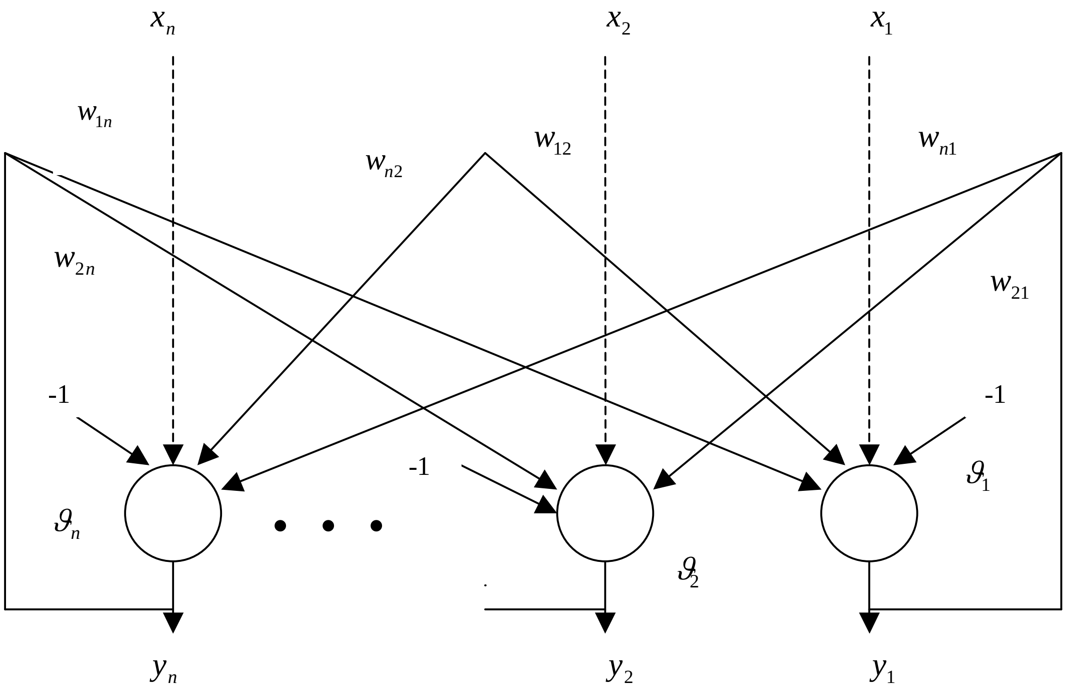


Figure 4.6. Structure of the Hopfield Network

Consider the Hopfield network shown in Figure. 4.6. Each node has an external input x_j and a threshold θ_j, where j=1,2,…,n. It is important to point out that there is no self-feedback in a Hopfield network. The jth node output is connected to each of the other node's inputs through a multiplicative weight w_{ij} for $i = 1,2,...,n, i \neq j$; that is, $w_{ii} = 0$ for i=1,2,…,n. The evolving rule (or ***update rule***) for each node in a discrete Hopfield network is

$$y_i^{(k+1)} = \text{sgn}(\sum_{\substack{j=1 \\ j\neq i}}^{n} w_{ij} y_j^{(k)} + \theta_i), \qquad i=1,2,\ldots,n \tag{4.38}$$

where sgn(.) is the signum function and the superscript k denotes the index of recursive update.

Associative memories

An associative memory [41], [113], [129-131] can store as set of patterns as memories. When the associative memory is presented with a *key pattern*, it responds by producing whichever one of the stored patters most closely resembles or relates to the key pattern. Hence, the recall is through association of the key pattern with the information memorized. Such memories are also called ***content-addressable memories*** in contrast to the traditional ***address-addressable memories*** [41] in digital computers in which a stored pattern (in bytes) is recalled by its address. The basic concept of using Hopfield networks as associative memories is to interpret the system's evolution as a movement of an input pattern most resembling the input pattern.

Two types of associative memories can be distinguished. They are ***autoassociative memory*** and ***heterassociative memory*** [41], [113]. Suppose we have p pairs of vectors $\{(\mathbf{x}^1,\mathbf{y}^1),(\mathbf{x}^2,\mathbf{y}^2),...,(\mathbf{x}^p,\mathbf{y}^p)\}$ with $\mathbf{x}^i \in \mathbb{R}^n$ and $\mathbf{y}^i \in \mathbb{R}^m$. In the autoassociative memory, it is assumed that $\mathbf{x}^i = \mathbf{y}^i$ and that the network implements a mapping Φ of $\mathbb{R}^n$ to $\mathbb{R}^n$ such that $\Phi(\mathbf{x}^i) = \mathbf{x}^i$. If some arbitrary pattern $\mathbf{x}$ is *closer* to $\mathbf{x}^i$ than to any other $\mathbf{x}^j, j = 1,2,..., p, j \neq i,$ then $\Phi(\mathbf{x}) = \mathbf{x}^i$; that is the network will produce the stored pattern $\mathbf{x}^i$ when the key pattern $\mathbf{x}$ is presented as input. In the heteroassociative memory, the network implements a mapping Φ of $\mathbb{R}^n$ to $\mathbb{R}^m$ such that $\Phi(\mathbf{x}^i) = \mathbf{y}^i$, and if some arbitrary pattern **x** is *closer* to $\mathbf{x}^i$ than to any other $\mathbf{x}^j, j = 1,2,..., p, j \neq i,$ then $\Phi(\mathbf{x}) = \mathbf{y}^i$. In the above, "closer" means with respect to some proper distance measure, for example, the Euclidean distance or the Hamming distance (HD). The ***Euclidean distance*** d between two vectors $\mathbf{x} = (x_1, x_2,..., x_n)^T$ and $\mathbf{x}' = (x_1^{'}, x_2^{'},..., x_n^{'})^T$ is defined as $d = [(x_1 - x_1^{'})^2 + ... + (x_n - x_n^{'})^2]^{1/2}$,and the ***Hamming distance*** is defined as the number of mismatched components of **x** and $\mathbf{x}'$ vectors [41], [113]. More specifically,

$$HD(\mathbf{x},\mathbf{x}') = \begin{cases} \sum_{i=1}^{n} \left| x_i - x_i^{'} \right| & \text{if } x_i, x_i^{'} \in \{0,1\} \\ \frac{1}{2}\sum_{i=1}^{n} \left| x_i - x_i^{'} \right| & \text{if } x_i, x_i^{'} \in \{-1,1\} \end{cases} \tag{4.39}$$

For example, if $\mathbf{x} = (1,1,0,1)^T$ and $\mathbf{x}' = (0,1,0,0)^T$, then $\mathrm{HD}(\mathbf{x},\mathbf{x}') = 2$. Similarly, if $\mathbf{x} = (1,-1,-1,-1)^T$ and $\mathbf{x}' = (1,1,-1,-1)^T$,then $\mathrm{HD}(\mathbf{x},\mathbf{x}') = 1$.

In a special case where the vectors $\mathbf{x}^i, i = 1,2,..., p$, form an orthonormal set, the associative memory can be defined as

$$\Phi(\mathbf{x}) = \mathbf{W}\mathbf{x} = (\mathbf{y}^1(\mathbf{x}^1)^T + \mathbf{y}^2(\mathbf{x}^2)^T + ... + \mathbf{y}^p(\mathbf{x}^p)^T\,\mathbf{x} \tag{4.40}$$

where **W** can be considered a weight matrix, called a ***cross-correlation*** matrix, of the network. It is easily seen that $\Phi(\mathbf{x}^i) = \mathbf{W}\mathbf{x}^i = \mathbf{y}^i$ since the set of **x** vectors is orthonormal. The associative network with the weight matrix defined as in Eq. (4.40) is called a ***linear associator***.

The linear associators are ***static*** or ***nonrecurrent*** memory networks since they implement a feedforward operation of mapping without a feedback, or recursive update, operation. We shall next introduce the *dynamic* or recurrent memory networks that exhibit dynamic evolution in the sense that they converge to a equilibrium state according to the recursive formula $\mathbf{y}^{(k+1)} = \Phi(\mathbf{x}^{(k)}, \mathbf{y}^{(k)})$, where k is the time step and Φ is a nonlinear mapping in the form of thresholding. Recurrent memory networks, because they threshold the output and recycle the output to input, are able to suppress the output noise at the memory output to produce an improved association.

Recurrent Autoassociative Memory - Hopfield Memory

In this paragraph we introduce ***Hopfield autoassociative memory*** (or ***Hopfield memory*** for short), that is, Hopfield networks utilized as autoassociative memories. A Hopfield memory is able to recover an original stored vector when presented with a probe vector close to it. Here, we focus on discrete Hopfield networks and consider continuous Hopfield networks as the former's hardware implementation with continuous transient responses. In Hopfield memory, is the data *retrieval rule* that is applied asynchronously and stochastically. The remaining problem is how to store data in memory. Assume bipolar binary vectors that need to be stored are $\mathbf{x}^k$ for k =1, 2,…, p. The ***storage algorithm*** for finding the weight matrix is

$$\mathbf{W} = \sum_{k=1}^{p} \mathbf{x}^k (\mathbf{x}^k)^T - p\mathbf{I}\,, \tag{4.41}$$

or

$$w_{ij} = \sum_{k=1}^{p} x_i^k x_j^k, \quad i \neq j;\;\; w_{ii} = 0 \tag{4.42}$$

where $\mathbf{x}^k = (x_1^k, x_2^k, ..., x_n^k)^T$ and **I** is an appropriate identity matrix. If $\mathbf{x}^i$ are unipolar binary vectors, that is, $x_i^k \in \{0,1\}$, then the storage rule is

$$w_{ij} = \sum_{k=1}^{p} (2x_i^k - 1)(2x_j^k - 1), \;\; i \neq j;\, w_{ii} = 0 \tag{4.43}$$

The weight assignment rule in Eq. (4.41) is basically the ***Hebbian learning rule*** [41], [113] with zero initial weights. Hence, this rule is called a *Hebbian-type* learning rule or an ***outer-product learning rule*** [41], [113], [131]. Additional autoassociations can be added to the existing memory at any time by superimposing new, incremental weight matrices. Autoassociations can also be removed by respective weight matrix subtraction. Moreover, the storage rule in Eq. (4.41) is invariant with respect to the sequence of storing patterns and also is invariant under the binary complement operation.

4.1.3 The Fuzzy Pattern Recognition Approach

Fuzzy logic [36-41] provides a mathematical framework to capture uncertainties associated with human cognitive systems, such as thinking and reasoning. Simply, it simulates human thinking which operates more likely on symbols than exact values. In fact, our daily thoughts and communication are full of these symbols or fuzzy expressions.

Pattern Recognition is, by its very nature, an inexact science. To deal with the ambiguity, it is helpful to introduce some “fuzziness” into the formulation of the problem. For example, the boundary between clusters could be fuzzy rather than crisp; that is, a data point could belong to two or more clusters with different degrees of membership. In this way, the formulation is closer to the real-world problem and therefore better performance may be expected. This is the first reason for using fuzzy models for pattern recognition. The second reason is that the optimization of a fuzzy PR formulation may be easier to solve computationally. This is due to the fact that a non-fuzzy model often results in an exhaustive search within a huge space (because some key variables can only take two values 0 and 1), whereas in a fuzzy model all the variables are continuous, so that derivatives can be computed to find the right direction for the search.

A ***fuzzy set*** can be considered as an extension of a classical (“crisp”) set: crisp sets permit only full membership or no membership; fuzzy sets permit partial membership with a certain degree. Indeed, a fuzzy set, say *A*, in a domain *X* is characterized by a ***membership function*** μ_A that takes values in the real interval [0,1]. The domain *U* is called ***universal set*** or (***universe of discourse***). For each $x \in X$, $\mu_A(x)$ gives the ***degree of membership*** of x to the set *A*, i.e., a real number in the range [0,1], where 1 denotes full membership and 0 denotes no membership [36-37].

The fuzzy set concept provides us with an intuitive method of representing one form of uncertainty, *vagueness*, by eliminating the sharp boundary that divides members of the class from non-members. In fuzzy sets, a value is assigned to each element *x* of the universal set *X*. signifying its degree of membership in a particular set with unsharp (fuzzy) boundaries. This is useful in situations where is not possible to draw crisp boundaries in deciding if a person is tall and in observing the shape of a growing animal cell in biology. However, in some decision-making situations such as judging if a defendant is guilty or not guilty, and in most measurements in physical sciences such as measurements of length, area, and weight, classes are defined with sharp boundaries. In the trial example, it is obvious that the group of guilty persons and the group of innocent persons are crisp sets. Since the evidence for trial judgement is rarely perfect and measurement error is unavoidable in most physical sciences, some uncertainty usually prevails. To represent this kind of uncertainty, known as ambiguity, we assign a value in the unit interval [0, 1] to each possible crisp set to which the element in question might belong. This value represents the ***degree of evidence*** or belief or certainty of the element's membership in the set. Such a representation of uncertainty is known as a ***fuzzy measure*** [39-41].

In general, a fuzzy measure is defined by a (set) function

$$g:\ 2^X \to \left[0,1\right] \tag{4.44}$$

which assigns to each crisp subset of a universe of discourse X a number in the unit interval [0, 1], where 2^X is the power set of X. When this number is assigned to a crisp subset $A \in 2^X$, $g\left(A\right)$ represents the degree of evidence or our belief that a given element $x \in X$ (which has not been previously located in any crisp subset of X) belongs to the crisp subset A. Notice that the domain of the function g is the power set 2^X of ***crisp*** subsets of X and not the power set 2^X of fuzzy subsets of X.

Several different measures such as ***belief measures***, ***plausibility measures***, ***necessity measures***, and ***possibility measures*** are referred in literature [39-41]. They are all functions applied to crisp subsets, instead of elements, of a universal set.

Based on the properties of fuzzy measure, ***fuzzy integral***, such as Sugeno and Choquet fuzzy integals [39-41], is an aggregation operator on multi-attribute fuzzy information. Fuzzy integral can be viewed as a fuzzy expectation when compared with statistical expectation, and as a non-linear integral in comparison with Lebesque integral.

It is well known that a combination of many different classifiers can improve classification accuracy. A variety of schemes have been proposed for combining multiple classifiers [20-21]. The ability of fuzzy integrals to combine the results of multiple classifiers has been mentioned in literature [20-21]. In practice, outputs from multiple classifiers are ususally highly correlated. Therefore, it is desirable to assign weights not only to individual classifiers but also to groups of classifiers. This expresses the correlations between different classifiers. Aggregation based on fuzzy integrals possess this valuable property. In such schemes, outputs of base classifiers are fused into a final decision by a fuzzy integral with respect to a fuzzy measure. However, to utilize this property, we need to construct fuzzy measures that express the actual interaction among classifiers with respect to classification performance. The fuzzy measures represent the weights on each group of classifiers. Most often a separate fuzzy measure is defined for each decision class.

In this thesis we focus on fuzzy measures and fuzzy integrals for multiple classifier fusion. A complete mathematical background on fuzzy measures and fuzzy integrals is introduced in Chapter 7. In addition to, a combining multiple classifiers method based upon ***Sugeno fuzzy measure*** and ***Choquet fuzzy integral*** for improving the classification of the individual leads in component quality inspection is tested and compared with other classifier fusion methods.

4.1.4 The Genetic Algorithms Pattern Recognition Approach

Based on the Darwinian survival of the fittest, ***genetic algorithms*** (GA) (or ***evolutionary computing***) are global search and optimization techniques. Inherently parallel, they operate on a set of candidate solutions that formulate a ***population***, whose size is maintained constant. Each solution is usually coded as a binary (or real) string called a ***chromosome***. The chromosomes of the initial population are randomly generated. Each iteration of the GA called a ***generation*** involves three stages:

- The current population is first evaluated and ranked with the aid of a ***fitness function*** (fitness measuring criterion).
- Chromosomes that possess the highest fitness values are probabilistically selected to construct the "parents" pool.
- From the selected "parents", the GA reproduces "children" performing the genetic operations of ***crossover*** and ***mutation***.

The GA terminates when an acceptable solution is found, or when a predetermined number of generations is reached.

The GA is not considered a mathematically guided algorithm. The optima obtained are evolved from generation to generation without stringent mathematical formulation such as the traditional gradient-type of optimizing procedure. In fact, GA is much different in that context. It is merely a stochastic, discrete event and a nonlinear process. The obtained optima are an end product containing the best elements of previous generations where the attributes of a stronger individual tend to be carried forward into the following generation.

The basic principles of GA were first proposed by Holland [139]. Thereafter, a series of literature [44-47], [140-141] became available. The use of GA for pattern recognition has been widely studied. They can be generalized and grouped into two categories: feature extraction and classification [47]. Application of GA to various pattern recognition problems is described in [46]. One such application for designing a classifier is to exploit the searching capability of GA for placement of a number of lines for approximating the decision boundaries [142-143].

Combining GAs and NNs can be generally divided into two broad categories in supportive and collaborative integration [17], [47]. In supportive integration, GAs can assist NNs in

- transforming the feature space used by a neural net classifier
- selecting the learning rule or the parameters that control learning the NN

In collaborative integration, GA can be used to optimize NN on the weight parameters and/or topology [15], [41], [44], [46].

In this Ph.D. thesis we propose the use of a GA embedded into the feature extraction process [138]. So, we equip the HONN structure with a GA to force the resulting classes in the feature space to be as seperable as possible, thus providing, an integrated design solution.

A genetic algorithm (GA) can be defined as follows [144]:

Definition (Genetic Algorithm) A Genetic Algorithm $GA = \left(B_0, M, \Omega, \Gamma, \Phi, \Theta, t\right)$ is a 7-tuple with:

- $B_0 = \left\{A_1, ..., A_M\right\} \in P\left(B\right)$ a population from the set of all possible populations *P(B)*, A_i is of real or binary coding
- *M* the size of the *Population*
- $\Omega : P(B) \rightarrow \mathbb{R}^+$ the *Fitness fuction*
- $\Gamma : P(B) \rightarrow P(B)$ the *Crossover function*
- $\Phi : P(B) \rightarrow P(B)$ the *Mutation function*
- $\Theta : P(B) \rightarrow P(B)$ the *Selection Strategy*, and
- $t : \mathbb{R}^+ \rightarrow \left\{0,1\right\}$ a *Termination Function*

The main idea of evolutionary computing is that, having a population of good and bad solutions for a problem, try to produce a new population by mixing the properties of the solutions you have. Then, some new solutions could collect more desired properties. In that way better solutions have been achieved. By mimicking the principles of natural genetics, and using a number of solutions instead of a single one, GAs are able to search the total universe of discourse and find nearly optimal solutions (the possibility of finding does not depend on initial conditions). The structure of a genetic algorithm in its standard form is illustrated in Figure 4.7 [144-145].

```
Genetic Algorithm
 begin (1)
    t : = 1
    Initialize Population(t)
    Evaluate fitness Population(t)
    While (t < Generations) do
     begin (2)
        Apply Selection on Population(t) for the
            construction base(t) of  Population(t + 1)
        Crossover on construction base(t) to build Population(t + 1)
        Apply Mutation on Population(t + 1)
        Evaluate fitness of  Population(t + 1)
        t : = t + 1
     end (2)
 end (1)
```

Figure 4.7 The Structure of Standard Genetic Algorithm

The first step in a genetic algorithm application is to decide the way that solutions will be represented. In order to solve the problem:

$$\textit{Maximize} \quad f(\mathbf{x}) \text{ under } \mathbf{x}_{\min} \le \mathbf{x} \le \mathbf{x}_{\max} \tag{4.45}$$

x values are typically coded in a string structure. The strings can be in a binary or real form. In the first case binary coded values of predefined length are used to represent solutions. The length is determined according to the desired accuracy. In the second case, quantization is avoided and parameters can be estimated in any level of accuracy. These strings which represent encoded solutions of the problem are called *chromosomes*. Each solution consists of a number of parameter subsets whose value should be estimated during genetic process. Those parameter subsets are called ***genes***. Typically, a chromosome is constructed of a number of genes. The length of chromosomes can be constant or variable. A number of solutions (chromosomes) are used in each step to construct a population. Populations of different evolution steps are called ***generations***. A fitness value is assigned in each chromosome to say how good is that particular solution for the problem. Fitness value is usually computed as a function of the respective objectives and constraints. In the most usual case global objective (fitness) functions containing both objectives and constraints are constructed. To evolve solutions, genetic operators are applied on the current population to produce a new one. That process simulates genetic reproduction. The most known genetic operators are ***selection/reproduction***, ***mutation*** and ***crossover***.

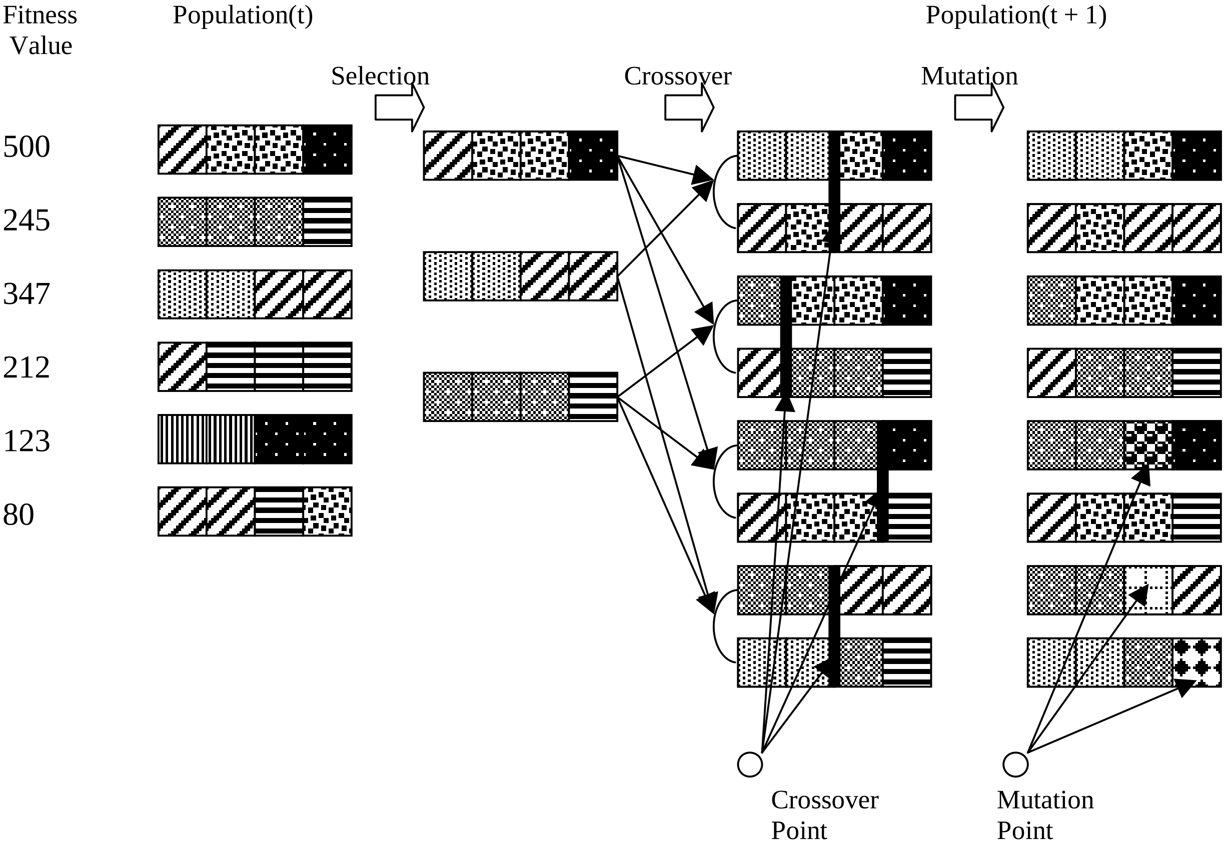


Figure 4.8 Sequential Application of Genetic Operators

In the ***selection*** process, a number of solutions within current population are selected to be the basis for the reproduction of the new population. That basis is often referred as a ***mating pool***. Selection strategy aims to solutions with highest fittest values, because they reasonably capture more desired properties. To construct a new population from that basis, the next two operators are used. *Roulette wheel parent selection* [141] and *linear selection* [146] are the most frequently used selection procedures.

Crossover is a way of creating new solutions, by randomly mixing properties between previous solutions. Different types of crossover operators have been proposed in the literature [44-47], [140-141]. In the standard form of single point crossover, two chromosomes are randomly selected from the mating pool (production basis), and some portion of the strings are exchanged between chromosomes. The operation is performed on two selected chromosomes and gives as output two new ones. The probability to perform crossover operation, i.e. ***crossover probability*** p_c [46-47], is chosen in a way so that recombination of potential strings (highly fitted chromosomes) increases without any disruption. Generaly, p_c lies in-between 0.6 to 1.0 [47].

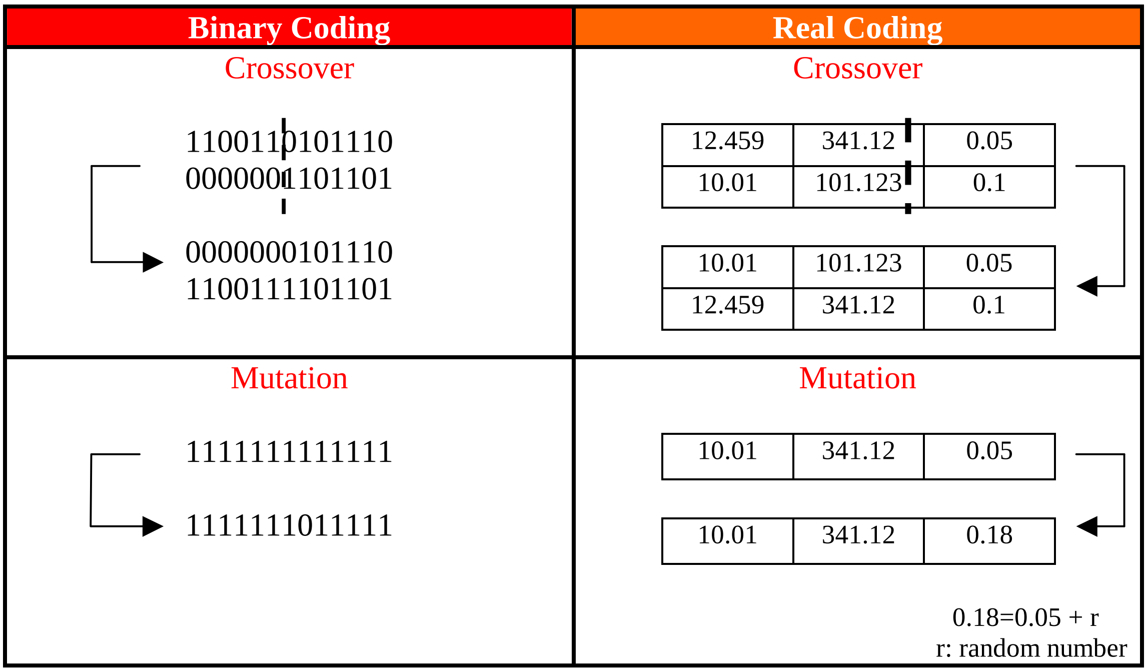


Figure 4.9 The difference between real and binary coding in the application of Crossover and Mutation operators

Mutation is performed upon a selected chromosome, by randomly changing the values of encoded parameters. In the case of binary coding, mutation changes a 1 to 0 and vice versa, depending on a small probability, while in the real coding case, random noise is added to encoded values. The need for mutation is to keep diversity in the population. Since mutation occurs occasionally, it is clear that the ***mutation probability*** p_m [46-47] will be very low. Typically, the value lies between 0.001 to 0.01 [46].

Even if different forms of selection, mutation and crossover have been proposed in the literature, their standard form is illustrated in Figure 4.8. The difference in the application of crossover and mutation operators in the case of binary and real encoding are presented in Figure 4.9.

4.1.5 The Feature Extraction and Feature Reduction Problem

So far we have been concerned with various techniques for pattern classification. Before a pattern recognizer can be properly designed, however, it is necessary to consider the ***feature extraction*** and ***feature reduction*** problems [7], [10], [11], [32], .

Any object or pattern which can be recognized and classified, possesses a number of discriminatory properties or features. The first step in any recognition process, performed either by a machine or by a human being, is to consider the problem of what discriminatory features to select and how to extract these features. It is evident that the number of features needed to successfully perform a given recognition task depends on the discriminatory qualities of the chosen features. However, the problem of feature selection is usually complicated by the fact that the most important features are not necessarily easily measurable, or, in many cases, their measurement is inhibited by economic considerations.

Usually the term ***feature extraction*** is used to describe the whole process of extracting suitable measurements from the 'raw' data. Feature extraction for classification is a process by which the original data is successively refined and reduced until the optimum number of features is selected for classification. The main scope of this process is to find salient features in the data, that is to find the best features or combinations of features to represent (and possibly explain) differences between different classes of data. For feature extraction, the original data, may be used or they may be processed or transformed in some way, for example to remove artifacts or noise or to make them more amenable to the extraction of relevant features.

Feature reduction is used to reduce the number of features, by combining the feature variables into a smaller set of new feature variables. While the number of initial features may be very large, the underlying dimensionality of the data, that is the intrinsic dimensionality may be quite small.

The main reasons for feature reduction are as follows:

- to facilitate visualisation of the data, and to allow the analyst to discern class structure and groupings within the data, and
- to reduce the number of variables for classification, either in order to reduce the ratio of variables to samples, or to reduce the computational complexity of estimating the density functions.

In this Ph.D. thesis we use a variety of techniques to extract relevant features from individual lead images for the particular problem of SMD PCB Post Placement Quality Inspection. Most spesifically, in Subsection 4.2.6.4 we present a feature extraction process based upon ***external image features*** (or ***boundary-based features***). Another feature extraction technique based on High Order Neural Networks receiving as input a normalized projection function of the tested individual lead image is introduced in Chapter 6. In addition to, the ***Principal Component Analysis*** (or ***Karhunen Loéve Tranformation*** as is also known in signal and image processing), [7], [11], [32] is then used for feature reduction and decorellation of the feature vectors.

4.1.5.1 The Karhunen Loéve Transformation for Feature Reduction

One of the simplest and commonly used statistical methods for reduction of dimensionality is principal component analysis (PCA) [9], [31], [32]. PCA operates by transforming the original variables into a new set of uncorellated variables called principal components (PC's). These new variables are linear combinations of the originals derived in decreasing order of importance. For example, the first PC accounts for as much as possible variation of the original data. If the original variables are highly correlated the first few PC's will account for most of the variation and the remaining PC's can be discarded with little loss of information. Ideally the first few components will be intuitively meaningful, will help us understand the data better, and will be useful in subsequent analyses where we can operate with a smaller number of variables.

The Karhunen Loéve Transformation (KLT) [7], [9], [11], [32] which relates to PCA, is a linear dimensionality reduction procedure. KLT is a useful method for reducing the dimensionality for both data display and classification, and forms a widely used method of dimensionality reduction.

Let **X** the matrix of the feature vectors of the training set (input matrix) and *N* the dimension of feature vectors (i.e., the dimension of feature space). In practice the algorithm of KLT proceeds as follows:

- **Step 1.** Compute the correlation matrix **R** of **X** (In this thesis we compute the ***correlation coefficient matrix*** , which gives us better results in classification).
- **Step 2.** Obtain the eigenvalues and corresponding eigenvectors of **R**. Normalize the eigenvectors.
- **Step 3.** Form the transformation matrix **Φ** from the *K* ($K \leq N$) eigenvectors corresponding to the largest eigenvalues of **R**.
- **Step 4.** Compute the final matrix **Y** by the equation $\mathbf{Y} = \mathbf{\Phi}^T \mathbf{X}$.

Thus the new feature space is a *K*-dimensional feature space.

In this PhD thesis, the KLT is used to de-correlate and reduce the dimensionality of feature vectors, disjoint class spaces in the new (reduced) feature space and aid the classifiers in performing accurate discrimination. Two different forms of the KLT are studied in this thesis. In the first only one KL transformation matrix (1 KLT) is created for the entire data set, whereas in the second one KLT matrix is created for each class (***multiple KLT*** approach) [148], [149].

4.1.5.2 The Proposed Multiple KLT Approach

We have proposed in [149] the application of ***multiple KLT*** approach as a general analytic tool. In multiple KLT approach, each individual class is represented by its most significant directions. For each vector in class i, only its projection on to the most significant directions of class i is preserved for classification. For each class, this approach preserves only the directions that best characterize the shape of its boundary and discards the rest. Thus, it encompasses class specific characteristics and uses them to better isolate and discriminate classes by avoiding class mixing in irrelevant directions. The fundamental goal of this approach is to create and localize many subspaces in the feature space, so that each class is best characterized by its own subspace reflecting the most characteristic vector directions and locations for this specific class. The directions and locations of feature vectors defining these reduced dimensionality subspaces are class-specific, even though the individual KLTs operate on the same initial feature space. Thus, a specific KLT projects feature vectors from its own class within its own subspace, whereas it forces vectors from “wrong” classes far away from this subspace.

During the review of our paper in Ref. [149], an independent work was published in Ref. [148] that studies the theoretical background of the multiple KLT approach. The experimental results presented in [149] and in Chapter 6 of this thesis fully support and verify the theoretical and experimental results of Ref. [148], which establish the multiple KLT approach as a general analysis methodology.

To emphasize the difference between the classical (simple) KLT (i.e., 1 KLT) approach and multiple KLT approach we visualize in Figures 4.10 and 4.11 the testing phase of classification process using a minimum-distance classifier along with 1 KLT matrix and 3 KLT matrices respectively for a three-class classification problem.

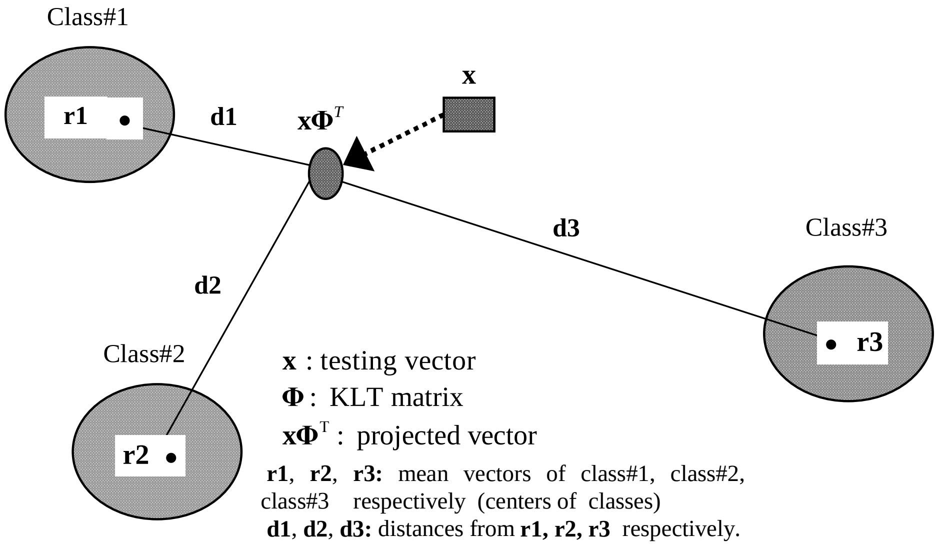
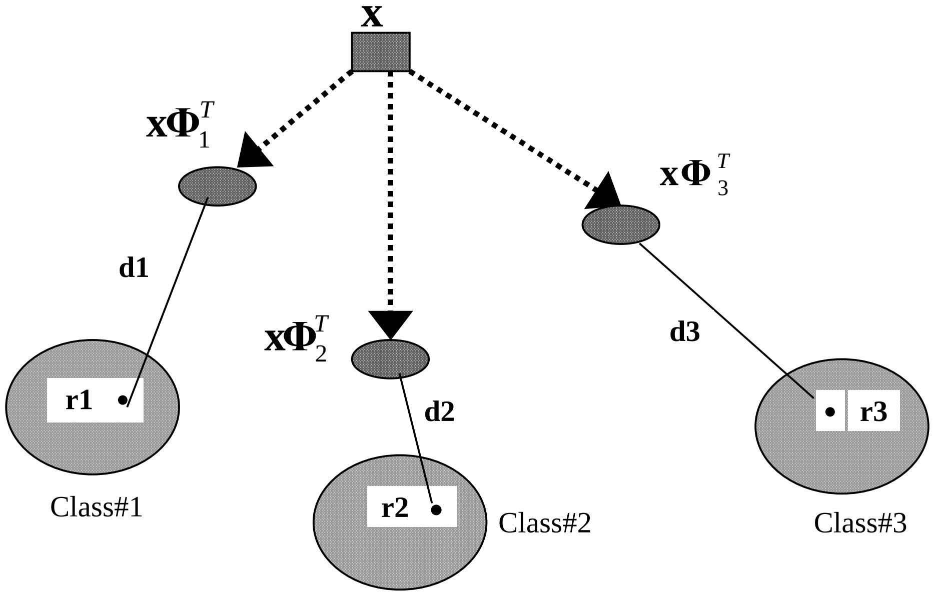

Class#1
r1
d1
x
xΦ^T
d3
Class#3
d2
r3
Class#2
r2
x : testing vector
Φ : KLT matrix
xΦ^T : projected vector
r1, r2, r3: mean vectors of class#1, class#2, class#3 respectively (centers of classes)
d1, d2, d3: distances from r1, r2, r3 respectively.

Figure 4.10 Testing phase of classification using a minimum-distance classifier and 1 KLT matrix

The classification approach for the minimum-distance classifier along with 1 KLT matrix presented graphically in Figure 4.10 can be described as follows:

(*M*-1) samples (class#1, class#2 and class#3) out of M, are used as feature vectors to create a $(M-1)\times K$ feature matrix. Feature matrix dimensionality is reduced from K to L to yield a $K \times L$ KLT matrix. The feature matrix is multiplied by the 1 KLT matrix resulting in a $(M-1)\times L$ feature matrix. This final matrix is used to train the above minimum-distance classifier. Notice that each projected feature vector is labeled, so that class statistics can be easily computed. The M^{th} sample (feature vector) is multiplied by 1 KLT matrix resulting in a $1\times L$ vector, which is then utilized as a ***testing vector*** (***jack-knifing process*** or ***leave one out technique***, [9], [17]) in the classifier. The above procedure is repeated M times until all feature vectors are utilized as testing vectors in the classifier. Each time the computation of the 1 KLT matrix and training is performed for the remaining (*M*-1) vectors.


x
$\mathbf{x}\mathbf{\Phi}_1^T$
$\mathbf{x}\mathbf{\Phi}_3^T$
$\mathbf{x}\mathbf{\Phi}_2^T$
d1
d2
d3
r1
r2
r3
Class#1
Class#2
Class#3

x: testing vector

$\mathbf{\Phi}_1, \mathbf{\Phi}_2, \mathbf{\Phi}_3$: KLT matrices for class#1, class#2, class#3 respectively

$\mathbf{x}\mathbf{\Phi}_1^{\mathrm{T}}, \mathbf{x}\mathbf{\Phi}_2^T, \mathbf{x}\mathbf{\Phi}_3^T$: class-projected testing vectors for class#1, class#2, class#3 respectively.

r1, **r2**, **r3 :** mean vectors of class#1, class#2, class#3 respectively (centers of classes)

d1,**d2**, **d3 :** distances from **r1, r2, r3** respectively.

Figure 4.11 Testing phase of classification using a minimum-distance classifier and 3 KLT matrices

The classification approach for the minimum-distance classifier along with 3 KLT matrices presented graphically in Figure 4.11 has as follows:

(M-1) samples (class#1, class#2 and class#3) out of M, where M is a multiple of 3, are used as feature vectors to create 3 different feature matrices (one feature matrix for each class). The dimension of the feature matrix for the class from which a feature vector has been drawn out is $\left\lfloor \frac{M-1}{3} \right\rfloor \times K$, whereas the dimension of the two others matrices is $\left\lfloor \frac{M}{3} \right\rfloor \times K$. The feature dimensionality is reduced from K to L to yield three $K \times L$ KLT class-matrices (one KLT matrix for each class). Each feature matrix is multiplied by corresponding KLT ***class-matrix*** resulting in the final ***class-feature matrix***. Thus, we take three final class-feature matrices. These matrices are used to train the above minimum-distance classifier. The M^{th} sample (feature vector) is multiplied by 3 KLT class-matrices resulting in three $1 \times L$ vectors, which are then utilized as ***class-projected testing vectors*** (***jack-knifing process*** or ***leave one out technique***, [9], [17]) in the classifier. The above procedure is repeated M times until all feature vectors are utilized as testing vectors in the classifier.

4.2 Image Analysis Techniques

Image analysis involves manipulating the image data to determine exactly the information necessary to help solve a machine vision problem. This analysis is typically part of a larger process, is iterative in nature, and allows us to answer application - specific questions such as: Do we need gray-scale or color information? Do we need to transform the image data into the frequency domain? Do we need to segment the image to find object information? What are the important features in the images?

Image analysis is primarily a data reduction process. Images contain enormous amounts of data, typically on the order of hundreds of kilobytes or even megabytes. Often much of this information is not necessary to solve a specific machine vision problem, so a primary part of the image analysis task is to determine exactly what information is necessary.

Image analysis is central to machine vision process and is often uniquely associated with machine vision. This high-level information may include shape parameters to control a robotic manipulator or color and texture features to help in the diagnosis of a tumor.

The image analysis process, illustrated in Figure 4.12, can be broken down into three primary stages: 1) Preprocessing, 2) Data Reduction and 3) Feature Analysis [58]. Preprocessing is used to remove noise (i.e. unwanted information that can result from the image acquisition process) and eliminate irrelevant, visually unnecessary information. In the second stage, ***data reduction***, we can perform ***segmentation*** on the image in the spatial domain or convert it into the frequency (or spectral) domain via a mathematical transform. After either of these processes we may choose to filter the image. This filtering process further reduces the data and allows us to extract the features that we may require for analysis.

In the third stage, feature analysis, the features extracted by the data reduction process are examined and evaluated for their use in the application. One of the important aspects of feature analysis is to determine exactly which features are important. So after the analysis we have a feedback loop that provides for an application-specific review. This approach often leads to an iterative process that is not complete until satisfactory results are achieved.

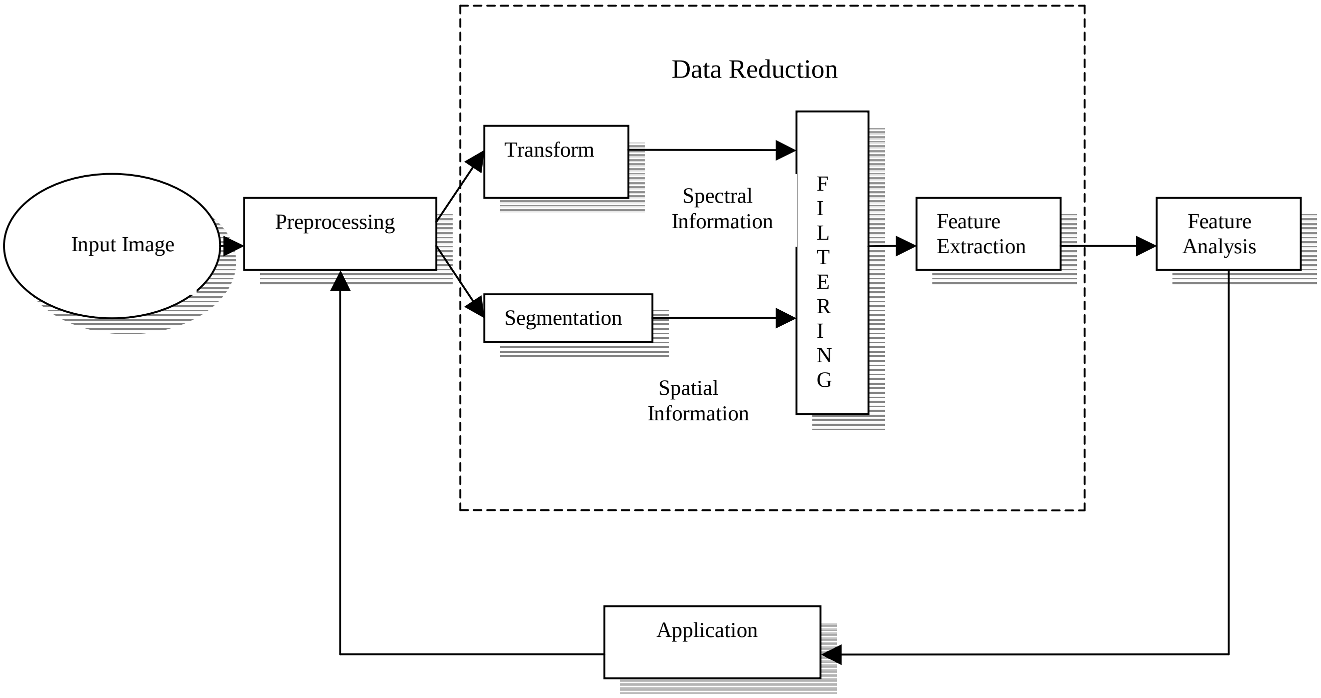


Figure 4.12 The stages of the image analysis process

4.2.1 Preprocessing Algorithms

The preprocessing algorithms, techniques, and operators are used to perform initial processing that makes the primary data reduction and analysis task easier. They include operations related to extracting ***regions of interest*** (ROIs), performing basic algebraic operations on images, gray-level or spatial quantization (reducing the number of bits per pixel), enhancing specific image features and reducing data in both resolution and brightness. Preprocessing is a stage where the requirements are typically obvious and simple, such as the removal of artifacts from images, or the elimination of image information that is not required for the application. For example, in one application we needed to eliminate borders from the images that had been digitized from film (the film frames); in another we had to mask out rulers that were present in skin tumor slides.

A central problem in machine vision and automatic visual inspection, in particular, is obtaining robust descriptions of the specific areas within the image, the so called ROIs. In complex pattern recognition applications such as SMD post placement inspection that we tackle in this dissertation, it is important to get an accurate and precise representation of the ROIs. To do this, we need initial operations that modify the spatial coordinates of the image, and these are categorized as image geometry operations. The image geometry operations include *crop*, *zoom*, *enlarge*, *shring*, *translate*, and *rotate* [58]. The ***image crop*** process is the process of selecting a small portion of the image, a subimage, and ***cutting*** it away from the rest of the image. After we have cropped a subimage from the original image, we can zoom in on it by enlarging it. This zoom process can be done in numerous ways, but typically a zero- or first-order hold is used. A zero-order hold is performed by repeating previous pixel values, thus creating a blocky effect. To extend the image size with a first-order hold, we do linear interpolation between adjacent pixels.

4.2.2 Image Segmentation

Image segmentation is important in many machine vision applications. The goal of ***image segmentation*** is to find regions that represent objects or meanigful parts of objects. Division of the image into regions corresponding to objects of interest is necessary before any processing can be done at a level higher than that of the pixel.

Image segmentation methods look for objects that either have some measure of homogeneity within themselves or have some measure of contrast with the objects on their border. Most image segmentation algorithms are modifications, extensions, or combinations of these two basic concepts. The homogeneity and contrast measures can include features such as gray-level, color, and texture. After we have performed some preliminary segmentation, we may incorporate higher-level object properties, such as perimeter and shape, into the segmentation process.

Different problems have been associated with image segmentation The major problems are a result of noise in the image and digitization of a continuous image. Noise is typically caused by the camera, the lenses, the lighting, or the signal path and can be reduced by the use of the preprocessing methods previously discussed. Spatial digitization can cause problems regarding connectivity of objects. These problems can be resolved with careful connectivity definitions and heuristics applicable to the specific domain.

Many techniques have been proposed in the literature for segmenting images. Some of them include histogram partitioning or ***thresholding***, region growing, border following, edge detection, spatial and measurement space clustering methods [22 – 27]. Selection of a segmentation technique depends highly on the nature of the image under consideration, i.e., whether the image is noisy, gray-level or is a color image.

4.2.3 Thresholding algorithms

The most widely used method for segmentation is the thresholding technique [150], [151]. Thresholding is performed using the histogram, $h(z)$, of the image, where z represents the gray-level, i.e., a plot of number of pixels versus number of gray-levels. In the thresholding method, the histogram of the given image is partitioned into a specified number of non-overlapping clusters. Each cluster specifies a range of gray-levels falling in a segment and all pixels whose gray-levels fall in that cluster are grouped together to form a segment. In the case of ***two-level thresholding*** [150], [151] only one threshold has to be decided, such that the objects are separated from the background. But, multiple objects (or ROIs) having different gray-level intensities necessitate finding multiple thresholds (***multi-level thresholding*** or ***multithresholding*** technique [150], [151]). Now the problem reduces to identiying the thresholds that in turn segment the image into multiple segments.

Segmentation by multithresholding started many years ago from simple beginnings, and in recent years has been refined into a set of mature procedures based on statistical decision theory and soft computing algorithms [152–163] . In-depth survey and evaluation of various thresholding methods are referenced in [164–167]. The outstanding problem is how to devise an automatic procedure for determining the optimal thresholds (i.e., ***automatic thresholding***). Automatic thresholding is an important technique in image segmentation and machine vision applications. The basic idea of automatic thresholding is to automatically select optimal gray-level threshold values for separating objects of interest (or ROIs) in an image from the background based on their gray-level distribution. This thresholding technique has been widely used in the industrial machine vision inspection systems [55].

Automatic thresholding techniques can be roughly categorized into global thresholding and local thresholding. Global thresholding selects a threshold value from the histogram of the entire image. Local thresholding uses localized gray-level information to choose multiple threshold values; each is optimized for a small region in the image. Global thresholding is simpler and easier to implement but its result relies on good (uniform) illumination. Local thresholding methods can deal with non-uniform illumination but they are complicated and slow. For industrial visual inspection applications, where non uniform illumination is usually not an issue due to controlled lighting conditions, global thresholding is commonly used for its simplicity and speed [157].

Among the global thresholding techniques, the Sahoo et al. [164] study concluded that the ***Otsu method*** [152] was one of the better threshold selection methods for general real world images with respect to uniformity and shape measures. This method selects threshold values that maximize the between-class variances of the histogram.

4.2.4 Region Identification

Image segmentation produces either a binary or a multilevel image output. The step following after segmentation is the identification of the image regions [22], [151]. Often, a region consists of a number of connected components. In order to extract features from the individual components it is necessary to identify and label the various connected components of each region. A ***connected component labeling*** algorithm assigns a unique integer number to each connected component and the largest integer label usually gives the number of regions in the image [22].

Labeling algorithms can be divided into two large classes: (a) local neighborhood algorithms and (b) divide-and-conquer algorithms [151]. The algorithms belonging to the first class perform iterative local operations. A simple divide-and-conquer algorithm can work along similar to the split and merge algorithm. Input to a labeling algorithm is usually either a binary or a multilevel image, where background may be represented by zero values, and objects (or ROIs) by non-zero values. A multilevel image is often used to represent the labeling result, background being represented by zero values, and regions represented by their non-zero labels.

4.2.5 Image Feature Extraction

The ***feature extraction*** aspect of image analysis seeks to identify inherent characteristics, or features of objects found within an image [22-27], [58], . These characteristics are used to describe the object, or attributes of the object, prior to the subsequent task of classification. Feature extraction operates on two-dimensional image arrays but produces a list of descriptions, or a *feature vector*. Features are used as inputs to the algorithms for classifying the objects into different classes. Object recognition can be done by analysing the morphology (shape and size), colour, texture (spatial distribution of colour), or a combination of these features of the images.

Two main categories of image features, namely, external and internal features have been distinguished in [168]. ***External image features*** (or ***boundary-based features***) describe the boundary information. Once the objects are separated from the background by segmentation, their boundary coordinates can be used to extract external features, such as perimeter, curvature, signature, bending energy, Fourier descriptors, 2-D transform coefficient features (e.g. Fourier, Haar, Hadamard, or Wavelet transform), image codes (e.g. run code, chain codes), etc [22 – 27]. External image features enjoy a certain popularity because they produce compact shape representations.

The features extracted from the properties of pixels inside the object boundary are called ***internal image features*** (or ***region-based features***). Spatial moments (e.g. ***center of gravity***, eccentricity, etc), ***area***, compactness, aspect ratio, major and minor axes lengths, ***elongation*** (or ***bounding box area***), ***projections***, ***skeletons***, colour and textural features are some of the most important intrenal image features [75], [151]. Internal image features have been used extensively in industrial machine vision applications [75], [151]. A detailed overview of image feature extraction algorithms is given in Appendix B.

4.2.6 Image Processing and Analysis Techniques for Lead Regions of the Components

From the aforementioned analysis in Subsections 3.4.1 and 3.4.2 of Chapter 3 becomes clear that the problems associated with the SMD inspection are complicated and become difficult due to the poor quality of the images obtained. The dynamic range of conventional CCD cameras is limited, so that the images are usually overexposed. The reflection on shiny parts (lead and solder paste) is heavy, further worsening the quality of the images. In this research, the high dynamic range achieved by the CMOS camera developed can result in high-quality images. Thus, the camera itself alleviates the quality problem, allowing the image analysis task to focus on the efficient extraction of relevant features for automated characterisation of the placement quality. Such image analysis approaches have been tested in the simulation platform and are presented in the following. The simulation platform has been based on images provided by industrial manufacturer. Throughout this consideration, a fundamental assumption is made regarding the high quality of the images provided by industrial manufacturer, that the different regions on the board after component placement can be discriminated on the images.

Each image covers one component in its entire extent. Nevertheless, there is no need to process the entire image to obtain the measurements required. The region of interest is restricted to rectangular windows that cover the lead regions of the component (*global ROIs*). Initially, one window for each side is required (two windows for the SO component and four windows for the QFP component, one covering each side of the component). Then, there is need for appropriate definition of sub-windows (*local ROIs*) for each lead region of the component (28 sub-windows for the SO component and 120 ones for the QFP component). After the sub-windows definition, an algorithm is required for segmentation of each lead region into four well isolated intensity segments: lead, pad, solder paste and dark background. Following the segmentation of lead regions, the next step is to derive objects related to lead, pad and solder paste. A connected components labeling algorithm for identification of closed regions is needed. Then feature extraction techniques must be developed to encode the characteristics of these regions. The abovementioned image processing and analysis techniques along with the classifier for pattern classification of each lead consitute the modules of a machine vision system for post placement quality inspection. Such a system is depicted in Figure 4.13.

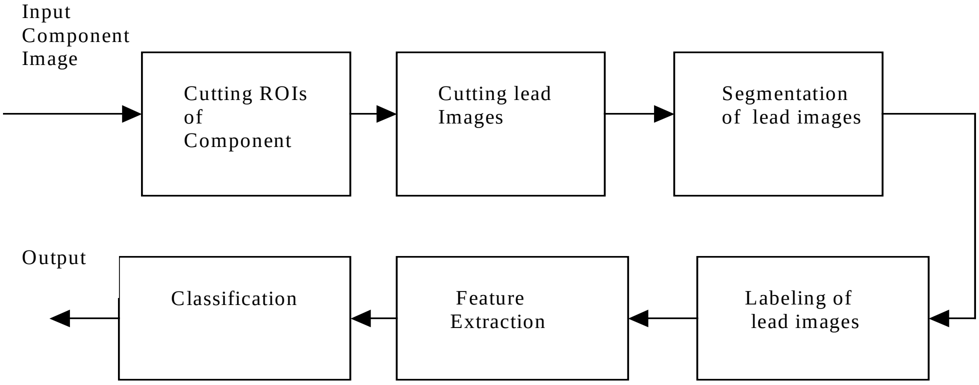


Figure 4.13 Machine vision system for post placement quality inspection

4.2.6.1 Isolation of Individual Lead Images

We describe below a simple procedure for isolation (or cutting process) of individual lead images from the QFP component image. The cutting process for the SO component becomes in a similar manner.

Taking into account the dimensions and the center (known coordinates from the placement machine) of the component, we capture the location of four global rectangular windows-images (global ROIs) to be acquired by the camera. Then in each *global ROI* we define a local window (*local ROI*) for each lead location to extract the individual lead-images. The whole process is illustrated in Figures 4.14 and 4.15.

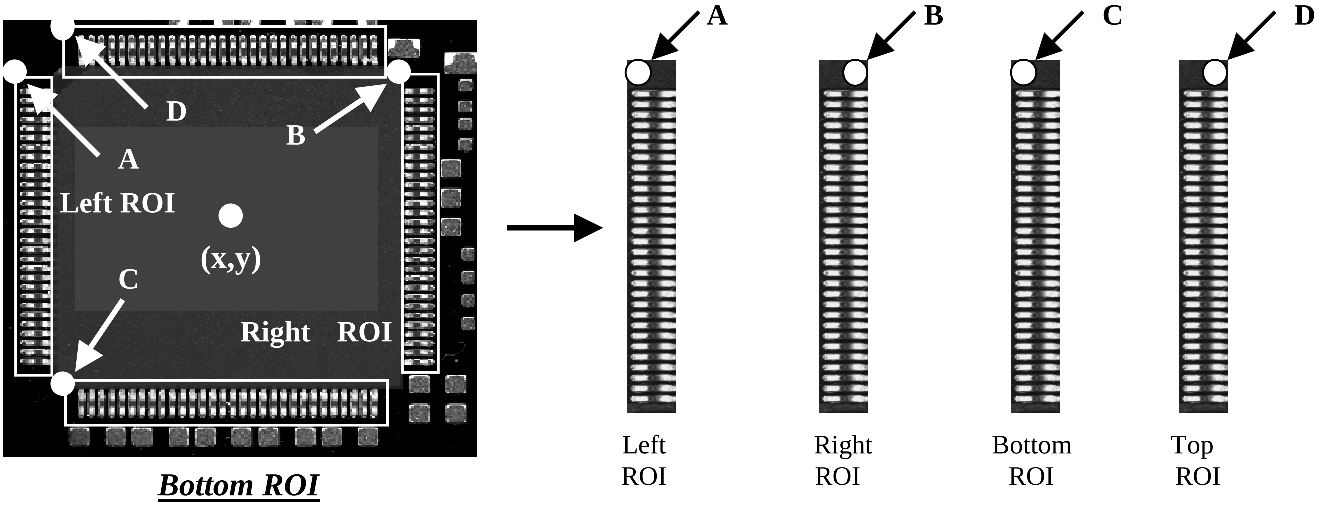


Figure 4.14 Definition of global ROIs, one for each side of the component

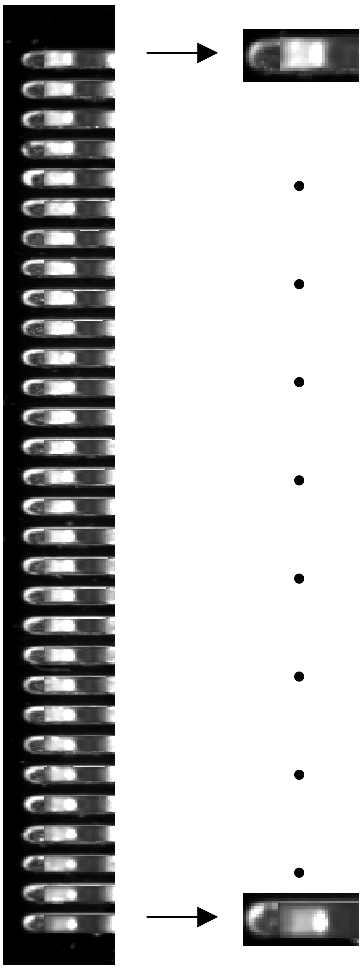


Figure 4.15 Isolation of individual lead images

Acquiring the global ROIs

Before we extract individual lead images, we must first acquire the global ROIs from each side of the initial image of the component (see Figure 4.14). Acquiring the global ROIs is simple task because we are based on the artwork of each image therefore we need two things:

1. The center of the component (see Figure 4.14). The center can easily be found setting horizontal and vertical rulers (we assume that it is given from the placement machine).
2. The dimensions of the component according Section 3.3 (see Figure 3.5).

Knowing these things we are able to calculate the necessary starting points **A**, **B**, **C**, **D** which are shown in Figure 4.14. These points are the starting points for raster global ROI acquisition. Therefore, the calculation of global ROI starting points for the component has as follows:

A:
$$\begin{aligned} x_A &= x - (body\_size/2) - length\_of\_lead - x\_tolerance \\ y_A &= y - (body\_size/2) - y\_tolerance \end{aligned} \qquad (4.46)$$

B:
$$\begin{aligned} x_B &= x + (body\_size/2) + length\_of\_lead + x\_tolerance - ROI\_columns \\ y_B &= y - (body\_size/2) - y\_tolerance \end{aligned} \qquad (4.47)$$

C:
$$\begin{aligned} x_C &= x - (body\_size/2) - x\_tolerance \\ y_C &= y + (body\_size/2) + length\_of\_lead + y\_tolerance - ROI\_columns \end{aligned} \qquad (4.48)$$

D:
$$\begin{aligned} x_D &= x - (body\_size/2) - x\_tolerance \\ y_D &= y - (body\_size/2) - length\_of\_lead - y\_tolerance \end{aligned} \qquad (4.49)$$

Since there is a likelihood the component to be misplaced and comes out of the pad the program applies a tolerance in the location of the starting points using two parameters which are defined as *x_tolerance*, *y_tolerance* for x and y axis respectively. At points B and C the program subtracts the quantity *ROI_columns* because the scanning of the image is from left to right and top down. Traversing the array that contains the pixels of the initial image and on the basis of these points we can return an array of the global ROI's pixels. The window's dimensions are *ROI_columns* and *ROI_rows*.

All global ROIs have the orientation of left ROI in order for processing them in the same way. To appear in this form right ROI undergoes a reflection, bottom ROI undergoes a 90° clockwise rotation and top ROI undergoes a 90° clockwise rotation followed by a reflection. The acquired global ROIs from the component are showed in Figure 4.14.

Cutting global ROI into individual lead images (local ROIs)

The isolation procedure proceeds to the extraction of smaller images (local ROIs) from the acquired global ROI having only one lead region at a time. Since pitch varies it is not effective to find the first lead region and then to cut the others in constant pace. So we developed an adaptive algorithm for locating the lead region. This is accomplished by doing a projection at y-axis (***horizontal projection***) in the array that contains the global ROI based upon the idea that lead region can be discriminated from the dark background because of their intensity level.

The projection of an image onto a line can be obtained by partitioning the line into bins and finding the number of 1 pixels that are on lines perpendicular to each bin and then dividing by the number of the bins [22 – 27]. Projection is a useful technique that manages to retain much information about the image. Horizontal and vertical projections can be easily obtained by finding the number of 1 pixels for each bin in the vertical and horizontal directions, respectively. The ***horizontal projection*** $H[i]$ along the rows and the ***vertical projection*** $V[j]$ along the columns of an image $I[i,j],\ i=1,...,n,\ j=1,...,m$ are given by:

$$H[i]=\sum_{j=1}^{m} I[i,j] \tag{4.50}$$

$$V[j]=\sum_{i=1}^{n} I[i,j] \tag{4.51}$$

The result of the horizontal projection of the global ROI is shown in the Figure 4.16.

The underlining assumption is that an area of lead corresponds with high values in the projection and an area of the PCB (dark background) corresponds with low values in the projection. So between two local minimums in the projection there is the area that corresponds to an individual lead image (local ROI). We applied the following algorithm:

1. Perform a horizontal projection in a constant number of rows. This constant number is quite enough to reach the first lead region. We expect a graph like in Figure 4.16 below.
2. Get the first point where the plot begins to rise. From the graph below we realize that the first rising of the plot is the beginning of first lead.
3. Extract an individual lead image from that point with height equal to pitch and length enough to encompass the whole lead region. Also perform projection from that point up to the point, which encompass the next lead region.
4. According to the graph in Figure 4.16 we are expected to meet two local maximums, one is the lead we have already found and the other is the next lead that we are going to locate.
5. Between the two successively maximums perform a sorting in order to get the local minimum. Go to step 3.
6. Repeat the steps from 3 to 5 until you acquire all leads (totally 30)

After the execution of aforementioned algorithm we acquire the individual lead images as have been illustrated in Figure 4.15.

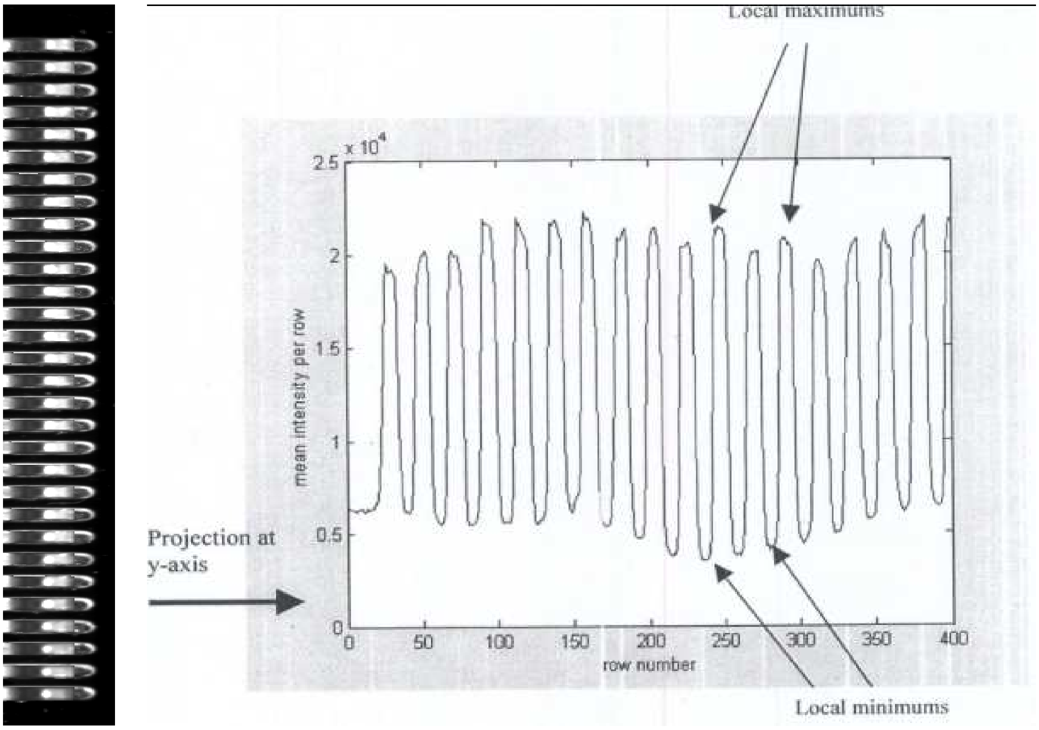


Figure 4.16 Horizontal projection of the global ROI

4.2.6.2 Threshold Selection Algorithms for Lead Images

Based on the assumption that lead regions can be discriminated by their intensity level, the histogram analysis (in each ROI) can be used for initial region segmentation. Four regions are of interest in the histogram, characterizing in decreasing intensity levels a) the pad and reflection regions, b) the lead regions, c) the solder paste, and d) the dark background. In this Ph.D. thesis several methods have been tested for threshold selection on the histogram.

- In first method a three-level thresholding algorithm in two stages have been used for segmentation of ROIs. In this method we find 2 thresholds on histogram of each side of component (global thresholds) and then we refine these thresholds finding new thresholds on histogram of each lead region (local thresholds) searching into small spaces of global thresholds.
- Then a variety of three-level and four level thresholding algorithms based upon Otsu's Thresholding method [152], Kittler and Illingworth's Minimum Error Thresholding Method [153] and Huang and Wang's Fuzzy Thresholding Algorithm [160] have been tested. These methods introduce problems in relation to quality of information into segmented regions
- Finally, the ***four-level Otsu's algorithm*** given the best segmentation results. So, this algorithm has been adopted for segmentation of lead regions.

We review the Otsu method for selecting optimal image threshold [152], [169] in Appendix C.

The Multilevel Thresholding Problem

An image can be represented by a 2D gray-level intensity function $I\left(x,y\right)$. The value z of $I\left(x,y\right)$is the gray-level (or the ***pixel intensity value***), ranging from 0 to $L-1$, where L is the number of distinct gray-levels.

The multilevel threshold selection can be considered as the problem of finding a set $T_k,\ k=1,2,...,K-1$ of threshold values, in order that the original gray-level image $I\left(x,y\right)$ would be transformed to a new one with only K levels (i.e. K classes) [159], [169]. More specifically, if $T_k,\ k=1,2,...,K-1$ are the threshold values with $T_1<T_2<...<T_{K-1}$, then the output image can be defined as

$$I_T(x,y)=\begin{cases} m_0, & \text{if} \quad I(x,y)\le T_1 \\ m_1, & \text{if} \quad T_1 < I(x,y) \le T_2 \\ \vdots & \\ m_{K-1}, & \text{if} \quad I(x,y) > T_{K-1} \end{cases} \tag{4.52}$$

where m_k represents the mean histogram values in the range $(T_k, T_{k+1}]$ with $T_0 = 0, \quad T_K = L-1$.

We assume that there are *K* classes, $C_1, C_2, ..., C_K$, in the image. We make use of the following parameters estimated from the histogram data to characterize the pixel value distribution in class *k*, T_k, $k = 1, 2, ..., K$.

Class a priori probabilities:

$$P_k(T_1,...,T_{K-1}) = \sum_{z=T_{k-1}}^{T_k-1} h(z) \tag{4.53}$$

where $h(z)$ is the normalized histogram function which represents the percentage of pixels having gray-level z over the total number of pixels of the image.

Note that there are *K*-1 independent class a priori probabilities since

$$\sum_{k=1}^{K} P_k(T_1,T_2,...,T_{K-1}) = 1 \tag{4.54}$$

Class means:

$$m_k(T_1,...,T_{K-1}) = \frac{1}{P_k(T_1,T_2,...,T_{K-1})} \sum_{z=T_{k-1}}^{T_k-1} zh(z) \tag{4.55}$$

Total mean:

$$m = \sum_{z=0}^{L-1} zh(z) = \sum_{k=1}^{K} P_k(T_1,...,T_{K-1}) m_k(T_1,...,T_{K-1}) \tag{4.56}$$

Total variance:

$$\sigma^2 = \sum_{z=0}^{L-1} (z-m)^2 h(z) \tag{4.57}$$

We can determine the optimal thresholds for ***Otsu's multilevel thresholding algorithm*** using the following criterion function:

$$J_{OT}(T_1,...,T_{K-1}) = \sum_{k=1}^{K} \sum_{z=T_{k-1}}^{T_k-1} h(z) c_{OT}^{(k)}(T_1,...,T_{K-1}) \tag{4.58}$$

where the cost function is

$$c_{OT}^{(k)}\left(z,T_1,\dots,T_{K-1}\right)=\frac{\left[z-m_k\left(T_1,\dots,T_{K-1}\right)\right]^2}{\sigma^2} \tag{4.59}$$

Segmentation results of individual lead images

Based on abovementioned Otsu's multilevel thresholding algorithm we implemented the four-level Otsu's algorithm, which given the best segmentation results for lead images. An example of input lead image and its segmented version is illustrated in Figure 4.17. The histogram along with thresholds of input lead image 4.17 (a) are presented in Figure 4.18. The four ROIs, viz: background (L1), solder paste (L2), pad (L3), and lead (L4), have been marked and illustrated in Figure 4.20 in Subsection 4.2.6.3 below.

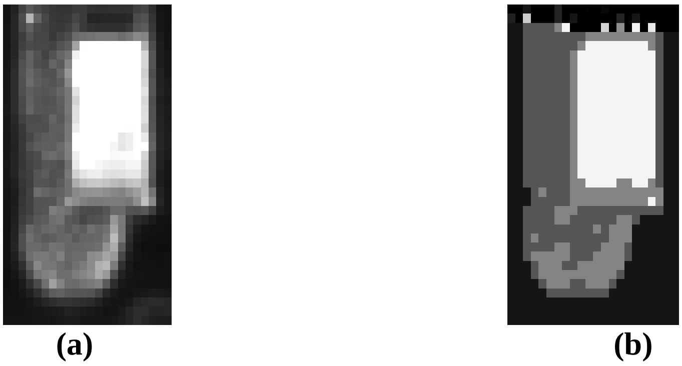


Figure 4.17 Segmentation of lead image based on the Otsu's four-level algorithm. **(a)** Original lead image **(b)** Segmented lead image

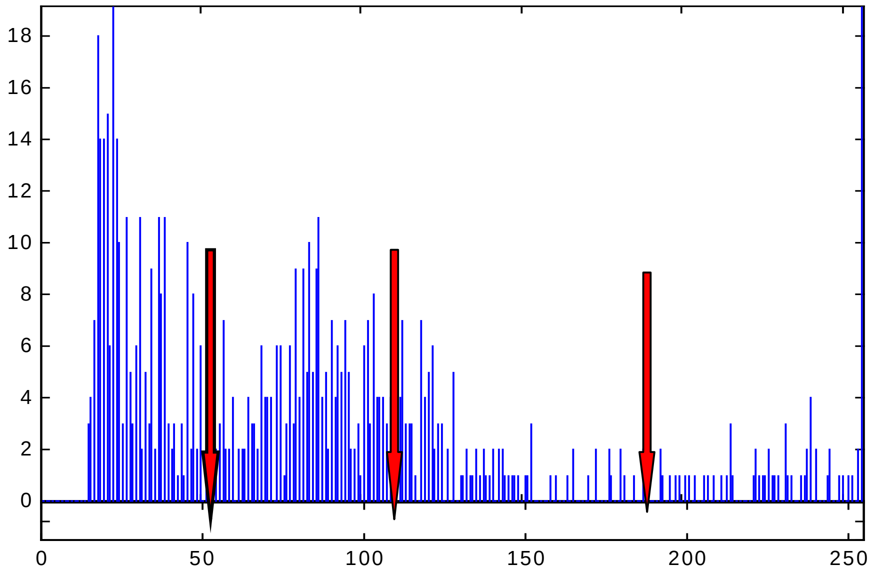


Figure 4.18 Histogram along with thresholds (red arrows) of original lead image.

Although a high dynamic-range camera has been used in this research, the illumination effects are obvious degrading the segmentation result, as can been regarded in Figure 4.19. The usual problems associated with illumination effects include the following: identification of multiple broken regions [broken pad in Fig. 4.19 (b), broken lead in Fig. 4.19 (d)]; lead and pad regions may appear with similar intensities [Fig. 4.19 (a)]; light diffusion effects at the borders of regions masking the solder paste [Figs 4.19 (a), (c)]; intense reflections on solder paste and pad regions [Figs 4.19 (a), (b),(c)] ; disappearing regions due to intensity similarity [Fig. 4.19 (e)]; union of regions [Fig. 4.19 (c)].

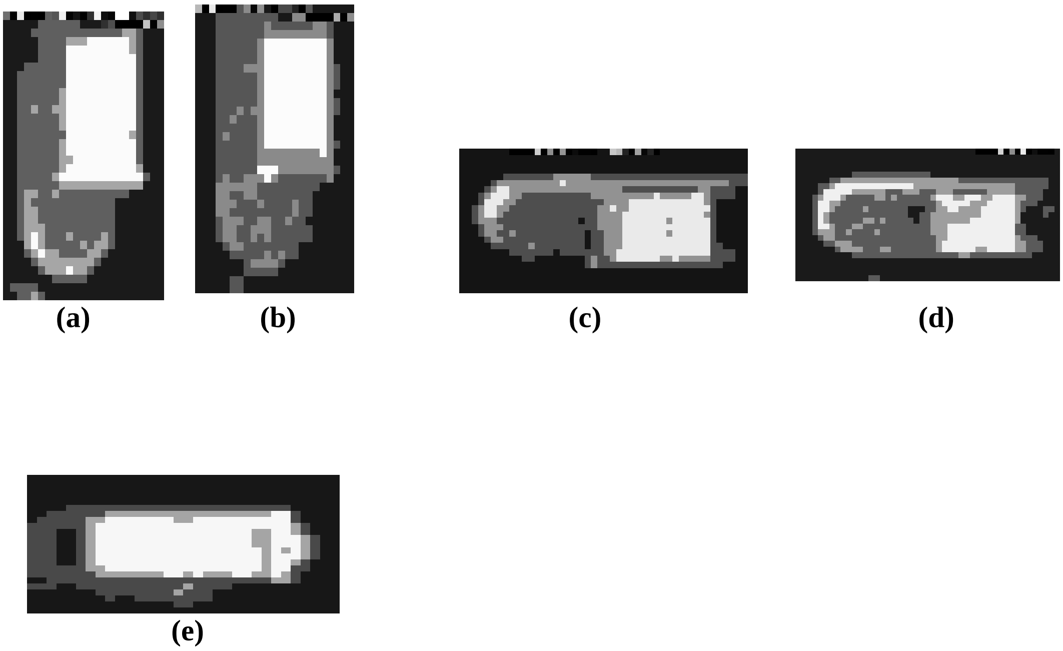


Figure 4.19 Segmentation problems in individual lead images after Otsu’s four-level thresholding algorithm

4.2.6.3 Component Labeling of Lead Images

The outcome of the Otsu’s four-level thresholding algorithm is a four level image that corresponds to the regions (ROIs) of background (L1), solder paste (L2), pad (L3), and lead (L4). These regions are illustrated in Figure 4.20.

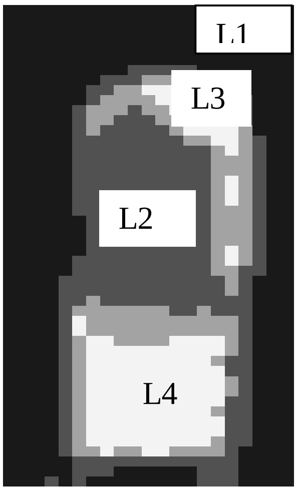


Figure 4.20 Regions (ROI's) after Otsu's four-level thresholding.

A component labeling algorithm that relies on certain criteria, such as intensity, shape, location, and size, is consequently applied in order to define (label) the four regions of interest (ROIs) and improve the segmentation results. A variety of image analysis techniques such as ***Aki and Toussaint's connecting component labeling algorithm*** [26], region growing and merging [23], [26], [151] line fitting [27], ***Graham's convex hull algorithm*** [170] along with heuristics methods have been used for the design and implementation of the aforementioned labeling algorithm. The main steps of this algorithm have as follows:

- Connected component labeling for complete segmentation and marking of included objects into initial segmented regions using Aki and Toussaint's algorithm.
- Removing all small regions from background region using heuristics methods.
- Correction of border using Graham's convex hull algorithm.
- Definition of lead region using criteria of intensity, position, area (bounding box area) of each lead label along with line fitting and region growing and merging algorithms to isolate the lead area.
- Definition of pad region using criteria of intensity, position and area (bounding box area) of each pad label.

The application of our component labeling algorithm results in a labeled image where the four ROIs are well-defined and segmentation problems have been corrected in a satisfactory grade. Such a labeled image is illustrated in Figure 4.21.

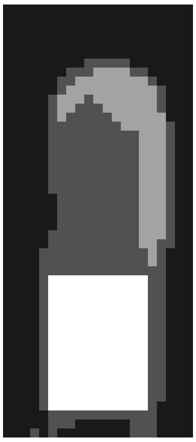


Figure 4.21 The lead image after component labeling.

4.2.6.4 Feature Extraction from Lead Images

After a component labeled image has been taken and the regions of interest have been well-defined, a feature extraction process, based on region-based features, is applied on labeled image to extract relevant features for classification. Our procedure proceeds as follows.

For each lead we define two lead sub-regions presented in Figure 4.22, based on the bounding rectangle. One region concerns the area where the lead is located and the other spans the area in front of the lead outwards of the component. The area from the lead to the backside of the component is disregarded, since it contains misleading (non-useful) information. The features of each sub-region are appropriately normalized to the length of the corresponding region, in order to make them independent of the axial (u-direction) shift of the lead within the area of its pad.

From **lead sub-region-1** we extract the following 7 features:

- Area of pad / L_1 (feature - 1)
- Area of solder paste / L_1 (feature - 2)
- Center-of-gravity distance on v axis between all non-background region and lead (feature - 3)
- Center-of-gravity distance on v axis between solder paste and lead (feature - 4)

- Center-of-gravity distance on v axis between solder paste and pad (feature - 5)
- Pad mean width on v axis (feature - 6)
- Pad total length on u axis / L_1 (feature - 7)

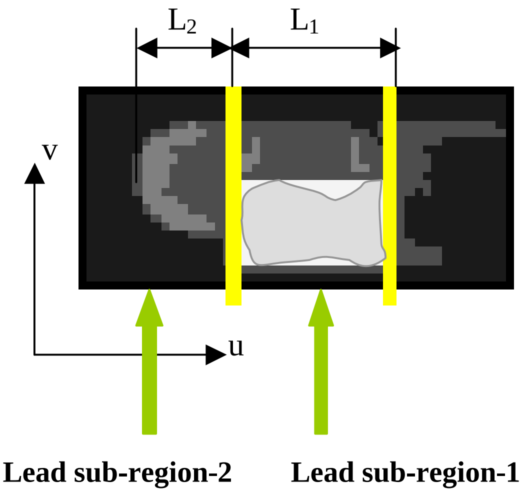


Figure 4.22 The lead sub-regions used in the feature extraction process.

From **lead sub-region-2** we extract the following 5 features:

- Area of pad / L_2 (feature - 8)
- Area of solder paste / L_2 (feature - 9)
- Center-of-gravity distance on v axis between all non-background region and lead (feature -10)
- Center-of-gravity distance on v axis between solder paste and pad (feature -11)
- Elongation of pad (feature -12)

The above 12 features constitute a feature vector for pattern classification of each individual lead. This set of features encodes optical characteristics (i.e. ***optical features***), by means of simple area measures that sustain the most desirable image attributes. For more detailed information regarding the calculation of the aforementioned features, the interested reader is referred to [23], [24], [26] and Appendix B.

Chapter 5

A Bayesian Framework for Multi-lead SMD Post-Placement Quality Inspection

5.1 Introduction

The SMD post placement quality inspection becomes essential as to comply with the zero defects policy of today's electronics manufacturing industry. Post placement inspection has the advantage that the inspection data are available immediately after placement, so no extra time and components are spend on an already faulty PCB. The later a defect is detected, the more expensive it is to repair. Thus, early detection is inherently cheaper. Moreover, correcting a defect after re-flow produces a more brittle joint and increases the risk of field failure.

In Subsection 3.5 we have presented the state of the art of SMD post placement quality inspection systems. In this chapter, we provide a novel framework to visually inspect the placement quality of SMDs immediately after they have been placed in wet solder paste on a PCB. **This work has been published in [19] and [30]**. Since we exploit only visual information, the parts inspected must be visible. Thus, our approach applies to "peripheral-type" SMD components with leads extending beyond the body of the component. Moreover, we only use simple light sources and fuse inaccurate information from many leads into a stochastic framework for accurate displacement estimation. In this respect, our approach is applicable to multi-lead components. Despite these limitations, there exist several types of SMDs that can be efficiently inspected by the proposed approach, including different versions of SOP (Small Outline Package), SOJ (Small Outline J-leaded Package), SOIC (Small Outline IC), and QFP (Quad Flat Package). In addition, our approach can be extended to different imaging technologies for inspecting other SMD packaging technologies, such as Chip-on-Film (COF) and Ball Grid Array (BGA) through X-ray and scanning acoustic microscopy, respectively [86-88].

In order to overcome the effects of poor quality, varying illumination images and/or inaccurate measurements, we introduce efficient modeling and stochastic estimation processes. Our approach exploits the fact that individual leads encode the same information regarding relative positioning of the rigid body of the component on the pad area. Positioning measurements on each lead can be viewed as individual (inaccurate) measurements of the same quantity regarding the component displacement and rotation. Instead of concentrating in one and every (poorly imaged) lead, we fuse complementary information from all leads into a Bayesian estimation framework.

The proposed analysis provides the necessary background for computing three measures of quality placement namely, overlap, insulation distance and slump gap, from individual lead images. These measures relate to distances of the regions of several types of material between the lead considered and its previous or subsequent lead regions. Using simple geometric relations, it can be shown that these measurements are only affected by the displacement (i.e., shift and rotation) of the component, relative to its pad region. It is worth mentioning that such a problem statement includes as a special case the visual solder joint inspection systems already reported [4], [98-100]. Thus, one major objective of this chapter is to derive an accurate estimation scheme for computing lead displacement based on observations. The proposed estimation approach operates in two levels. The first level considers a crude computation of quantized displacement of each lead. This is done through classification. The second level operates in a Bayesian framework and aims to accurately model the estimation of component displacement based on quantized lead displacements. A second objective is to provide a unified framework for the consideration of components and their different lead sides. This is achieved by establishing the necessary geometric relations that map all sides to a reference position and translate the effects of component displacement and rotation to that reference position.

5.1.1 Rationale of Proposed Approach

We can separate the consideration of important measures into two categories, namely microscopic and macroscopic. In the microscopic consideration we are interested in measures related to individual leads, whereas in the macroscopic consideration we study measures associated with the entire component as a rigid body (i.e., component displacement-shift and rotation). One may argue that lead shifts can be measured directly from the captured images, within isolated windows of each individual lead. Nevertheless, these measurements are bound to inaccuracies due to segmentation and labeling errors for the regions of pad, solder paste and lead. Alternatively, the lead shift can be inferred from the overall component displacement knowing the exact location of the lead on the component. In this approach, the inaccurate lead measurements can be considered as different measurements of the component displacement and can be used to estimate the total component displacement through a Bayesian estimation approach. Hence, there is a need to relate lead shifts to component displacement in a thorough geometric background, which we also provide. The displacement measurement of individual leads is modeled as a classification problem where the actual displacement is inferred from a feature vector computed on this particular lead. This classification process inherits several sources of inaccuracy, involving measurement errors on the lead image, non-separability of the feature space, classification and quantization errors. The Bayesian estimator developed attempts to diminish these sources of error. Subsequently, we may employ the geometric relations to infer a new refined measurement of the shift of each individual lead. Finally, we can estimate the lead quality measures through the relations that associate quality measures to lead shifts. The overall process is summarized in Figure 5.1.

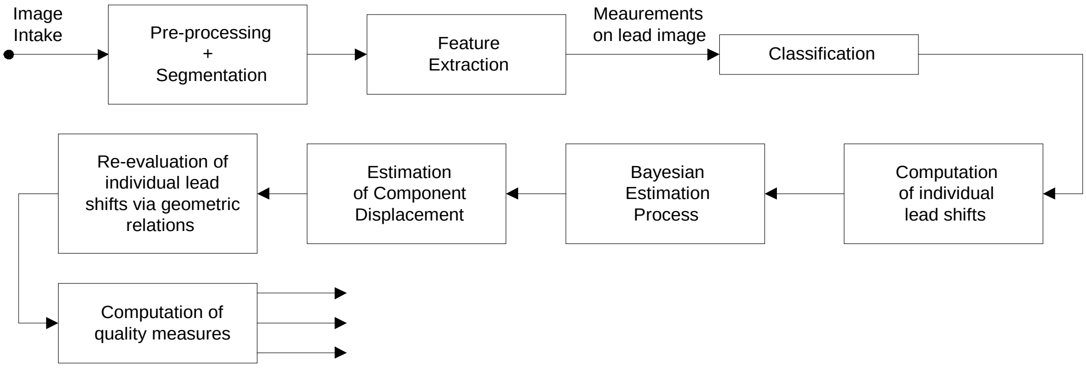


Figure 5.1 Post placement quality inspection algorithm in block diagram form.

5.1.2 Experimental Procedure

The inspection method proposed in this chapter is tested on an experimental set-up simulating the operation of placement machines. A motorized xyz-stage equipped with illumination and a camera with appropriate optics is used for collecting images from boards with peripheral type SMDs. The board is fixed to the base plate, which is localized at the appropriate position underneath the camera via the motorized control system. The actual location of the component is derived from the image itself during the processing stage. The illumination system is composed of three layers of ring-shaped LEDs with different illumination angles as to provide a wide angle of incidence (from 20 to 90 degrees) and simulate the “clouded-sky” type of illumination. The spectral distribution of the LEDs in the area of red and near infrared results in good contrast for the all cases tested (with green PCB). The 10-bit CMOS camera from Vector International is used for image capturing. The optics of the camera provide a telecentric view with a field of view 20x20mm for viewing the entire component on the PCB. The density of the CMOS sensor is 1024x1024 pixels, deriving an image resolution of 20x20μm per pixel. To capture the entire area of interest around each lead, the size of the lead images is set to 36x56 pixels. Notice that the extraction of lead images can be easily customized to any conventional SMD component.We successfully tested rectangular QFP components with 120 leads uniformly distributed along the four sides and SO components with 24 leads distributed along the two vertical sides. The results presented are for the more demanding case of the QFP120, which has side length in the range of [15.8-16.2] mm, according to its specifications. The pitch of the component is 0.4mm and the width of each lead is [0.13-0.23] mm. Thus, in the processed images the component pitch corresponds to 20 pixels whereas the lead-width may vary from 6 to 12 pixels.

In order to de-couple the training from testing of our classification-estimation scheme, we collected two sets of images. The training is based purely on ***Monte-Carlo simulated images*** generated from images of the PCB with solder paste but without the components placed on them and images of the component itself. Five sets of PCB and component images are acquired with different illumination levels and camera offset. Individual lead and pad segments are extracted from these images. All samples in each group of segments (pad and lead) are intermixed in a Monte-Carlo framework and used to provide samples of individual leads placed on pads at several levels of displacement.

The testing stage considers actual images of components already placed on their pad locations from PCBs obtained from an automatic placement machine. For the purpose of testing we used ten different boards, each containing two QFP120 components, with different settings from the placement machine. More specifically, the arrangements of components on the ten boards are as following. One board with perfect match; three boards with horizontal displacment of + 20, +60 and + 100 μm; three boards with both horizontal and vertical displacements at (+40, - 60) μm, (+ 20, -20) μm and (-120, +60) μm; three boards for testing the effects of rotation at multiples of 0.2 degrees.

5.2 Geometric Relations Between Lead & Component Displacement

In this section we establish how a rigid body component rotation or displacement relates to lead shifts, which is necessary for the calculation of the displacement quality criteria defined above. For the purpose of analysis, we use the model of a QFP (Quad Flat Pack) component, which is shown in Figure 5.2. Sample leads extracted from the sides of the component are presented in Figure 5.2(c).

5.2.1 Reorganization of Component Sides

The camera reads four ROIs (Regions Of Interest), one for each side of the component. Since we are not interested in the exact location of individual lead but rather on its displacement relative to its ideal position (i.e., pad), there is no need to define a coordinate system for each ROI. We can only define a single coordinate system (u,v) for all leads, independent of the lead’s location at the side of the component, and relate displacement on (u,v) to displacement on the (x,y) system. The (x,y) is a cartesian coordinate system established at the center of gravity of the component, as shown in Figure 5.2. From now on, we refer to this system as the component coordinate system. The (u,v) coordinate system is defined and oriented in the (x,y) space for all leads on each side of the component. Different sides impose different orientations on the lead coordinate system and they define different relations between the (x,y) and (u,v) systems.

To preserve consistency in the study of all sides of the component, we present each side on the same coordinates, placing the side of the component's body always at the same orientation. Thus, the component coordinates (x,y) may be different from the lead coordinates (u,v). As becomes apparent from Figures 5.2 and 5.3, we make the following transformation on the leads of each side of the component: the left ROI is kept unchanged; the right ROI sustains a mirror reflection of the horizontal axis (x-axis), over its vertical axis (i.e., the y-axis); the bottom ROI is rotated clockwise (CW) by 90°; finally, the top side ROI is first rotated CW by 90° succeeded by a mirror reflection of the x-axis over the y-axis. The component sides as transformed to lead coordinates are presented in Figure 5.2(b).

5.2.2 Effects of Component Displacements

Owing to the aforementioned transformation, a translation (δx, δy) of the component, results in a corresponding translation on the lead coordinates (u, v) that is summarized in the Table 5.1 and is graphically depicted in Figure 5.3.

Table 5.1 Translations measured on the lead coordinate system caused by a corresponding translation (δx, δy) of the component.

axis	Left ROI	Right ROI	Bottom ROI	Top ROI
δu	δx	-δx	δy	-δy
δv	δy	δy	-δx	-δx

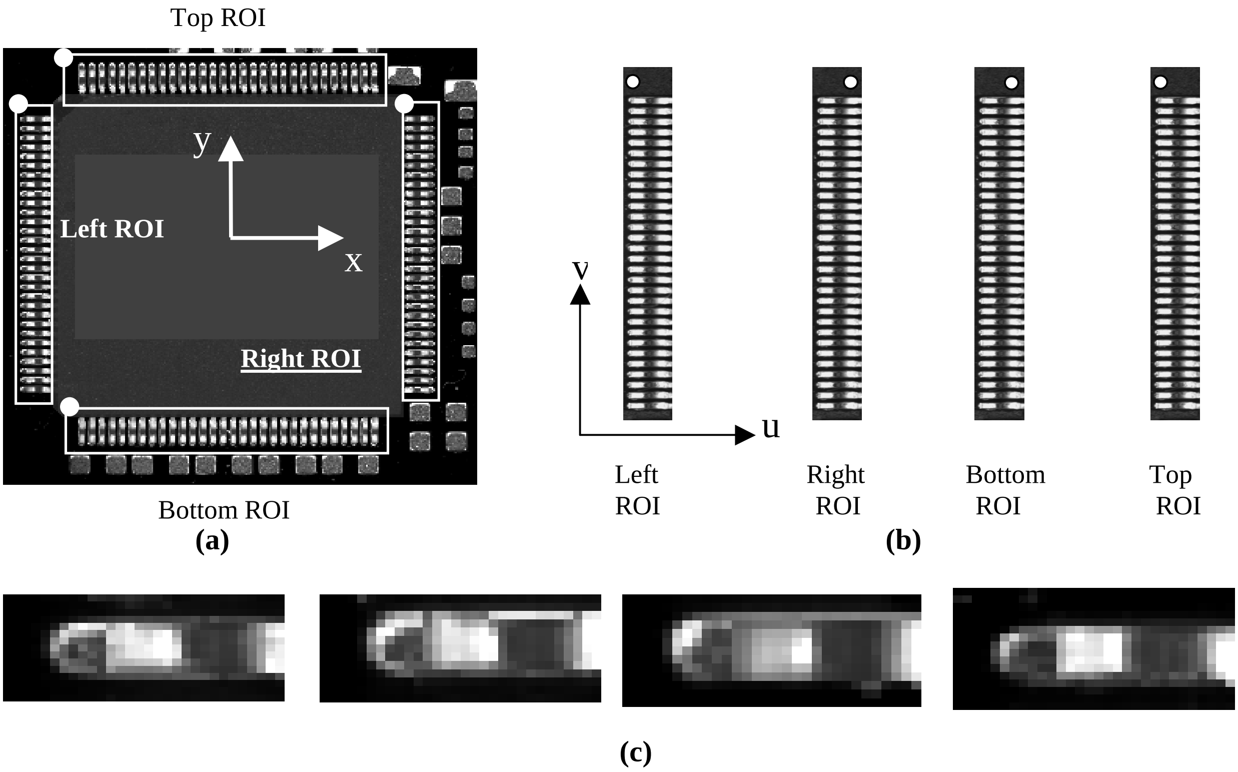


Figure 5.2 ROI selection and transformation; **(a)** location of ROIs at the left, right, bottom and top sides of the component, **(b)** spatial transformation/relocation of ROIs, **(c)** sample lead images from the different sides.

We now consider the effects of component rotation. Let a lead L_0 (see Figure 5.4) at the right side of the component, located on the x-axis of the component coordinate system, at distance R from its center. A clockwise (CW) component rotation by φ degrees, is equivalent to a Counter Clockwise (CCW) rotation by the same angle of its coordinate system. It is readily derived that such a rotation will impose a translation (*dx*,*dy*) of the lead, expressed in the (*x*,*y*) component coordinate system as:

$$\begin{aligned} dx &= R\cos\varphi - R \\ dy &= R\sin\varphi \end{aligned} \qquad (5.1)$$

Consider now a lead L_h (see Figure 5.4) at the right side of the component, displaced by a distance h from the x-axis of the component coordinate system. In order to express the effects of rotation on L_h similar to that of L_o we introduce a new coordinate system (x_h, y_h) which is formulated by vertically translating the (x,y) system such as the x_h-axis is passing by the center of L_h. In this way the coordinates of the center O_h of the (x_h, y_h) system with respect to the (x,y) system is (0, h). Introducing this new system, the original rotation around *O* can be decomposed in two parts; one that reflects the change of coordinate systems and the effects of rotation over *O* to the new center O_h and a second one that considers the effects of rotation over the new center, with respect to the new coordinate system itself. Thus, a CW component rotation by φ degrees causes a displacement of lead L_h that is decomposed into two parts; one related to displacement of the coordinate center O_h at a new position $O_h^{'}$ (due to its own rotation around the center *O* of the component coordinate system) and a second related to the rotation of the lead around $O_h^{'}$.

The second part of the transform causes a translation of L_h as in Eqn (5.1), whereas the former one causes a translation of L_h by:

$$\begin{aligned} dx_h &= -h\sin\varphi \\ dy_h &= h\cos\varphi - h \end{aligned} \qquad (5.2)$$

Overall, in the component coordinate system *(x,y)* the total displacement imposed on L_h (on the right side of the component) due to CW component rotation by φ degrees is given by:

$$\begin{aligned} dx(r) &= \left(R\cos\varphi - R\right) - h\sin\varphi \\ dy(r) &= R\sin\varphi + h\cos\varphi - h \end{aligned} \tag{5.3}$$

For $|\varphi| < 1^{\circ}$, as in the specifications of commercial placement machines, $\cos\varphi \cong 1$ and $\sin\varphi \cong \varphi$, yielding:

$$\begin{aligned} dx(r) &= -h\varphi \\ dy(r) &= R\varphi \end{aligned} \tag{5.4}$$

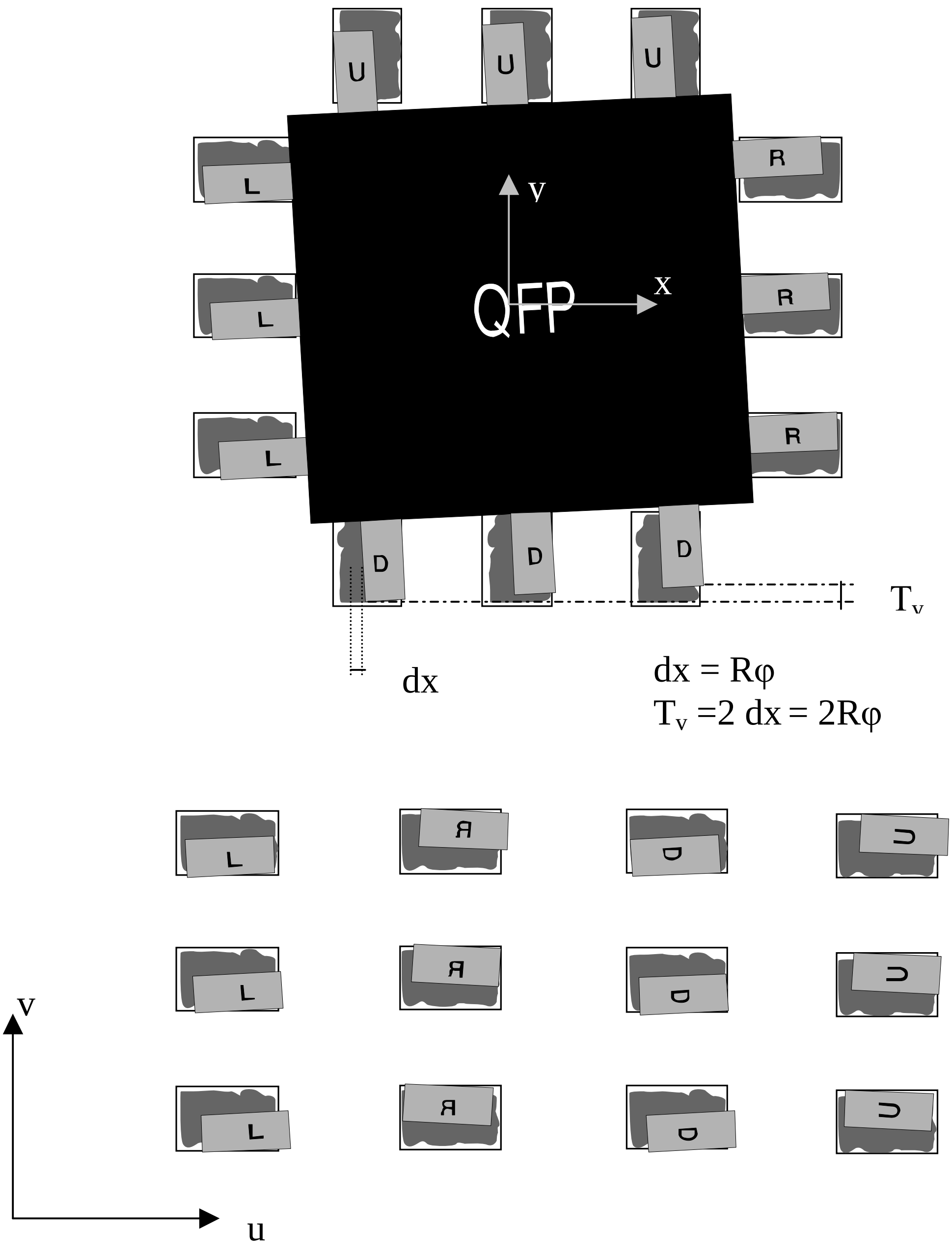


Figure 5.3 Component translation and rotation effects measured on the lead coordinate system.

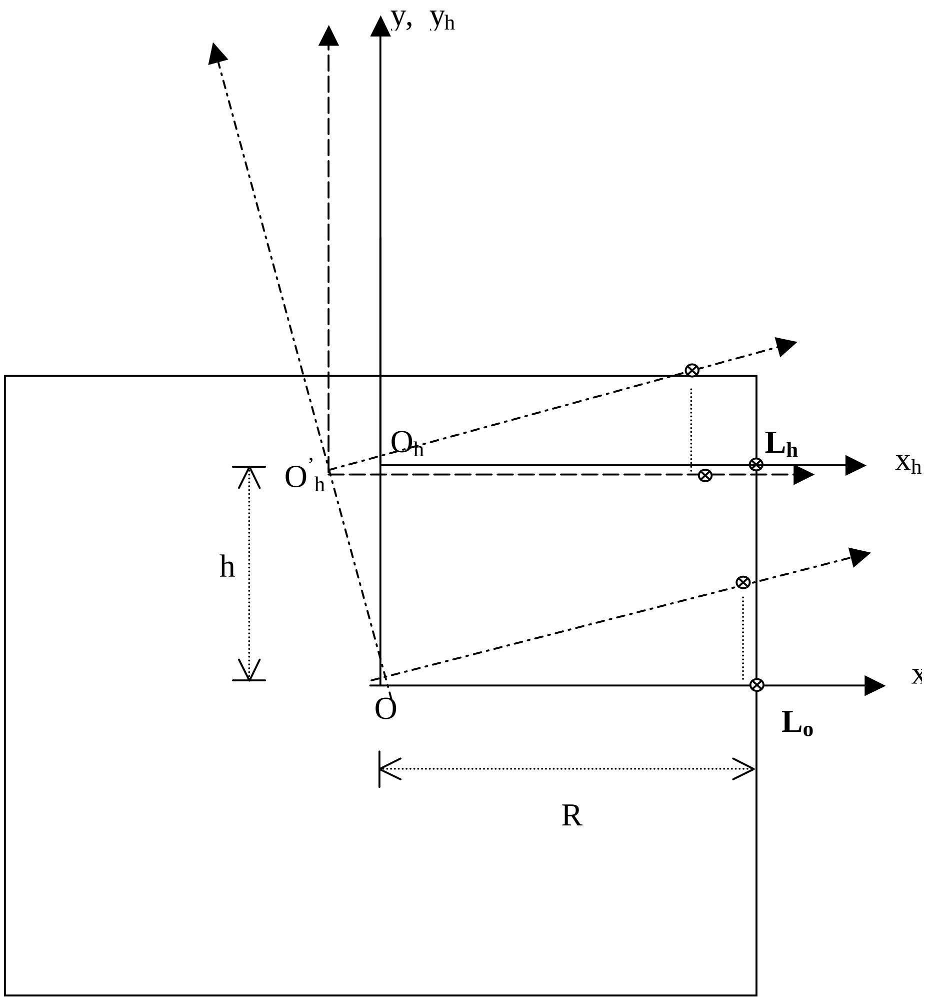


Figure 5.4 Analysis of component rotation on the shifts measured on the lead coordinate system.

Performing similar analysis for the left, up and down sides of the component, we obtain:

$$\begin{array}{l} dx(l) = -h\varphi \\ dy(l) = -R\varphi \end{array}, \quad \begin{array}{l} dx(u) = -R\varphi \\ dy(u) = w\varphi \end{array} \quad \text{and} \quad \begin{array}{l} dx(d) = R\varphi \\ dy(d) = w\varphi \end{array} \tag{5.5}$$

where w is the distance of the center of a lead which is placed on the up or down sides of the component from the y-axis of the component coordinate system. To arrive at the effects of rotation on the lead coordinate system (u,v), we further introduce the relations of Table 5.1, resulting in the relations of Table 5.2.

Table 5.2 Translations measured on the lead coordinate system caused by pure component rotation by φ degrees.

axis	Left side	Right side	Down side	Up side
δu	-hφ	hφ	wφ	-wφ
δv	-Rφ	Rφ	-Rφ	Rφ

Notice that h and w are signed lead distances from the origin of the (x,y) system, for the left/right and up/down sides respectively.

Overall, if a component translation by δx, δy is experienced in addition to a pure component rotation by φ degrees, the total lead displacements are summarized in Table 5.3

Table 5.3 Translations measured on the lead coordinate system when both component translation by δx, δy and rotation by φ degrees are present.

axis	Left side	Right side	Down side	Up side
δu	δx –hφ	-δx +hφ	δy +wφ	-δy -wφ
δv	δy -Rφ	δy +Rφ	- δx -Rφ	-δx +Rφ

5.2.3 Estimating Component Displacements from the Leads

The directional relations of Table 5.3 can be specialized to the k-th lead of each side, depending on the appropriate h or w location of the lead. For the k-th lead r_k of the right side, as example, we use the convention $\delta u(r_k) = -\delta x + h_k\varphi,\ \delta v(r_k) = \delta y + R\varphi$ to denote the corresponding directional shifts. It is obvious, therefore, that provided an accurate estimation of the component displacement and rotation, the effects can be accurately reflected to the individual leads. The reverse argument holds also true: provided the lead shifts (*δu, δv*), we can infer the component parameters (*δx, δy, φ*). Notice that the lead shift δv is independent of the exact location of the lead on its corresponding side. Moreover,

$$\begin{aligned}\delta u(l_k) + \delta u(r_k) &= 0\\ \delta v(l_k) + \delta v(r_k) &= 2\delta y\end{aligned} \tag{5.6}$$

Similarly,

$$\begin{aligned} \delta u(d_k) + \delta u(u_k) &= 0 \\ \delta v(d_k) + \delta v(u_k) &= -2\delta x \end{aligned} \tag{5.7}$$

The rotation angle φ can also be derived from similar combinations of lead displacements. These interrelations between lead and component displacements form the essence of our approach in accurately estimating the component parameters, i.e. we can focus on lead measurements and then reflect measurements to component displacement parameters. Nevertheless, since we accept that the measurements on each individual lead are erroneous, we combine measurements from many leads in a Bayesian framework, for inferring an accurate overall estimate of component parameters. As indicated by Eqns (5.6) and (5.7), we need only measure lead displacements δv on their cross-axial direction. Thus, we emphasize that our measurement scheme is only limited to lead displacement δv.

Let d_L, d_R, d_D, d_U denote the displacements on the cross-axial lead direction (v axis) of the left, right, down and up side, respectively. These measures may represent the displacement of an individual lead, or the combined displacement of the entire side (average of individual lead displacements along this side), since we consider the component as a rigid body. In fact, each of these displacements may represent an accurate estimate of the side's displacement based on inaccurate displacement measurements of several individual leads on the same side of the component, (as described in Section 5.3). From those estimates, we can arrive at the estimated horizontal and vertical component shifts δx, δy respectively, as well as the estimated rotation angle φ, exploiting of the relations (5.8):

$$\begin{aligned} \delta x &= -\frac{(d_D + d_U)}{2} \\ \delta y &= +\frac{(d_L + d_R)}{2} \\ \varphi &= -\frac{(d_L - \delta y)}{R}\frac{180^\circ}{\pi} = \frac{(d_R - \delta y)}{R}\frac{180^\circ}{\pi} = -\frac{(d_D - \delta x)}{R}\frac{180^\circ}{\pi} = \frac{(d_U - \delta x)}{R}\frac{180^\circ}{\pi} \\ &= \frac{d_R - d_L + d_U - d_D}{4R}\frac{180^0}{\pi} \end{aligned} \tag{5.8}$$

5.3 Lead Displacement Estimation

The computation of lead displacements on each side of the chip proceeds in three steps. The first step implements a feature extraction process based on a segmented lead image. In the second step, an appropriately designed classifier takes as input the extracted feature vector for each lead and produces an estimated quantized displacement y_k^r classified as l_r (i.e. with label l_r), where the index *k* refers to the k-th lead of a component's side. The quantization step can be fixed to any size, as long as the training of the classifiers is performed with this resolution. In our approach we consider pixel size and multiples of it. The third step operates on all quantized lead shifts computed from each side and implements a likelihood estimation process in order to estimate a total shift on that specific side. From this estimation process, we can easily proceed to the evaluation of quality measures and the computation of the component's displacement in (x,y) and rotation, using the relations in Eqn (5.8).

5.3.1 Feature Extraction and Classification

After a SMD component image has been acquired and the regions of interest have been defined, a four-level Otsu algorithm, which has been presented in Subesection 4.2.6.2 of Chapter 4, is applied on each ROI, to segment the lead images that are included in the examined ROI image. The outcome of the segmentation algorithm is a four level image that corresponds to the regions of lead, pad, solder paste and background. As indicated by Figure 5.2(c), the segmentation of lead and other regions might be quite difficult depending on the illumination incident to that lead area. In our approach the lead is only roughly segmented, as has been illustrated in Figure 4.22 of Chapter 4, and is simply enclosed by its bounding rectangle. The labeling algorithm that has been given in Subsection 4.2.6.3 is consequently applied in order to define (label) the four areas of interest. The introduced in Subsection 4.2.6.4 feature extraction process results in 12 relevant features (i.e. ***optical features***).

The abovementioned 12 optical features constitute a feature vector for pattern classification of each lead. Any classifier can be utilized to perform this task. In this work, we use a Bayes classifier and a LVQ neural network classifier, reinforced with a Karhunen-Loeve (K-L) transformation as to reduce the dimensionality of the input feature space.

5.3.2 Quantization and Classification Errors

Inevitably, a classification process that produces displacements quantized to the pixel size introduces two types of error, namely the quantization and the classification error. Since we measure lead displacements via a discrete classification process, the resulting measurements are quantized and admit only discrete values l_r. Nevertheless, the actual displacements we attempt to estimate are real-valued. Hence, an error occurs in the classification of the individual lead's displacement, which is referred to as the quantization error.

In addition, the classifier as a system yields a classification decision that is not 100% accurate. This error is called the classification error. Let the outcome of the classifier l_r be associated with the true class ω_i, i.e., the classifier yields a classification l_r whereas the true class of the lead is ω_i (must be correctly classified as l_i). A measurement vector (i.e., lead features) from true class ω_i may be assigned any allowable level l_r with the probability $P\left(l_r / \omega_i\right)$, which models the probability of classifying a true class ω_i at level l_r. These probabilities can be estimated through extensive testing of the classifier.

Both these errors associated with our classification process can be considered within a Bayesian estimation process, which aims to approximate the actual component displacement from the inaccurate classifications of several individual leads. In this framework, we view the classified (quantized) displacement of each lead as an individual measurement of the same process, i.e. the displacement of the component as a rigid body. Thus, from *K* individual quantized measurements y_k^r from one side of the component, with ***probability of misclassification*** $P\left(l_r / \omega_i\right)$, we proceed to the estimation of the real-valued displacements of the entire component side. Recall at this point that the lead displacement measured is along the cross-axial direction of the lead, i.e., axis *v* in Figure 5.3. Taking under consideration the relation of the lead and component coordinate systems, it becomes obvious that the above estimation process concerns only the δy displacement for the left and right sides and the δx displacement for the up and down sides of the component. This effect has been also embedded in the relations of Eqn (5.8).

5.4 Component Displacement Estimation

5.4.1 Definitions

In this section we use the symbols *x* and *y* to denote random variables for continuous and discrete (or quantized) displacement measurements, respectively. The essential displacements that we need to accurately estimate are the lead displacements on their cross-axial direction, i.e. d_L, d_R, d_D, and d_U from Subsection 5.2.3. Let s be the actual (true) value representing any one of these displacements, which can also be considered as an overall component displacement at this side due to either horizontal or vertical shift, or rotation. The component, as a rigid body, transfers this displacement to individual leads. The displacements we measure in Section 5.3 on individual leads can be considered as inaccurate measurements of s. The task considered in this section is to derive an estimation process for inferring component displacements from the individual measurements of lead displacements. Before proceeding, we make the following definitions:

- s : is the real-valued (continuous) component displacement in one direction.
- x denotes the continuous random variable for the displacement of leads on the appropriate sides.
- x_k is the continuous lead displacement of k-th lead of one side of the componenet (i.e., the k-th measurement of s).
- ω_i : is the *i*-th class (with label l_i) of discrete lead displacement.

Since we measure lead displacements via a discrete classification process, the resulting measurements are quantized and take only discrete values y_k^r.

- y is the discrete random variable of for the lead displacements.
- y_k^r is the discrete lead displacement of the *k*-th lead classified as l_r from classifier.

For a specific displacement s, the measurement x may assume a Gaussian distribution around s, with mean s and variance $\sigma_x^2 = \sigma^2$. This models the distribution $P(x/s)$. The distribution of the quantized measurements $P(y/s)$ is a discrete one that can be obtained from $P(x/s)$.

Let the quantization levels be defined as l_r, $r \in [-M, \ldots, 0, \ldots, M]$, ($r$ being an integer) and the limits of quantization for l_r be a_r and b_r. We consider uniform quantization with $b_r - a_r = T$ so that $a_r = l_r - T/2$ and $b_r = l_r + T/2$. Thus, the probability of quantized measurent y given s, $P(y/s)$ is determined by the probabilities $P(l_r / s)$ of the quantization levels l_r as:

$$P(l_r / s) = \int_{a_r}^{b_r} P(x/s)dx = \frac{1}{\sqrt{2\pi}\sigma} \int_{l_r - T/2}^{l_r + T/2} e^{-(x-s)^2/2\sigma^2} dx \qquad (5.9)$$

In case that the mean s coincides with the central level of quantization l_0, (i.e., $s = l_0$), the distribution $P(y/s)$ is a binomial one. In case that s falls within the quantization region of any level l_r, then the distribution $P(y/s)$ approximates a binomial one around l_r. Throughout our estimation process we do not consider such a binomial approximation for the entire variable y, but rather consider local approximations of $P(l_r / s)$ in (5.9) around l_r.

Since we do not measure directly the lead displacement $y(y_k)$, but we only obtain it indirectly through a classification process, we need to consider the effects of the classification error. Let the outcome of the classifier y_k at the k-th trial be associated with the class ω_r, $r \in [-M, \ldots, M]$ (level l_r) as organized by the specific classifier. A measurement Y, (its feature vector), may be classified to class ω_r (level l_r) and consequently be assigned the value l_r, with a certain probability $P(Y/\omega_r)$, which models the probability of classifying Y into any class, even though the actual shift belongs to class ω_i. This models the classification error, or the probability of classifying a measurement y_k to class ω_r instead of ω_i. We consider this error independent of the actual displacement s, depending only on the classifier itself. Thus, we estimate these probabilities through extensive testing of the classifier.

One last variable that we need to model is the actual displacement s of the component, which relates to the placement error of the PCB machine. Let $P(s)$ be its distribution. From nominal measurements s, can be modeled as a Gaussian with mean s_0 and variance σ_0^2.

5.4.2 Derivation of the Estimation Process

We are now ready to proceed with the estimation of the component displacement. Two approaches may be considered. The maximum likelihood and the maximum aposteriori. Given a classified measurement set $y=\left\{y_1^r, y_2^r, \ldots, y_K^r\right\}$, $r \in \left[-M,...,0,...,M\right]$, the maximum likelihood approach proceeds as:

$$\max_s P(y/s) = \max_s \prod_{k=1}^{K} P(y_k^r/s) \text{ , or}$$

$$\max_s \log P\left(y/s\right) = \max_{\mathrm{s}} \sum_{k=1}^{K} \log P\left(y_k^r/s\right), \text{ or}$$

$$\sum_{k=1}^{K} \frac{\partial}{\partial s} \log\left\{P(y_k^r/s)\right\} = 0 \qquad (5.10)$$

Observing that the classification y_k^r (with label l_r) may result from any actual class ω_i and assuming that the classification error only depends on the classifier and is independent of the actual displacement, we obtain:

$$\begin{aligned}P(y_k^r/s) &= \sum_i P(y_k^r, \omega_i/s) \\ &= \sum_i P(y_k^r/\omega_i, s) P(\omega_i/s) \\ &= \sum_i P(y_k^r/\omega_i) P(\omega_i/s) \\ &= \sum_i P(y_k^r/\omega_i) \frac{1}{\sqrt{2\pi}\sigma} \int_{l_i-T/2}^{l_i+T/2} e^{-(x-s)^2/2\sigma^2} dx \qquad (5.11) \\ &= \sum_i P(y_k^r/\omega_i) \int_{a_i}^{b_i} P(x/s) dx \\ &= \sum_i \alpha_i^{k,r} \int_{a_i}^{b_i} P(x/s) dx\end{aligned}$$

where $\alpha_i^{k,r} = P(y_k^r/\omega_i)$, the probability of misclassification.

It is proved in Appendix D1 that there exists a certain value $x_0 \in \left[\alpha_i, b_i\right]$ that describes the integral in the form:

$$\int_{a_i}^{b_i} e^{-\frac{(x-s)^2}{2\sigma^2}} dx = \left(b_i - a_i\right) e^{-\frac{(x_0-s)^2}{2\sigma^2}} \qquad (5.12)$$

and can be approximated by:

$$x_0 = \frac{a_i + b_i}{2} = l_i \qquad (5.13)$$

with l_i indicating the central point of this interval. Notice that this approximation holds at the shoulders of the distributions, but it overestimates the actual integral around the center of $P\left(\omega_i / s\right)$, i.e. for the class that correctly estimates the actual displacement *s*. In this case, the contribution to the overall $P\left(y_k^r / s\right)$ is amplified. Alternatively, far from the correct displacement of *s*, the contribution to $P\left(y_k^r / s\right)$ is attenuated. In this form, the approximation in (5.13) is validated since it emphasizes the contribution of classifications close to the correct displacement of *s*, while reducing the effect of far two erroneous classifications. Using (5.13) into (5.12) and introducing $P\left(y_k^r / s\right)$ into the middle form of (5.10), the maximum likelihood criterion is written as:

$$\max_s \left[\sum_k \log\left(T \sum_i a_i^{k,r} e^{-\frac{(l_i - s)^2}{2\sigma^2}} \right) \right] \tag{5.14}$$

or

$$\max_s \left[\sum_k \log\left(\sum_i a_i^{k,r} f_i(s) \right) \right] \tag{5.15}$$

where $f_i(s) = Te^{-\frac{(l_i - s)^2}{2\sigma^2}}$, and $0 < \sum_i a_i^{k,r} f_i(s) \le 1$, $0 < f_i(s) \le 1$ as probabilities. Since by Taylor expansion we have $\log(1+x) = x - \frac{x^2}{2} + \frac{x^3}{3} \mp \ldots$, for $-1 \le x \le 1$, we can approximate $\log(x) \cong x - 1$ for $0 < x \le 1$, so that:

$$\log\left(\sum_i a_i^{k,r} f_i(s) \right) \cong \sum_i a_i^{k,r} f_i(s) - 1 \tag{5.16}$$

and

$$\sum_i a_i^{k,r} \log f_i(s) \cong \sum_i a_i^{k,r} \left(f_i(s) - 1 \right) = \sum_i a_i^{k,r} f_i(s) - \sum_i a_i^{k,r} \tag{5.17}$$

From Equations (5.16), (5.17) we obtain the approximation

$$\log\left(\sum_i a_i^{k,r} f_i(s)\right) \cong \sum_i a_i^{k,r} \log\left(f_i(s)\right) - \left(1 - \sum_i a_i^{k,r}\right) \tag{5.18}$$

Introducing the latter approximation into (5.15) and expressing $f_i(s)$ in its exponential form, we get:

$$\max_s \left[-\sum_k \sum_i a_i^{k,r} \frac{\left(l_i - s\right)^2}{2\sigma^2} - \sum_k \left(1 - \sum_i a_i^{k,r}\right)\right] \tag{5.19}$$

or

$$\min_s \left[\sum_k \sum_i a_i^{k,r} \left(l_i - s\right)^2 \right] \tag{5.20}$$

Taking the derivative over s we obtain $\sum_k \sum_i a_i^{k,r} \left(l_i - s\right) = 0$. Thus, the maximum likelihood estimator is derived as:

$$\hat{s} = \frac{\sum_k \sum_i a_i^{k,r} l_i}{\sum_k \sum_i a_i^{k,r}} \tag{5.21}$$

where $\hat{s}$ is the estimate of s. Notice that if $a_i^{k,r} = P(y_k^r / \omega_i) = \delta\left(i - r\right)$, with $\delta\left(i - r\right)$ a δ-function, then $\hat{s} = \frac{\sum_k y_k^r}{K}$.

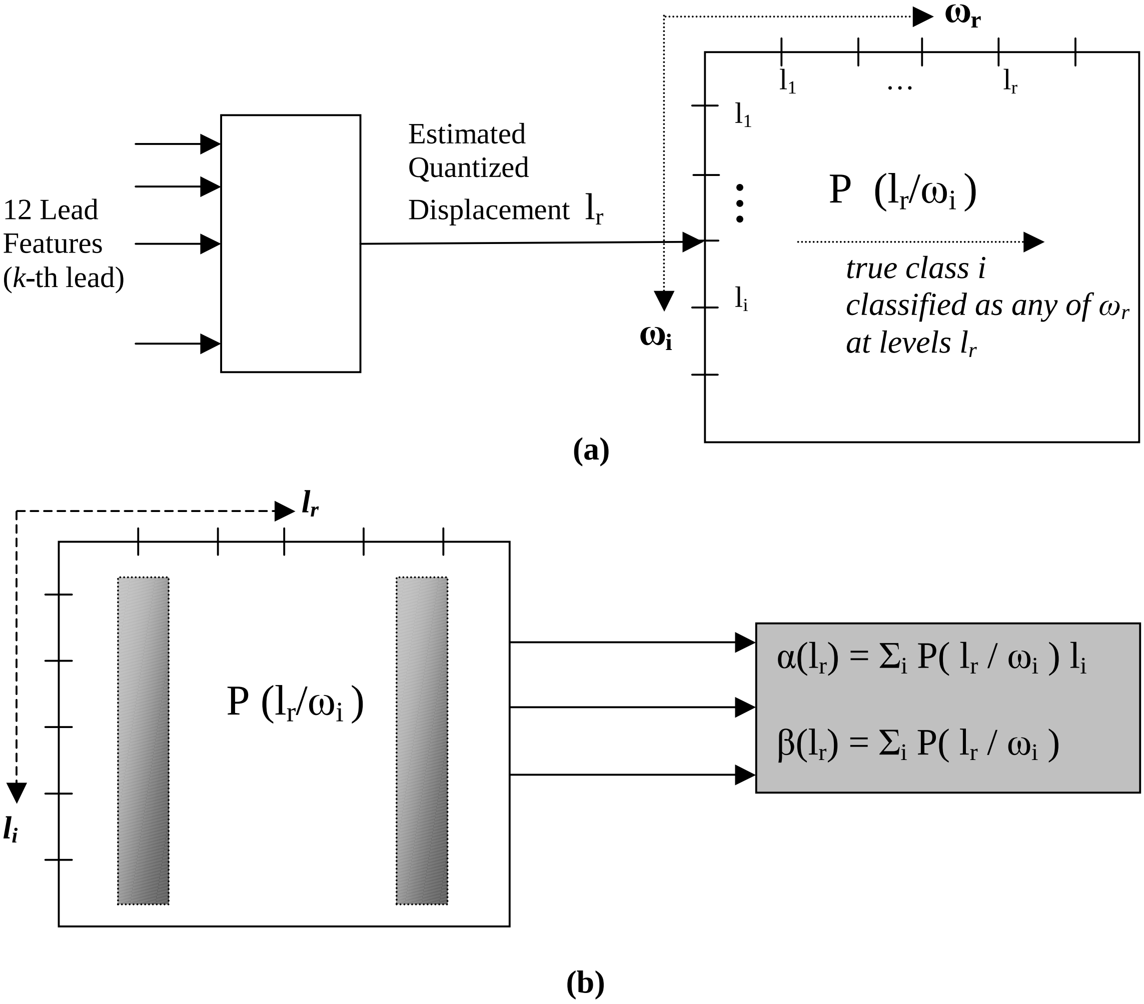
Remark 1 The value of measurement y_k^r may result from correct classification of feature vectors in classes ω_i, or from misclassification of neighboring classes. Our approach reflects the fact that the class ω_i is correctly assigned the quantized value l_i with probability $P(l_i / \omega_i) = P\left(y_k^i / \omega_i\right) = a_i^{k,i}$, but can also be assigned neighboring values with probability $P(l_r / \omega_i) = P\left(y_k^r / \omega_i\right) = a_i^{k,r}$.

Remark 2 Notice that the derivation of $\hat{s}$ is straightforward. The probabilities that formulate $\alpha_i^{k,r}$ are a priori calculated during the construction of the classifier and can be stored in a matrix A. The levels l_i are also predetermined. The estimator $\hat{s}$ reads for the *k*-th lead the classification output (l_r) and isolates the corresponding row of the A matrix, which is consequently multiplied with the vector $\mathbf{L} = \left[l_1, l_2, \ldots, l_{2M}\right]$, (where $2M$ is the number of classes), thus obtaining the numerator of $\hat{s}$. This process if further elucidated in next section.

Remark 3 It can be proved (the proof can be found in Appendix D2), that $E\{\hat{s}\} = s$, thus resulting in an almost unbiased estimator, with $\sigma^2\{\hat{s}\} = \frac{1}{K}\sigma^2$.

5.4.3 Implementation of Component Displacement Estimation

The classification process takes as input the feature vector extracted from each lead and produces an estimated quantized displacement y_k^r with label l_r, where the index *k* refers to the *k*-th lead of a component's side. During the training phase, the classifier produces a matrix $P(y_k^r / \omega_i) = P(l_r / \omega_i)$ for the probabilities of misclassification. Figure 5.5(a) illustrates the training process.



ω_r
l_1
…
l_r
l_1
Estimated Quantized Displacement l_r
12 Lead Features (*k*-th lead)
$P\ (l_r/\omega_i)$
true class i classified as any of ω_r at levels l_r
l_i
ω_i
l_r
$P\ (l_r/\omega_i)$
$\alpha(l_r) = \Sigma_i\ P(\ l_r / \omega_i\ )\ l_i$
$\beta(l_r) = \Sigma_i\ P(\ l_r / \omega_i\ )$
l_i

(a)

(b)

Figure 5.5 The classification procedure during the training phase; **(a)** computation of classification-error probabilities, **(b)** computation of quantities required by the estimator for each classification level l_r.

This process encodes the classification error for the specific type of classifier used. The quantization error is considered in the Bayesian estimation process during the testing of each component. In this phase, the same classifier classifies the feature vector of each lead as l_r. All leads from the component sides are uniformly classified according to the geometrical dependencies of Section 5.2. The classification results from all leads of the appropriate sides are used in the Bayesian estimator to produce the estimate of the component displacement $\hat{s}$. The contribution of each lead to the numerator and denominator of this estimator can be simply extracted from a look-up-table that is constructed during the training phase of the classifier, since it only depends on the classification error. More specifically, by the end of training phase, the classifier can compute the quantities:

$$\begin{aligned} \alpha(l_r) &= \sum_i P\left(y_k^r / \omega_i\right) l_i = \sum_i P(l_r / \omega_i) l_i \\ \beta(l_r) &= \sum_i P\left(y_k^r / \omega_i\right) = \sum_i P(l_r / \omega_i) \end{aligned} \tag{5.22}$$

and store them in the look-up-table for each classification level l_r as illustrated in Figure 5.5(b). Finally, the component displacement is estimated as:

$$\hat{s} = \frac{\sum_k \alpha(l_r)}{\sum_k \beta(l_r)} \tag{5.23}$$

The entire classification/estimation process in its operating stage (testing) is demonstrated in Figure 5.6.

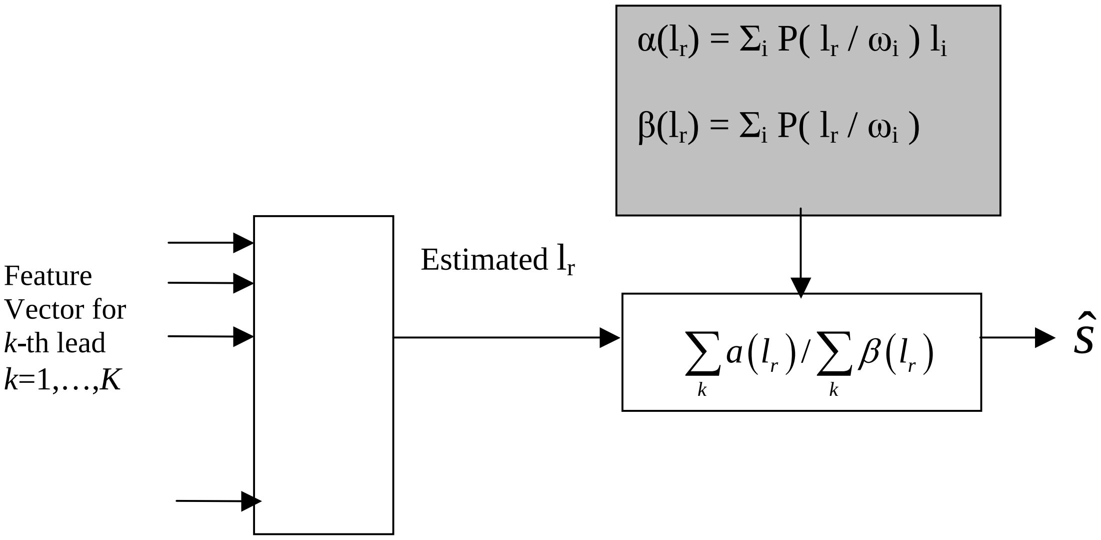

α(l_r) = Σ_i P(l_r / ω_i) l_i
β(l_r) = Σ_i P(l_r / ω_i)
Feature Vector for k-th lead k=1,…,K
Estimated l_r
Σ_k a(l_r)/Σ_k β(l_r)
ŝ

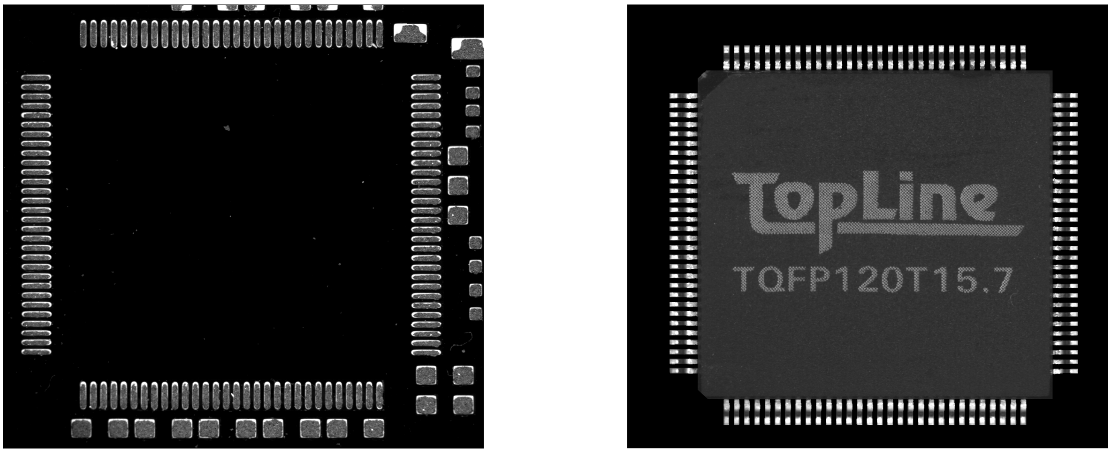
Figure 5.6 The classification/estimation procedure during its operation (testing phase).

Finally, having the accurate displacement measures for all sides, it is straight forward to apply the relations in Eqn (5.8) as to derive the component displacement and rotation parameters.

5.4.4 Monte Carlo Simulation for generating the PCB images

In this Ph.D. thesis we use the ***Monte Carlo simulation*** process [171-174], in order to generate lead samples with appropriate size and intensity distributions for trainining the classifiers. Since several actual images from the test bed are hard to be obtained we synthesize images with different translation of lead in reference to pad raising different types of faults. Our purpose is to simulate the entire component i.e. each side of the component, consequently the four global ROIs (Left, Right, Bottom, Top ROI). There are two major reasons affecting us to make this decision. First the generation of the entire image 1024×1024 pixels is time consuming because the pre-processing requires image enhancement techniques and second all the useful information is inside these four ROIs.

The data available are one set of 4 actual images with only pad and smeared solder paste and another set of 5 actual images with only the component QFP 120 in front of a dark background (as in Figures 5.7(a) and 5.7(b), respectively). Thus, the process of simulation of new images with controlling translations will be based on the their constituents ones which are individual pad with solder paste and individual lead.



(a) **(b)**

Figure 5.7 Pad and QFP component images; **(a)** only pad and smeared solder paste, **(b)** only the component QFP 120.

One could claim that the simulation of the ROIs is accomplished performing a superposition of figure 5.7(b) over figure 5.7(a). But, this cannot be applied directly due to the varying dimensions (length and height) and intensity levels of the lead area and so we cannot easily and accurately the translation of the chip over the pad area. Also the training of the classifiers is based on images like Figure 5.2 (c) (i.e., cutted individual lead image after the placement of QFP 120 component on its own pad area). So, better results are obtained if the superposition is performed using individual pad-images and lead-images like the ones in Figures 5.8(a) and 5.8(b), respectively).

(a) **(b)**

Figure 5.8 Individual pad and lead-images; **(a)** individual pad-image **(b)** individual lead-image

It is apparent that the leads inside a global ROI have varying dimensions and intensity levels. Also the pads have varying intensity levels due to the different distribution of the smeared solder paste. In order to simulate a ROI correctly and approximate a real case we must preserve these observations. Therefore for every lead-image we find length, height and mean intensity and for every pad-image we find mean intensity (providing from the specifications that pads have constant dimensions), considering them as characterisics. Simulating a ROI we must draw 30 pad-images and 30 lead-images with random characteristics. This procedure is accomplished using the Monte Carlo simulation process. The displacement is constant for the entire component but the relative displacement of each lead over its own pad is a function of the considered side (either left or right or bottom or top side). The general procedure we follow is given below [52]:

- Firstly, we extract lead/pad regions from actual images and then we compute the above-mentioned characteristics.
- Secondly, we classify each pad/lead according to its characteristics and then we derive statistics from the data. In addition we develop a ***database*** comprising all the extracted images.

- Thirdly, we use Monte Carlo simulation to select leads/pads according to the specified distributions and create component ROIs.

5.4.4.1 Modeling and Implementation of Monte Carlo Simulation

Simulation means driving a model of a system with suitable inputs and observing the corresponding outputs. Simulation models can be distinguished between *static* or *dynamic*, *deterministic* or *stochastic*, *discrete* or *continuous* [230]. A static simulation model is a representation of a system at a particular time. Monte Carlo simulation models are typically of this type.

We define Monte Carlo simulation to be a scheme employing random numbers, that is U(0,1) (uniform distributed) random variables which is used for solving certain stochastic or deterministic problems where the passage of time plays no substance role.

Monte Carlo simulation is widely used for solving certain problems in statistics which are not analytically tractable. For example, it has been applied to estimate the integral [172 – 173]

$$I = \int_a^b g(x)dx \tag{5.24}$$

where $g(x)$ is a real valued function which is not analytically integrable. To see how this static problem can be approached by Monte Carlo simulation let *Y* be the random variable (*b*-*a*)*g*(*X*) where *X* is a continuous random variable uniformly distributed on the interval [*a*, *b*]. Then it can be shown that the expected value of *Y* is given by

$$\begin{aligned} E(Y) &= E\left[(b-a)g(X)\right] \\ &= (b-a)E\left[g(X)\right] \\ &= (b-a)\int_a^b g(x)f_x(x)dx \\ &= (b-a)\frac{\int_a^b g(x)dx}{b-a} = I \end{aligned} \tag{5.25}$$

where $f_x(x) = 1/(b-a)$ is the probability density function of a $U(a,b)$ random variable. Thus the problem evaluating the integral has been reduced to one of a estimating the expected value *E*(*Y*). In particular we shall estimate *E*(*Y*) by the sample mean

$$\overline{Y}(n) = \frac{\sum_{i=1}^{n} Y_i}{n} = (b-a)\frac{\sum_{i=1}^{n} g(X_i)}{n} \tag{5.26}$$

because it is unbiased estimator of *E*(*Y*) where $X_1, X_2, \ldots, X_n$ are ***independent identically distributed*** (IID) $U(a,b)$ random variables.

Model analysis with Monte Carlo simulation is similar to ‘what if’ scenarios in that it generates a number of possible scenarios. However, it goes one step further by effectively accounting for every possible value that each random variable could take and weights each possible scenario by the probability of its occurrence. In order to model each uncertain variable within a model we must determine its probability distribution. Monte Carlo technique involves the random sampling of each probability distribution within the model to produce thousands of scenarios (also called iterations). Each probability distribution is sampled in a manner that reproduces the distribution’s shape [173]. The distribution of the values calculated for the model outcome reflects the probability of the values that could occur. Monte Carlo simulation offers many advantages as:

- The distribution of the model’s random variables does not have to be approximated in any way.
- Correlations and other inter- dependencies can be modelled.
- Greater levels of precision can be achieved by simply increasing the number of iterations.

In our situation the procedure followed in order to perform Monte Carlo simulation is [52]:

- Select distribution for lead (length, height, mean intensity) using the histogram procedure
- Select distribution for pad (mean intensity) using the histogram procedure
- Perform Maximum Likelihood Estimators procedure to find the optimum parameters for each distribution
- Use Chi-square goodness-of-fit test to attest if the chosen distribution fits the data adequately
- Random number generator U(0,1)
- Random variable generation by sampling each distribution.

5.5 Results

In our experiments, we use the Bayes and the LVQ classifiers both with and without a K-L transform for feature reduction. The Bayes classifier and LVQ neural network classifier are well-established and quite successful techniques in pattern classification. The theoretical background of these classifiers has been introduced in Chapter 4. the LVQ neural network architecture was defined by the feature vector size training set size and output class mapping. In particular for use with 12 geometric features the LVQ input layer consisted of 12 neurons. In accordance to LVQ theory the hidden competitive layer contained neurons, equal to the number of training set cases. In the output layer for 5 classes (2 pixel shift precision) 5 output neurons were needed. Accordingly discrimination of 7 classes required 7 output neurons. The model was trained for 1000 epochs with a learning parameter a=0.09.

For designing and training the classifiers we employ the *Monte Carlo simulation* process, in order to generate lead samples with appropriate size and intensity distributions. Notice that the size of leads on an actual component is not constant, the size of solder paste varies from board to board, and the illumination conditions do not remain fixed in a realistic production environment. The reference for the distributions used in Monte Carlo simulations is obtained from one set of 4 actual images with only pad and smeared solder paste and another set of 5 images with only the component QFP 120 in front of a dark background (as in Figures 5.7(a) and 5.7(b), respectively). We use the aforementioned images for the creation of Monte Carlo samples and training, whereas we have available 20 new (post-placement) component images (as in Figure 5.2 (a)), obtained from the actual placement environment, that are used only for testing; never for training. The Monte Carlo process simulates variable size and illumination conditions and implements component displacements on the pad regions, which are employed in training. In fact, we use 13 classes of component displacements i.e., {-6, -5,…, 5, 6} pixels. Each displacement involves three neighboring cases for testing (e.g., class –4 involves displacements {-4.2, -4, -3.8}). Both directional displacements have been considered namely horizontal and vertical. Notice that a component shift in the *x*-direction engages leads on down and up sides (on their corresponding cross-axial *v* direction), whereas a component displacement in the *y*-direction affects leads on left and right sides. Thus, for testing the 13 classes of lead shifts, each class contains 3x2=6 directional cases and 30 leads per side for two sides, making 360 lead samples per lead-shift class. As stated in the previously, 12 features per lead formulate the feature vector that forms the input to each classifier.

The classifiers are trained for 5 and 7 classes. The first case involves classes {-6,-3,0,+3,+6} whereas the second case considers training on classes {-6,-4,-2,0,+2,+4,+6}. Thus, for training we use 120 lead samples per class resulting in totally 600 lead samples for the 5-class training set and the 840 lead samples for the 7-class training set correspondingly. These two cases study the ability of the classifiers to discriminate classes in the feature space separated by 3 and 2 pixels apart, respectively. We do not consider training on all 13 classes, since all these classes are not separable in the 12 dimensional feature space defined. In our approach we reduce our expectation on the quantization error, since the classification is followed by the Bayesian estimation scheme designed to account for and take care of such errors. Table 5.6 shows the Bayes probabilities (%) (i.e., the probability of a correct class ω_i to be classified as l_r). The probabilities are computed via a ***jack-knifing process***, which approximates the true probabilities of classification [9], [171]. This process sweeps along all sample vectors and every time extracts one sample out of the training data. It trains the classifier with all other vectors and classifies the extracted vector that has not been seen by the classifier. Finally, it computes the percentage of correct and incorrect classifications for all data available, in order to derive the classification probabilities of the classifier. Bayes-1 incorporates a K-L transform, while Bayes-2 operates on the entire feature space without transformations. Each row presents the probabilities of classification for the corresponding true class ω_i to any other class ω_r. Table 5.7, gives the same information for the LVQ classifier. In general, we notice that the probability of correct classification is particularly high in the case of 5 classes, indicating an improved ability of the classifiers to discriminate per 3 pixels shift rather than 2 pixels shift. Moreover, the use of K-L deteriorates the performance of classifiers in the specific problem. In all cases, the probabilities of misclassification are distributed around the correct class. For example, class +3 in the Bayes-2 classifier is misclassified as class 0 or class +6, but not as a class with negative shift. The sources of misclassification are further considered in later parts of this section.

In order to test the generalization ability of our combined classification-estimation process, we proceed with testing the classifiers on all available classes. Thus, we first train the classifiers with the corresponding classes, for 5 or 7-class quantization. Then, we classify all samples in every one of the 13 classes and use our estimation process to derive a single displacement for all leads in each class. Table 5.8 gives the entire class displacement estimation using Bayes classifiers, with and without K-L transform, for 5 and 7 class training. The corresponding results for the LVQ classifiers are presented in Table 5.9. Notice that the classifiers are trained on 5 or 7 out of 13 classes and are used to classify all 13 classes, interpolating in-between via our estimation approach. The Bayes classifier without K-L derives the best estimates, in terms of their proximity to the actual values. Moreover, the effects of larger quantization error in 5-class training as compared to that of 7-class training are diminished, due to the corresponding larger probabilities of correct classification. From these tests we conclude that the Bayes classifier (without K-L) on 5-class quantization give the best overall estimation results and it will be employed for further testing of our approach. At this end, we indicate a small bias of the estimator, which is partially due to the non-uniform distribution of the classification probabilities.

In the sequel we test our approach on a set of 20 new component images from the actual placement environment. Ten actual boards with different shifts and rotations are given, with two images from each case. Each individual case is controlled by the placement machine and conveys the limited accuracy of placement. Notice that these images are used only for testing; the classifiers have never seen these images before. The results are given in Table 5.10. The first column indicates the number of the image, whereas the second column gives its displacement information from the placement machine. For each image, the first row presents the displacement results from our classification approach. In order to test these results with some accurate measure, we zoom on each image and manually measure the displacement of leads on each side of the component. These manual measurements are given in the second row associated with each image. Based on the relations of Subsection 5.2.3, we compute the average horizontal and vertical displacement of the component, as well as its rotation. Furthermore, in order to provide an additional rule for comparing the results with respect to component rotation, we also measure the total lead shift for one side (from end-to-end leads) along the axial *(u)* direction of leads. This is the measure T_v in Figure 5.3, which adds up to twice the cross-axial *v*-displacement of each lead due to rotation. This additional measure is presented in the last column, along with some manual measurements in parenthesis. In this consideration, the positive rotation angle is measured in the counter clock-wise (CCW) direction and the positive side-to-side displacement along the u-axis of the leads in each side is defined on the basis of the positive CCW rotation. For the computation of the angle, the width of the component form its center (variable R in Eqn (5.8)) is manually measured for all sides.

From table 5.10 it become immediately clear that the accuracy of the placement machine is quite limited and our proposed estimation scheme can effectively improve this accuracy. Consider, for instance, the case of t01_2, which involves considerable displacement on both the horizontal and the vertical direction, while the controls of the placement machine indicate no shift at all. Similarly, in the case of t07_1, the placement machine indicates no rotation, while both our classification scheme and the manual measurements indicate rotation larger than 0.15^{o}. Comparing now the results of the proposed classification scheme and the manual measurements, we can generally observe that the classification results are within 0.6 pixels from the manual measurements (see Table 5.13). It is interesting to consider the case of t08_1, with respect to the total (end-to-end) axial lead displacement in the last column. The classification approach gives a more accurate estimate than the result through manual measurements of the cross-axial lead displacement on each side. This observation indicates that (as expected) the manual measurements on individual leads cannot provide consistent accuracy, especially when the component engages rotation in addition to translation.

Table 5.6 Bayes Probabilities (%)

a) Bayes-1 classifier for 5 classes

-6 pixels shift	-3 pixels shift	0 pixels shift	+3 pixels shift	+6 pixels shift
97.47	1.96	0.00	0.00	0.56
3.67	**92.93**	2.54	0.00	0.84
0.00	2.82	**93.78**	2.25	1.12
0.00	0.00	5.14	**87.42**	7.42
0.29	1.16	0.29	1.45	**97.09**

b) Bayes-2 classifier for 5 classes

-6 pixels shift	-3 pixels shift	0 pixels shift	+3 pixels shift	+6 pixels shift
97.75	2.24	0.00	0.00	0.00
2.25	**94.35**	3.10	0.00	0.28
0.00	2.82	**94.35**	1.69	1.12
0.00	0.00	2.57	**93.14**	4.28
0.58	0.00	0.00	0.87	**98.25**

c) Bayes-1 classifier for 7 classes

-6 pixels shift	-4 pixels shift	-2 pixels shift	0 pixels shift	+2 pixels shift	+4 pixels shift	+6 pixels shift
92.71	6.44	0.84	0.00	0.00	0.00	0.00
16.76	**73.01**	9.37	0.00	0.00	0.00	0.85
0.00	7.51	**80.05**	10.98	0.00	0.00	1.44
0.00	0.00	9.03	**80.22**	9.32	0.00	1.41
0.57	0.00	0.00	12.39	**73.77**	10.95	2.30
0.00	0.00	0.28	0.28	4.89	**73.77**	20.74
0.58	0.00	1.45	0.00	0.00	4.06	**94.18**

d) Bayes-2 classifier for 7 classes

-6 pixels shift	-4 pixels shift	-2 pixels shift	0 pixels shift	+2 pixels shift	+4 pixels shift	+6 pixels shift
92.99	6.44	0.56	0.00	0.00	0.00	0.00
10.51	**81.82**	7.38	0.00	0.00	0.00	0.28
0.00	10.69	**75.72**	13.29	0.00	0.00	0.28
0.00	0.00	5.08	**85.87**	7.90	0.28	0.84
0.00	0.00	0.00	11.52	**77.80**	8.64	2.01
0.00	0.00	0.28	0.00	3.17	**85.87**	10.66
0.29	0.00	0.58	0.00	0.29	7.26	**91.86**

Table 5.7 LVQ Probabilities (%)

a) LVQ-1 classifier for 5 classes

-6 pixels shift	-3 pixels shift	0 pixels shift	+3 pixels shift	+6 pixels shift
94.67	4.76	0.00	0.00	0.56
2.25	**94.06**	3.10	0.00	0.56
0.00	13.55	**75.70**	9.88	0.84
0.00	0.00	1.42	**95.71**	2.85
0.57	0.00	1.15	3.18	**95.07**

b) LVQ-2 classifier for 5 classes

-6 pixels shift	-3 pixels shift	0 pixels shift	+3 pixels shift	+6 pixels shift
93.27	6.16	0.00	0.56	0.00
5.64	**90.67**	3.10	0.00	0.56
0.00	7.90	**78.24**	12.99	0.84
0.00	0.00	0.57	**95.42**	4.00
0.57	0.00	1.15	3.47	**94.78**

c) LVQ-1 classifier for 7 classes

-6 pixels shift	-4 pixels shift	-2 pixels shift	0 pixels shift	+2 pixels shift	+4 pixels shift	+6 pixels shift
78.43	21.00	0.00	0.00	0.00	0.00	0.56
9.09	**73.86**	16.19	0.00	0.00	0.00	0.85
0.00	15.31	**74.85**	6.64	1.73	0.57	0.86
0.00	3.38	25.42	**53.10**	17.23	0.00	0.84
0.57	0.00	0.00	8.35	**81.55**	8.64	0.86
0.00	0.00	0.28	0.00	10.95	**77.52**	11.23
0.57	0.00	0.57	0.57	00.28	12.75	**85.21**

d) LVQ-2 classifier for 7 classes

-6 pixels shift	-4 pixels shift	-2 pixels shift	0 pixels shift	+2 pixels shift	+4 pixels shift	+6 pixels shift
83.47	15.96	0.00	0.00	0.56	0.00	0.00
13.35	**76.42**	9.37	0.00	0.00	0.28	0.56
0.00	10.11	**78.90**	8.38	1.44	0.86	0.28
0.00	0.00	20.33	**57.06**	21.75	0.00	0.84
0.57	0.00	0.00	3.74	**80.40**	14.40	0.86
0.00	0.00	0.00	0.28	10.95	**74.35**	14.40
0.57	0.00	0.00	0.86	00.57	16.81	**81.15**

Two cases where the estimation approach fails to produce results very close to the manual measurements are the t02_1 (top side) and the t09_1 (left side). The first case fails due to insufficient training data in the classification process, whereas the second case fails due to measurement errors in the computation of the feature vector. At this end, notice that our classification approach has been trained to produce displacements of up to +6 or –6 pixels. In case we encounter displacements larger than these limits, our approach of course fails to produce reliable displacement estimates. Such are the cases of t09_2, t10_1, and t10_2. Even in such cases, though, the rotation is estimated more accurately by our approach than by the manual measurements on sides, as it is indicated by the end-to-end axial lead-shift in the last column.

Table 5.8 Displacement Estimation of Entire Class (~360 samples per class) with Bayes Classifiers

Bayes	B1_5Classes with K-L	B2_5Classes	B1_7Classes with K-L	B2_7Classes
Class 0	0.152	0.025	0.108	0.071
Class -1	-0.717	-0.972	-0.861	-0.742
Class +1	0.628	0.518	0.821	0.667
Class -2	-2.167	-2.415	-1.721	-1.850
Class +2	1.880	1.980	1.881	1.859
Class -3	-2.825	-2.889	-2.635	-3.098
Class +3	3.010	3.015	2.875	3.331
Class -4	-3.628	-3.757	-3.970	-3.936
Class +4	4.241	3.935	4.069	4.053
Class -5	-5.036	-5.213	-4.990	-4.976
Class +5	5.284	5.401	4.828	4.750
Class -6	-5.765	-5.838	-5.495	-5.641
Class +6	5.430	5.731	5.169	5.434

Table 5.9 Displacement Estimation of Entire Class (~360 samples per class) with LVQ Classifiers.

LVQ	L1_5Classes with K-L	L2_5Classes	L1_7Classes with K-L	L2_7Classes
Class 0	-0.062	0.219	-0.289	0.068
Class -1	-1.707	-1.536	-1.376	-1.141
Class +1	1.271	1.426	1.131	1.290
Class -2	-2.432	-2.568	-1.904	-1.752
Class +2	2.448	2.491	1.893	1.956
Class -3	-2.710	-2.981	-2.759	-2.782
Class +3	2.862	2.800	2.841	2.817
Class -4	-3.393	-3.530	-3.646	-3.929
Class +4	3.361	3.230	3.870	3.914
Class -5	-4.626	-4.753	-4.371	-4.675
Class +5	4.770	4.452	4.576	4.642
Class -6	-5.634	-5.524	-5.181	-5.309
Class +6	5.542	5.566	5.150	5.137

Finally, in order to assess our results for displacement estimation of entire class presented in Tables 5.8 and 5.9 we calculate the *absolute value* of the difference between the actual class and its estimated value (i.e. ***ABS(Class – Estimation)***) per class for Bayes and LVQ classifiers. Then we compute the *average value* of *ABS(Class – Estimation)* values for all 13 classes. These assessment results are presented in Tables 5.11 and 5.12 for the Bayes and LVQ classifiers, respectively. The Bayes classifier achieves its best performance for the 5-class formulation, whereas LVQ attains its minimum error for the 7-class, both without the use of K-L.

In the sequel, in order to assess our results presented in Table 5.10 on actual component images with prior known shifts and rotations we first analyze the images manually, as to compute displacements that can be used as “golden-truth” values. From the first examination of the available images, it becomes evident that the placement machine attributes for displacement and rotation are far from accurate. For this reason we resort to the tedious process of manual measurements, which is far more accurate. Subsequently, we compute the *absolute difference* between the estimated and the manual measurements (i.e. ***ABS(Estimation – Manual Measurement)***), as well as the *absolute difference* between the placement machine and the manual measurements (i.e. ***ABS(Placement Machine Measurement – Manual Measurement)***) for all boards and all global ROIs. Then we consider statistical measures on each difference set in order to access the accuracy of estimation. The abovementioned assessment results are presented in Tables 5.13 and5.14 for our estimated and placement machine measurements, respectively. In order to compare the above two distributions we use the ***quantile – quantile plot*** (q-q plot) [171]. Typically, a q-q plot (sometimes called an ***empirical quantile plot***) is used to determine whether two random samples are generated by the same distribution. The q-q plot was originally proposed by Wilk and Gnanadesikan [171] to visually compare two distributions by graphing the quantiles of one versus the quantiles of the other. Assume that we have two data sets consisting of univariate measurements. We denote the order statistics for the first data set by $x_1, x_2, ..., x_n$ and the order statistics for the second set by $y_1, y_2, ..., y_m$, with $m \le n$. In our case, the sizes of the data sets are equal, so *m=n*. In this case we plot as points the sample quantiles of one data set (i.e. *ABS(Estimation – Manual Measurement)* values) versus the other data set (i.e. *ABS(Placement Machine Measurement – Manual Measurement)* values). The obtained q-q plot is illustrated in Figure 5.8. The q-q plot follows approximately a straight line (except some outliers), indicating that the data sets come from the same distribution (i.e. Gaussian distribution). Furthermore, the regression line is with slope close to 0.3, at least for machine attributes smaller than 4 pixels, indicating that the estimated results are at the order of three times more accurate than those of the actual placement machine.

Overall, in terms of time requirements of our approach, the processing, segmentation and feature extraction (12 features) takes about 0.75sec for processing the entire QFP120 component, or 6.3msec per individual lead, on a Pentium II processor at 366MHz. The combined classification/estimation step for testing the entire QFP120 component takes 0.098sec and 0.011sec when using the Bayes and the LVQ classifiers, respectively. The classification of a single feature vector in the testing phase takes 0.81msec for the Bayes classifier, whereas it only requires 0.09msec for the LVQ classifier. The training phase for 5 classes and 1760 feature vectors takes 24.2msec for the Bayes classifier and increases to 1.65 sec for the LVQ classifier, which requires iterative training.

Table 5.10. Test results on actual component images with a priori known shifts and rotations

Board no	Board Mount Description	Left Side	Right Side	Bottom Side	Top Side	Horiz. Shift	Vert. Shift	*Rotation CCW in Degrees*	*Pixel Displacement in entire Side due to Rotation*
t01_1	Golden board; no shifts	-0.226	-2.93	3.067	0.173	-1.578	1.62	-0.184	-2.799
		0.1	-2.5	2.5	0.5	-1.2	1.5	-0.151	-2.3 (-2)
t01_2		3.273	3.287	0.734	3.176	3.28	1.955	0.081	1.228
		3	3.5	1.5	3	3.25	2.25	0.065	1 (+1.5)
t02_1	20 micron right shift in Y direction	-3.035	-3.133	2.252	0.188	-3.084	1.22	-0.071	-1.081
		-2	-2.5	2	1.5	-2.25	1.75	-0.032	-0.5
t02_2		2.124	2.124	2.766	2.974	2.124	2.87	0.006	0.104
		2.5	2	2.5	2.5	2.25	2.5	-0.016	-0.25
t03_1	60 micron right shift in Y-direction	-5.158	-5.863	1.645	-1.181	-5.5105	0.232	-0.116	-1.765
		-4.5	-5	1.5	0	-4.75	0.75	-0.066	-1 (-1)
t03_2		-1.570	-1.264	-0.113	2.019	-1.417	0.953	0.080	1.219
		-0.5	-0.5	-0.1	1	-0.5	0.45	0.036	0.55 (+1)
t04_1	100 micron right shift in Y-direction	-4.987	-5.36	1.757	-0.026	-5.1735	0.8655	-0.071	-1.078
		-5.5	-6	2.5	0.5	-5.75	1.5	-0.082	-1.25
t04_2		-2.628	-2.93	1.445	1.646	-2.779	1.5455	-0.003	-0.050
		-1.5	-1.5	1.5	1.5	-1.5	1.5	0	0
t05_1	40 micron right shift in X-direction & 60 micron left shift in Y-direction'	1.255	0.366	0.175	-0.12	0.8105	0.0275	-0.039	-0.592
		1.1	0.5	0.1	0.1	0.8	0.1	-0.019	-0.3
t05_2		5.786	5.693	-0.016	-2.131	5.7395	-1.0735	-0.073	-1.104
		7	5	0.5	-1.5	6	-0.5	-0.132	-2
t06_1	20 micron right shift in X-direction & 20 micron left shift in Y-direction'	-0.317	-2.347	-0.307	-0.317	-1.332	-0.312	-0.067	-1.020
		0.5	-1.5	-0.5	-0.5	-0.5	-0.5	-0.066	-1
t06_2		5.149	3.298	-0.216	-0.224	4.2235	-0.22	-0.061	-0.929
		5	4	0	0	4.5	0	-0.033	-0.5
t07_1	120 micron left shift in X-direction & 60 micron right shift in Y-direction'	-3.225	-5.336	5.786	2.761	-4.2805	4.2735	-0.169	-2.568
		-3	-6	6	2.5	-4.5	4.25	-0.214	-3.25 (-3)
t07_2		-0.285	-0.419	5.786	5.786	-0.352	5.786	-0.004	-0.067
		-0.5	0.1	6	6.5	-0.2	6.25	0.036	0.55 (0.1)
t08_1	0.2 rotation angle (3 pixels shift) & 50 micron paste shift	-2.729	-5.766	0.175	-2.93	-4.2475	-1.3775	-0.202	-3.071
		-0.5	-5	0.5	-3	-2.75	-1.25	-0.263	-4 (-3)
t08_2		2.974	0.681	-0.518	-2.93	1.8275	-1.724	-0.155	-2.352
		3.5	1	-0.5	-2.5	2.25	-1.5	-0.148	-2.25 (-2)
t09_1	0.2 rotation angle (3 pixels shift)	0.685	-5.182	3.287	-2.93	-2.2485	0.1785	-0.398	-6.042
		1.5	-5.5	4	-2	-2	1	-0.428	-6.5 (-5.5)
t09_2		5.421	-0.618	5.786	-4.296	2.4015	0.745	-0.531	-8.060
		10	-0.5	7.5	-3.5	4.75	2	-0.709	-10.75 (-8)
t10_1	0.6 rotation angle (9 pixels shift)	3.07	-5.46	5.786	-5.561	-1.195	0.1125	-0.655	-9.938
		3	-9.5	7	-5.5	-3.25	0.75	-0.824	-12.5 (-9)
t10_2		5.593	-2.93	5.786	-2.93	1.3315	1.428	-0.568	-8.619
		8	-2.5	8	-2.5	2.75	2.75	-0.692	-10.5 (-8)

Table 5.11 ABS(Class – Estimation) for Bayes Classifiers

Bayes	B1_5Classes with K-L	B2_5Classes	B1_7 Classes with K-L	B2_7Classes
Class 0	0.152	0.025	0.108	0.071
Class -1	0.283	0.028	0.139	0.258
Class +1	0.372	0.482	0.179	0.333
Class -2	0.167	0.415	0.279	0.150
Class +2	0.120	0.020	0.119	0.141
Class -3	0.175	0.111	0.365	0.098
Class +3	0.010	0.015	0.125	0.331
Class -4	0.372	0.243	0.030	0.064
Class +4	0.241	0.065	0.069	0.053
Class -5	0.036	0.213	0.010	0.024
Class +5	0.284	0.401	0.172	0.250
Class -6	0.235	0.162	0.505	0.359
Class +6	0.570	0.269	0.831	0.570
Average value of ABS(Class-Estimation) for 13 Classes	**0.232**	**0.188**	**0.225**	**0.207**

Table 5.12 ABS(Class – Estimation) for LVQ Classifiers

LVQ	L1_5Classes with K-L	L2_5Classes	L1_7 Classes with K-L	L2_7Classes
Class 0	0.062	0.219	0.289	0.068
Class -1	0.707	0.536	0.376	0.141
Class +1	0.271	0.426	0.131	0.290
Class -2	0.432	0.568	0.096	0.248
Class +2	0.448	0.491	0.107	0.044
Class -3	0.290	0.019	0.241	0.218
Class +3	0.138	0.200	0.159	0.183
Class -4	0.607	0.470	0.354	0.071
Class +4	0.639	0.770	0.130	0.086
Class -5	0.374	0.247	0.629	0.325
Class +5	0.230	0.548	0.424	0.358
Class -6	0.366	0.476	0.819	0.691
Class +6	0.458	0.434	0.850	0.863
Average value of ABS(Class-Estimation) for 13 Classes	**0.386**	**0.415**	**0.327**	**0.275**

Table 5.13 ABS (Estimation-Manual Measurement)

Board no	Board Mount Description	Left Side	Right Side	Bottom Side	Top Side
t01_1	Golden board; no shifts	0.326	0.43	0.567	0.327
t01_2		0.273	0.213	0.766	0.176
t02_1		1.035	0.633	0.252	1.312
t02_2		0.376	0.124	0.266	0.474
t03_1	60 micron right shift in Y-direction	0.658	0.863	0.145	1.181
t03_2		1.07	0.764	0.013	1.019
t04_1	100 micron right shift in Y-direction	0.513	0.64	0.743	0.526
t04_2		1.128	1.43	0.055	0.146
t05_1	40 micron right shift in X-direction &	0.155	0.134	0.075	0.22
t05_2		1.214	0.693	0.516	0.631
t06_1	20 micron right shift in X-direction & 20 micron left shift in Y-direction'	0.817	0.847	0.193	0.183
t06_2		0.149	0.702	0.216	0.224
t07_1	120 micron left shift in X-direction & 60 micron right shift in Y-direction'	0.225	0.664	0.214	0.261
t07_2		0.215	0.519	0.214	0.714
t08_1	0.2 rotation angle (3 pixels shift)	2.229	0.766	0.325	0.07
t08_2		0.526	0.319	0.018	0.43
t09_1	0.2 rotation angle (3 pixels shift)	0.815	0.318	0.713	0.93
t09_2		4.579	0.118	1.714	0.796
t10_1	0.6 rotation angle (9 pixels shift)	0.07	4.04	1.214	0.061
t10_2		2.407	0.43	2.214	0.43
VARIANCE	0.595031				

Table 5.14 ABS (Placement Machine Measurement – Manual Measurement)

Board no	Board Mount Description	Left Side	Right Side	Bottom Side	Top Side
t01_1	Golden board; no shifts	0.1	2.5	2.5	0.5
t01_2		3	3.5	1.5	3
t02_1	20 micron right shift in Y direction	1	1.5	2	1.5
t02_2		3.5	3	2.5	2.5
t03_1	60 micron right shift in Y-direction	1.5	2	1.5	0
t03_2		2.5	2.5	0.1	1
t04_1	100 micron right shift in Y-direction	0.5	1	2.5	0.5
t04_2		3.5	3.5	1.5	1.5
t05_1	40 micron right shift in X-direction &	1.9	2.5	2.1	2.1
t05_2		4	4	2.5	0.5
t06_1	20 micron right shift in X-direction & 20 micron left shift in Y-direction'	0.5	2.5	0.5	0.5
t06_2		4	3	1	1
t07_1	120 micron left shift in X-direction & 60 micron right shift in Y-direction'	0	3	0	3.5
t07_2		2.5	2.9	0	0.5
t08_1	0.2 rotation angle (3 pixels shift)	3.5	2	2.5	0
t08_2		0.5	4	3.5	0.5
t09_1	0.2 rotation angle (3 pixels shift)	1.5	2.5	1	1
t09_2		7	2.5	4.5	0.5
t10_1	0.6 rotation angle (9 pixels shift)	6	0.5	2	3.5
t10_2		1	6.5	1	6.5
VARIANCE	**2.472960**				

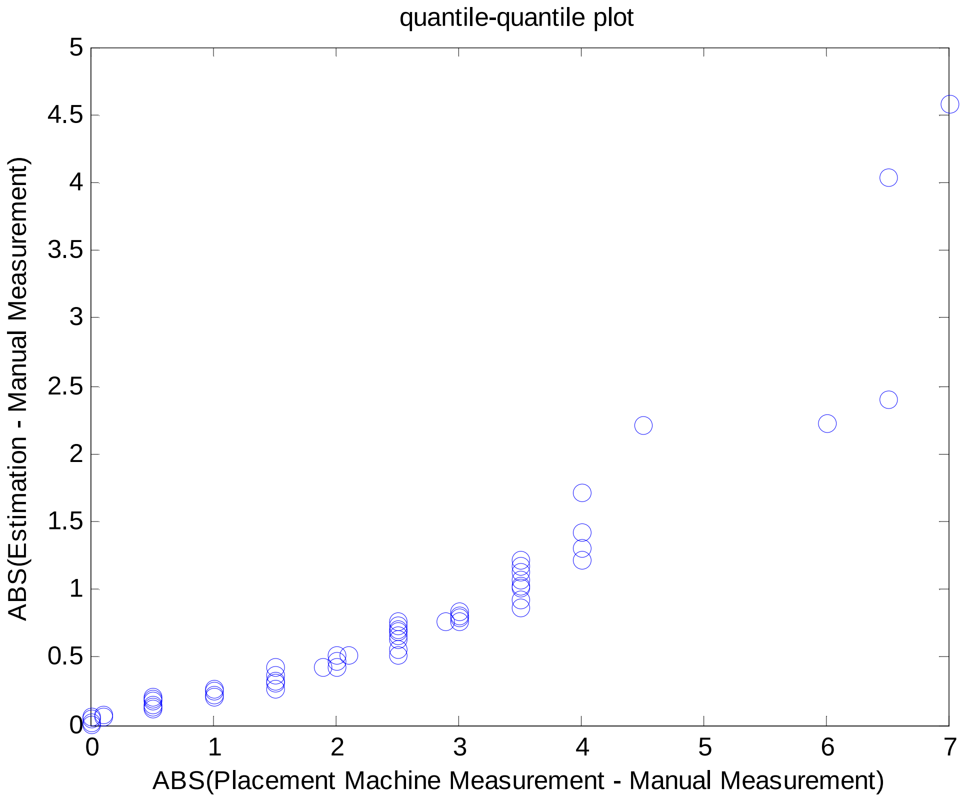


Figure 5.8 q-q plot of ABS(Estimation – Manual Measurement) versus ABS(Placement Machine Measurement – Manual Measurement)

Chapter 6

Data-Space Reduction using topological and projection features for Component Quality Inspection

6.1 Introduction to Data-Space Reduction

In this chapter we consider two approaches to overcome computational complexity of classical machine vision quality inspection of SMDs on a PCB presented in Chapter 5. The first employs associative memories to implement the reduced information content in image intensity levels. The idea is to compare the edge structure of a lead image with that of stored fundamental patterns. The second scheme compresses the data space by considering only an image projection function of the data. A non-linear filter based on high order neural networks is used to encode the characteristics of each projection function. Both methodologies are tested on real industrial PCB images. The quality of inspection slightly deteriorates while the computational time is significantly reduced, when compared to classical visual inspection techniques. **This research has been published in [28].**

One of the most difficult and important problems in automating machine vision is to understand what kind of information is required and how is translated into measurements or ***features*** extracted from images. A descriptive set of uncorrelated features can drastically boost the classification success rate. Some of the most ordinary inspected features (dimensional, structural, etc.) are reported in [2]. Such features can be processed and analyzed via statistical or emerging soft-computing techniques (e.g. neural networks, fuzzy systems, wavelets, or genetic algorithms) [61]).

Classical visual inspection techniques require extensive image processing and analysis for improving the image quality and deriving characteristic features. The limitation of computer-based tools related to computer time and working space poses a high priority on the objective choice of a limited number of essential characteristics (state-space or feature-space reduction) but also on the exclusion of redundant observations (sample-space or data-space reduction). Thus, the concept of ***approximate processing*** [29], [175] has been considered in real-time applications, where there is a necessity for approximating a given algorithm with another that has reduced computational cost. In any problem-solving domain, an approximation to a given algorithm may be defined as an algorithm that is computationally more efficient and requires reduced computational cost, but produces lower-quality results according to some standards of accuracy, certainty, and/or completeness.

A wide-spread approach related to approximate processing deals with ***feature-space reduction*** and attempts to preserve the most important information conveyed by features extracted from the input data, while simplifying the required computations by reducing the dimensionality of the feature space. Feature-space reduction results in the representation of high-dimensional patterns in a low-dimensional subspace based on transformations that optimize specific criteria in that subspace. Thus, feature-space reduction removes redundant information and allows for more efficient classification. Principal components analysis (PCA) [9], [31], [32] is a well-established feature-space reduction technique employed in different forms including Factor Analysis [31], Karhunen-Loeve Transform (KLT) [7], [9], [11], [32] , and Hotelling Transform [31], [32], depending on the application.

Another approach to information reduction, referred to as ***data-space reduction***, exploits the fact that the underlying dimensionality of the data (intrinsic dimensionality) may be small, even though the input dimensionality is quite large expressing high correlation among input data. Unsupervised linear-mapping approaches in the form of PCA and orthogonal subspace projections are designed to decorrelate the data and maximize the information content in a reduced dimensionality space [176]. Moreover, supervised projection approaches, such as discriminant analysis and Bayesian techniques take advantage of class distributions and are more appropriate for classification purposes, as they can accentuate features of particular interest by maximizing a separation criterion or a Bayesian error criterion, respectively [177].

Motivated by the capabilities of approximate processing and the need for reducing time requirements and overcoming inaccuracies due to "*microscopic*" pixel-level consideration of images, we study in this paper "*macroscopic*" approaches that do not consider pixel processing but rather define in an abstract way the characteristic features of images. More specifically, we consider one approach that only analyzes the sketch of patterns in the image and a second approach that processes only the projection of patterns at a single relevant orientation. In this way we attempt to efficiently balance the amount of relevant information exploited and the computational load of the algorithm. Formally set, we adopt two different forms of data-space reduction directly on the initial image space, affecting: 1) the intensity levels or dynamic range, by transforming the gray-scale image into a binary edge image and 2) the number of independent variables, by utilizing only specific projections of the data. In the former approach referred to as ***reduced dynamic-range processing*** we preserve only the most descriptive information of the image in the form of edges, whereas in the latter approach referred to as ***reduced input-dimension processing*** we preserve the entire content of info but compress it in only a single dimension through projection. In general, the use of edge images takes advantage of the intuitive way that humans operate for recognizing spatial scenes. The characteristics that distinguish images of highly similar low-level properties (pixel distribution) are the shape and topology of objects they contain. Thus, several related methods have been developed for image query and retrieval using ***skeletons***, ***caricatures*** and ***sketches*** (i.e. ***topological features*** of an image) [178 - 180]. Furthermore, the concept of image projection has been used as an effective method for extracting image features (i.e. ***projection features***) in facial image recognition [181] and character recognition [182].

We employ this framework for analyzing SMD images and estimating lead displacements. A novel Bayesian framework for such analysis has been proposed in Chapter 5 based on the fact that positioning measurements on each lead can be viewed as individual (inaccurate) measurements of the same quantity regarding the component displacements. We can view the estimation process operating in two levels. The first level considers a crude computation of quantized displacement of each lead through classification in a limited number of quantized displacements. The second level operates in a Bayesian framework and aims to accurately model the estimation of component displacement based on quantized lead displacements. In this chapter we focus on quantized classification of lead displacements based on reduced dynamic-range and/or input-dimension processing of individual lead images. The motivation behind our work is to avoid and/or overcome problems introduced by segmentation process and reduce the computational complexity of classical machine vision approaches for quality inspection. Compared with feature space reduction that extracts features at pixel level of abstraction and then reduces the feature space, the proposed data-space reduction approach for approximate processing first reduces the data space according to some particular abstract characteristic (either dynamic range or dimension) and then defines features at a higher level of abstraction. In this form, we expect to design algorithms that effectively reduce the complexity of processing by defining features of lower intrinsic dimensionality, while overcoming inaccuracies in image acquisition expressed at pixel-level resolution. Following the process of quantized classification, we can further proceed with the Bayesian estimation approach developed in Chapter 5 to accurately estimate component displacements based on the measurements from many individual leads, i.e. the quantized lead displacements.

6.2 Experimental Set up

The experimental set up for this research is the same as the presented in Subsection 5.1.2. Our approach aims to estimate each lead displacement over its ideal position centered at the pad/paste region. For the purposes of estimating other quality measures, only the displacement along the side of the component is essential [19], [30]. Thus, our problem is restated as estimating lead displacement at the direction perpendicular to the lead axis. Essentially, we consider quantized displacement estimations organized at multiples of a pixel displacements. The displacement classes we consider are again {-6, -3, 0, +3, +6} and {-6, -4, -2, 0, +2, +4, +6}, in pixel displacements over the lead over its central position.

Visual inspection techniques usually proceed in steps as outlined in Figure 4.13 of Chapter 4. The features are extracted from segmented lead images. An example of input lead image and its segmented version is illustrated in Figure 6.1.

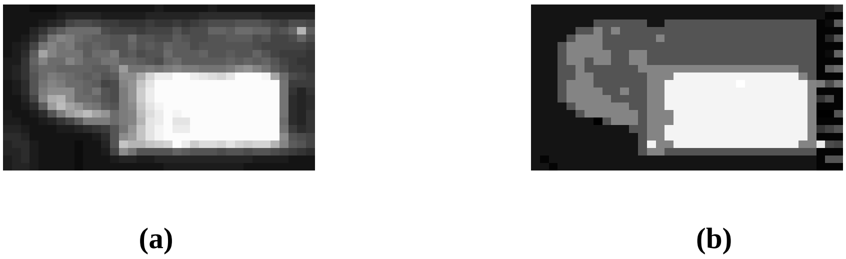


Figure 6.1 Segmentation of lead image based on the 4-level Otsu algorithm. **(a)** Original lead image **(b)** segmented lead image.

Although a high dynamic-range camera is used, the illumination effects are obvious degrading the segmentation result. The usual problems associated with illumination effects have been described in detail in Subsection 4.2.6.2. The proposed reduced dimensionality approach overcomes such problems by establishing a macroscopic consideration of features at a higher level of abstraction. More specifically, the image presented for reduced dynamic-range processing preserves only an abstract sketch of the image edges, whereas the data given for reduced input-dimension processing represents a projection of the input image and reflects the abstract structure of interest on a single direction, as illustrated in Figure 6.2 (c).

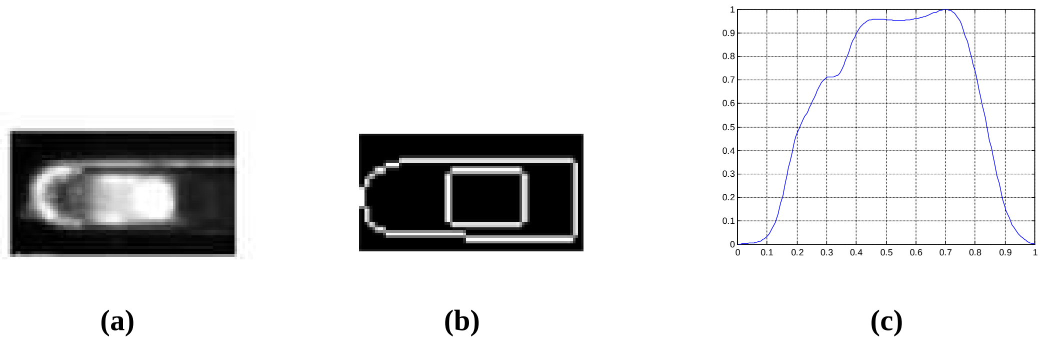


Figure 6.2 Typical images used in data-space reduction approaches.
(a) original image, **(b)** reduced dynamic-range image; **(c)** reduced input-dimension data

6.3 Reduced Dynamic-Range Processing

In our first approach related to data-space reduction, we utilize the edge structure extracted from the input lead image for classification purposes. In most cases, the derived edge structure is partially deformed or destroyed. Thus, the major task is to relate edge patterns so that we can recall a class assignment for each test pattern that may be presented for classification. We exploit the concept of associative memories (AMs) [41], [113], [127-131] as stored patterns representing the desirable classes, and the ***Hamming distance*** [41], [113] for quantifying the distance between the input pattern and each one of the stored memories. The theoretical background of associative memories has been given in Subsection 4.1.2.3. For classification of input patterns we use the ***Hamming neural network*** [41], which is a maximum likelihood classifier used to determine the proximity of an input vector to several exemplar vectors or prototype patterns. An input pattern that partially resembles the stimulus of an association invokes the associated response pattern by means of the shortest Hamming distance. Thus, an associative memory can retrieve a stored pattern given a reasonable subset of the information content of that pattern. Moreover, an associative memory is error correcting in the sense that it can override inconsistent information in the cues presented to it. The input pattern to the network is a binary edge pattern (as in Figure 6.2(b)) obtained from the grayscale input lead image (as in Figure 6.2 (a)) through segmentation and edge detection. The stored AM patterns reflect the edge structure of the "typical" edge image representing each class of lead displacements. Thus, the reduced dimensionality, binary edge image is fed to the Hamming network to determine pattern similarities and implement the desirable classifier. The classifier is trained for 5 and 7 classes, corresponding to integer lead displacements from –6 to +6 pixels per 3 and 2 pixel displacements, respectively. The derivation of the binary edge image, the construction of appropriate fundamental memories and the Hamming classifier are presented in the following sections. The overall classification system is presented in Figure 6.3.

6.3.1 Preprocessing: Derivation of Binary Images

The 8-bit lead image reflects different areas of interest, including the background, exposed pad, solder paste and lead regions. Our study aims to estimate the lead displacement over the pad/paste region. The binary image used in classification reflects the edges of the lead versus the outside pad/paste edges. In order to preserve the simplicity of our approach, we implement a two level segmentation process followed by edge detection to derive the desirable binary image.

The segmentation process aims to separate the lead area from the other region of interest. This process involves the following steps.

- By applying a small threshold, the dark background area is easily removed.
- The application of a second, large threshold derives several bright regions corresponding to the lead and/or the exposed pad areas. Due to the large extent of the lead area and the intense reflection on its surface, the largest of segmented bright regions represents a portion of the lead area. By using a region growing technique to expand the extent of this (largest) region followed by a line fitting process to define its enclosing rectangle, the lead region is effectively segmented and separated from the remaining pad/paste regions.
- A Laplacian edge detector followed by simple thresholding is used to define the edges of the segmented image and derive the binary edge image representing the pattern to be tested through the Hamming neural network.

An example of resulting binary images from the abovementioned procedure is illustrated in Figure 6.2 (b).

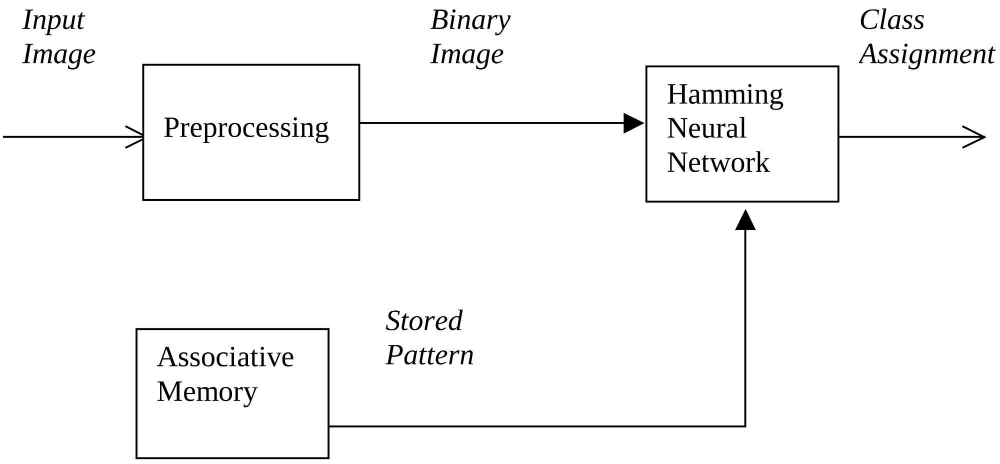


Figure 6.3 Associative memory classification system

Since we exploit the concept of associative memory, the input pattern must have a structure similar to its closest one of fundamental memories. In order to enforce such pattern similarity, we fix the location of the lead in both the test image and the fundamental memories, so that large pattern differences in the comparison of two images can only be attributed to different shifts of the outside boundaries over the fixed lead location. Thus, the pre-processing stage also shifts the centre of gravity of the lead region of each image (test or fundamental) to the same reference position, the center of the image.

.

6.3.2 Fundamental Memory Construction

An important issue of associative memories is the definition of its ***fundamental memories***. Each fundamental memory comprises the specific characteristics discriminating its class. Moreover, the fundamental memories used in lead displacement must assess the standard characteristics of the problem, such as same image size, uniform lead position, etc. To satisfy these requirements, we first select the memory for one displacement (0 pixels) and then construct the memories associated the other classes by shifting the outside edge structure with respect to the fixed structure of the lead. The basic fundamental memory at shift 0 is selected from a number of test images reflecting exactly this specific case through statistical analysis of the mean pattern in this class.

To ensure high performance of the Hamming network, we consider the distance of fundamental memories themselves, by means of the correlation coefficient

$r_{xy} = \sum_i x_i y_i / \sqrt{\sum_i x_i^2 \sum_i y_i^2}$ between two random patterns *x*, *y*. The larger the distance (smaller the cross-correlation) of a class-pattern from its closest neighbors, the better the expected discrimination ability of the classifier. To preserve a consistent correlation form between each fundamental memory and its neighboring patterns, we further process them with the morphological transformation *dilation* using a rhomboid-structuring element with radius 1 pixel. The resulting memories are depicted in Figure 6.4, whereas their correlation table is given in Table 6.1. Observe in Table 6.1 that (as expected) the cross-correlation as a function of class dissimilarity decreases and quickly stabilizes to a constant level.

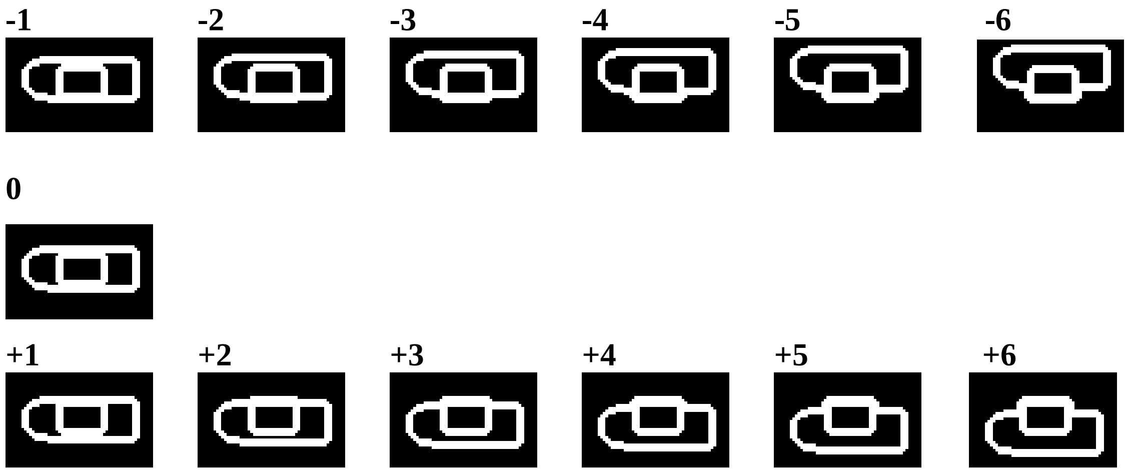


Figure 6.4 Fundamental memories after dilation.

Table 6.1 Correlation coefficients between fundamental memories after dilation.

	-6	-5	-4	-3	-2	-1	0	1	2	3	4	5	6
-6	1.000	0.875	0.750	0.625	0.609	0.591	0.569	0.552	0.510	0.502	0.496	0.490	0.492
-5	0.875	1.000	0.875	0.750	0.623	0.605	0.583	0.565	0.520	0.512	0.502	0.496	0.490
-4	0.750	0.875	1.000	0.875	0.748	0.619	0.597	0.579	0.534	0.522	0.512	0.502	0.496
-3	0.625	0.750	0.875	1.000	0.873	0.744	0.611	0.593	0.548	0.536	0.522	0.512	0.502
-2	0.609	0.623	0.748	0.873	1.000	0.871	0.738	0.605	0.560	0.548	0.534	0.520	0.510
-1	0.591	0.605	0.619	0.744	0.871	1.000	0.867	0.734	0.605	0.593	0.579	0.565	0.552
0	0.569	0.583	0.597	0.611	0.738	0.867	1.000	0.867	0.738	0.611	0.597	0.583	0.569
1	0.552	0.565	0.579	0.593	0.605	0.734	0.867	1.000	0.871	0.744	0.619	0.605	0.591
2	0.510	0.520	0.534	0.548	0.560	0.605	0.738	0.871	1.000	0.873	0.748	0.623	0.609
3	0.502	0.512	0.522	0.536	0.548	0.593	0.611	0.744	0.873	1.000	0.875	0.750	0.625
4	0.496	0.502	0.512	0.522	0.534	0.579	0.597	0.619	0.748	0.875	1.000	0.875	0.750
5	0.490	0.496	0.502	0.512	0.520	0.565	0.583	0.605	0.623	0.750	0.875	1.000	0.875
6	0.492	0.490	0.496	0.502	0.510	0.552	0.569	0.591	0.609	0.625	0.750	0.875	1.000

Thus, the correlation of class 0 with classes {4, 5, 6} is almost the same. This preserved correlation among stored memories degrades the distinction ability between classes as discussed in the examples.

Based now on the design of the fundamental memories, we need to train the network so that it recovers the closest stored pattern in response to each test-input. An example of the desirable operation of the associative memory in the case of a test image with +3 pixels lead-shift is illustrated in Figure 6.5.

(a) **(b)**

Figure 6.5 Associative memory operation. **(a)** testing image **(b)** output response

6.3.3 Classification using the Hamming network

The comparison between the input (edge) pattern and the stored memories requires the use of a distance measure, for example the Euclidean distance or the Hamming distance, for quantizing the output to the fundamental memories representing the desirable classes. To implement this quantization via the Hamming similarity measure, a Hamming network is employed. Its operation aims to select one of the stored patterns (or classes) that is at a minimum Hamming distance (HD) from the binary input vector. The Hamming network consists of two layers. The first layer calculates the (*N*-HD) between the input vector $\mathbf{p}^{probe}$ and the stored $\mathbf{p}^1, \mathbf{p}^2, ..., \mathbf{p}^M$ *N*-dimensional fundamental memories in a feed-forward pass. The strongest response of neurons in this layer is indicative of the minimum HD between the input and the fundamental memories. In our implementation the input in Hamming neural network is a binary image 36×56=2016 pixels. Thus, the input vector of Hamming neural network has dimension 2016, i.e., the first layer of Hamming neural network is constituent of 2016 neurons. The second layer of the Hamming network is a winner-take-all network (MAXNET), implemented as a recurrent network. The MAXNET’s ε parameter was set to ε=0.0385. The MAXNET suppresses all of its input values except the one at the maximum node of the first layer.

Given a set of binary prototype (exemplar) vectors $\mathbf{p}^j, \; j = 1, \ldots, M$, the operation of the Hamming network with N input nodes (number of components of any input vector) and M output nodes (number of prototype vectors) is summarized as follows.

1. For storing the M prototype vectors, compute the weights:

$$w_{ij} = \frac{p_i^j}{2}, \; \left(i = 1, \ldots, N; \; j = 1, \ldots, M \right)$$

and the bias terms:

$$b_j = \frac{N}{2} \; , \; \left(j = 1, \ldots, M \right)$$

2. For each unknown N-dimensional input vector **x**, do steps 3 – 4.

3. compute the net input y_j to each unit Y_j of second layer (MAXNET):

$$y_j\left(0\right) = b_j + \sum_{i=1}^{N} w_{ij} x_i\left(0\right), \; (j = 1, \ldots, M).$$

4. MAXNET iterates to find the best-match exemplar pattern based upon the equation:

$$y_j\left(t\right) = f\left(y_j\left(t-1\right) - \varepsilon \sum_{j \neq i} y_j\left(t-1\right) \right)$$

where f is the activation function : $f\left(x\right) = \begin{cases} x, \; x > 0 \\ 0, \; x \leq 0 \end{cases}$ and ε is a small parameter $0 < \varepsilon < \dfrac{1}{M}$. In our application we set $\varepsilon = \dfrac{1}{2}\left(\dfrac{1}{M}\right) = 0.0385$.

6.4 Reduced Input-Dimension Processing

In this approach we exploit the structure of the lead image profile (projection) along one, the most descriptive direction vertical to the lead axis, for extracting meaningful features related to displacement measurements. The important component of this classification scheme is its feature extraction unit. We propose a complete feature extraction and classification approach that consists of three distinct modules. The first module receives the lead projection function at its input and utilizes a nonlinear filter based on a high-order neural network (HONN) for feature extraction. Feature values encode the five lead displacements (i.e., 0, -3, -6, +3,+6 pixels). The second module implements feature reduction and de-correlation of the feature space by using the Karhunen-Loeve transform (KLT). The third module comprised by the Bayes classifier serves as a classifier that assigns each feature vector to one of the predetermined classes. This pattern classification system is illustrated in Figure 6.6, for 5-classes assignment, and is further discussed in the next sections.

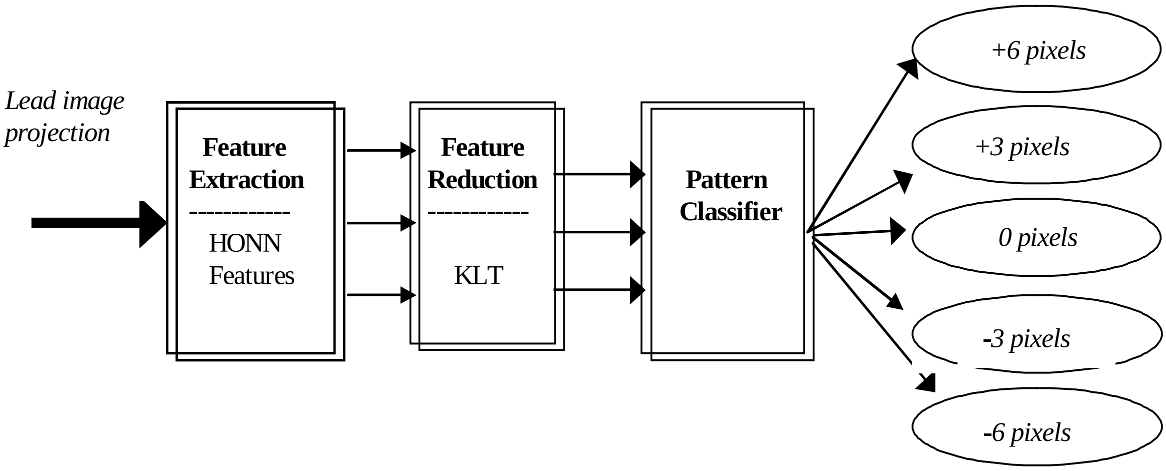


Figure 6.6 Pattern classification for lead image projections

6.4.1 High Order Neural Networks (HONNs)

HONNs are fully interconnected single layer networks, containing high order connections of sigmoid functions in their neurons [132-136]. If we define as x, y its input and output respectively, with $x \in \mathbb{R}^n$ and $y \in \mathbb{R}^m$ the input-output representation of a HONN is given by:

$$y = \mathbf{W}^T \mathbf{S}(x) \qquad (6.1)$$

where $\mathbf{W}$ is a $\mathrm{q} \times m$ matrix of adjustable synaptic weights and $\mathbf{S}(x)$ is a q-dimensional vector with elements $S_i(x),\ i = 1, 2, \ldots, q$ of the form

$$S_i(x) = \prod_{j \in I_i} \left[s(x_j) \right]^{d_j(i)} \qquad (6.2)$$

where $I_i,\ i = 1, 2, \ldots, q$ are collections of q not-ordered subsets of $\{1,2,\ldots,n\}$ and $d_j(i)$ are non-negative integers. In Eqn (6.2) $s(x_j)$ is a monotone increasing, smooth, function, which is usually represented by sigmoids of the form:

$$s\left(x_j\right)=\frac{\mu}{1+e^{-l(x_j-c)}}+\lambda \tag{6.3}$$

for all j=1,2,…,n. In (6.3) the parameters μ, l represent the bound and maximum slope of sigmoid curvature while λ, c are the vertical and horizontal functional shifts, respectively.
For the HONN model described above, it can be seen [132-136], that there exist integers $q,\ d_j\left(i\right)$ and optimal weight values $\hat{\mathbf{W}}$ such that for any smooth but unknown function $f\left(x\right)$ and $\varepsilon \geq 0,\ \left|f(x)-\hat{\mathbf{W}}^T\mathbf{S}\left(x\right)\right| \leq \varepsilon,\ \forall x \in M$ where $M \subset \mathbb{R}^n$ is a compact region. In other words, for sufficient high order terms, there exist optimal weight values $\hat{\mathbf{W}}$ such that the HONN structure $\hat{\mathbf{W}}^T\mathbf{S}\left(x\right)$, can approximate $f\left(x\right)$ to any degree of accuracy, in a compact domain.

Remark 1: Observe that HONNs posses a linear-in-the-weights property. Depending on the form of its regressor terms (6.1) may represent various well-known neural network structures [137].

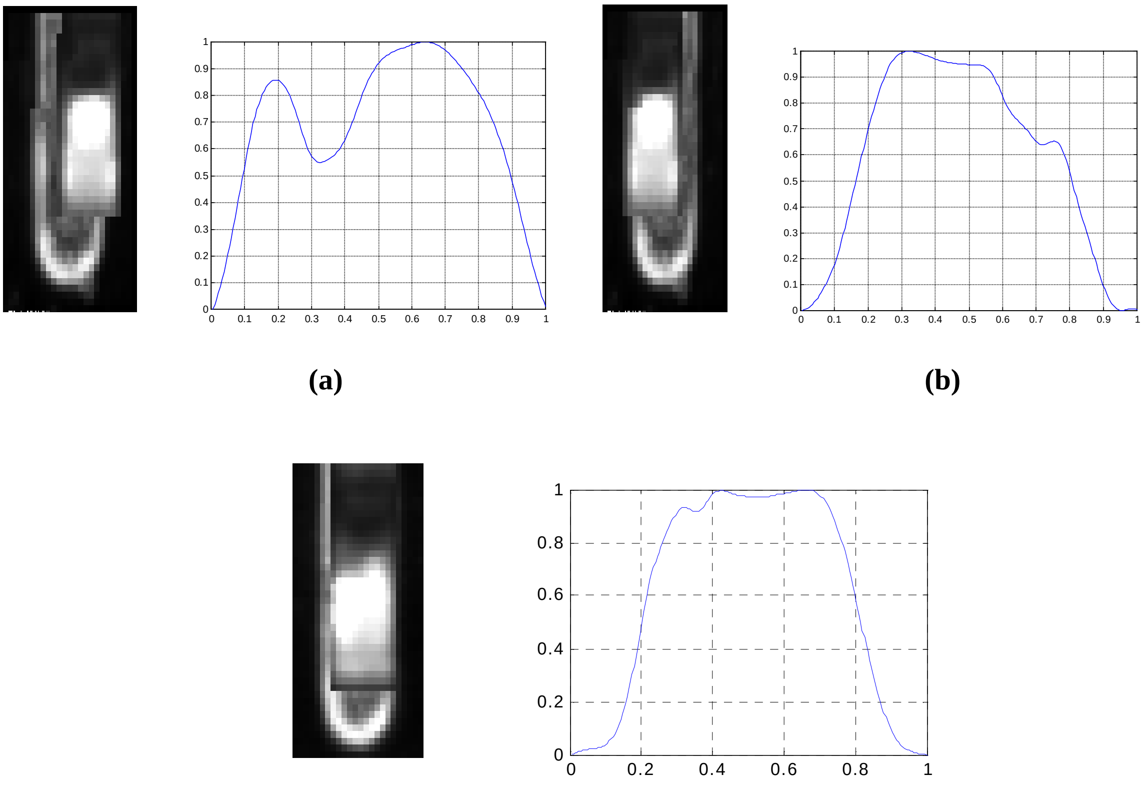
6.4.2 HONN Based Feature Extraction

The HONN based feature extraction module receives as input a normalized projection function of the tested lead image and updates its weights by stable Lyapunov learning laws as to approximate that input function. Prior to entering the input function is linearly transformed in the range [0,1], as to avoid the appearance of destabilizing mechanisms caused by purely numeric issues, (i.e., large variations on the image projections data). Moreover, for uniformity reasons, the rising point of this function is shifted to the origin. Three displacement examples reflecting + 3 pixels, -3 pixels, and 0 pixels lead shift are presented in Figure 6.7. Notice that the unimodal form of the projection function for a centered lead tends to a bimodal structure for increasing lead displacements. This bimodal structure can make the distinction of the direction of displacement quite difficult, since the lobes of the projection function attributed to lead and pad regions can be indistinguishable. Such problems are further discussed along the presentation of results.

In the following we study the construction of the feature extraction system and we rigorously analyze its performance. Let $x \in \mathbb{R}_+$ be the data point on the projection axis,

$y \in \mathbb{R}_+$ be the projection value of lead image ($\mathbb{R}_+$ denotes the set of positive real numbers), and f represent the actual but unknown projection function. Obviously the projection profile is modeled as a function $y = f(x)$. Moreover, let $\hat{y} = \mathbf{W}^T\mathbf{S}(x)$ be a HONN approximation of the actual projection function $f(x)$. Due to the one-dimensional structure of the problem, the HONN is designed for scalar input/output pairs linked at a higher dimension with a weight vector $\mathbf{W}$. Define the projection approximation error as

$$e = f(x) - \mathbf{W}^T\mathbf{S}(x) = y - \hat{y} \tag{6.4}$$



(a) **(b)**

(c)

Figure 6.7. Original lead images and projection functions for (**a**) +3 pixels, (**b**) -3 pixels, and (**c**) 0 pixels lead shift

Observe that e is directly measured even though $f(.)$ is unknown. It has been shown in [137] that the nonlinear adaptive filter

$$\dot{z} = -az + y - \mathbf{W}^T\mathbf{S}(x), \;\; \alpha > 0, \; z \in \mathbb{R} \tag{6.5}$$

equipped with the update law

$$\dot{\mathbf{W}} = -\gamma\mathbf{W} + z\mathbf{S}(x) \;, \;\; \gamma > 0 \tag{6.6}$$

guarantees the uniform ultimate boundedness of its output $z \in \mathbb{R}$ with respect to the arbitrarily small set

$$Z = \left\{ z \in R : |z| \leq \frac{\varepsilon}{2\alpha} + \frac{1}{2}\sqrt{\left(\frac{\varepsilon}{a}\right)^2 + \frac{2\gamma\left|\hat{\mathbf{W}}\right|^2}{\alpha}} \right\} \qquad (6.7)$$

as well as the boundedness of the optimal HONN weights $\hat{\mathbf{W}}$ $\forall x \geq 0$. In the aforementioned relations α, γ are design constants and $\varepsilon \geq 0$ is an unknown but small bound on the HONN reconstruction error.

After convergence of the HONN, the feature vector **F** is formulated by the vector of trained weights $\hat{\mathbf{W}}$ augmented with the approximation error *e*. In our approach, we form the feature vector $(\mathbf{F})$ as

$$\mathbf{F} = \begin{bmatrix} \hat{\mathbf{W}} \\ e \end{bmatrix} = \begin{bmatrix} \hat{w}_1 \\ \hat{w}_2 \\ \vdots \\ \hat{w}_N \\ e \end{bmatrix} \qquad (6.8)$$

where $\hat{\mathbf{W}} = \begin{bmatrix} \hat{w}_1 \\ \hat{w}_2 \\ \vdots \\ \hat{w}_N \end{bmatrix}$ is the optimal HONN weights vector (of specific dimension *N*=12) and *e* is the approximation error. This selection allows **F** to encode all HONN variables that characterize the projection function. An obvious feature is the approximation error *e*. Furthermore, since the HONN possesses a linear-in-the-weights property, the existence of a unique optimal vector $\hat{\mathbf{W}}$ different for each different projection function is guaranteed. Thus, the weights vector $\hat{\mathbf{W}}$ also serves the purpose of a relevant feature.

6.4.3 Parameters Selection of the HONN Feature Extractor using a Genetic Algorithm

A serious drawback in designing of the HONN based feature extraction module is the requirement for manually selecting an optimal set of structural parameters of the non-linear filter. More precisely, the structural parameters of the HONN feature extractor include:

- the HONN order

- the sigmoid function parameters μ, l, λ, c
- the parameter $a > 0$ that appears in the nonlinear adaptive filter (6.5), and
- the parameter $\gamma > 0$ that appears in the weights update law (6.6).

Selecting all the above parameters through a trial and error procedure is very time consuming and requires vast engineering experience. Moreover, the final result (i.e., classifier's success rate), may be far from optimal. Obviously, an automated design process aiming at optimizing the classifier's decision, will serve as a significant addition to the already proposed feature extraction system.

In order to facilitate the design of the HONN feature extractor and automate the process of parameter selection, we introduce the use of a genetic algorithm (GA) embedded into the feature extraction module, as has been proposed in [138]. This approach aims to overcome the problems introduced by the nonlinearity in the parameters property, which implies an infinite set of different parameter combinations that lead to the same feature space topology. The proposed design procedure is illustrated in Figure 6.8.

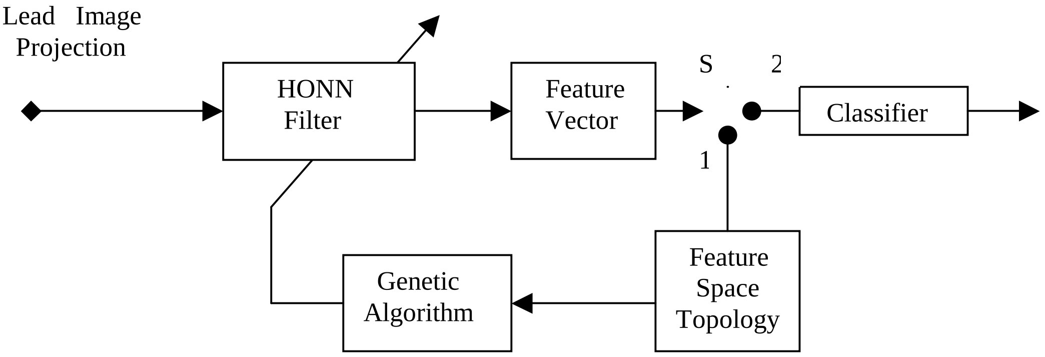


Figure 6.8 The HONN-based feature extraction module parameters selection process.

To enable the feature extraction module parameter selection process, the switch S that appears in Figure 6.8 is set to position 1. After the parameters have been determined, the switch is set to position 2, to proceed to classifier construction.

From the above discussion it becomes apparent that the GA plays a key role in the parameter selection process. In Subsection 4.1.4 a general introduction to genetic allgorithms has been given. In what follows we will describe in more detail the basic ingredients of the used GA in parameters selection of the HONN-based feature extraction process.

In Figure 6.9 we visualize the employed GA in block diagram form. Notice the circular topology of the algorithm. Each cycle is called a generation. First, the number of chromosomes is specified. This number remains constant for all future generations throughout the termination of the genetic process. The chromosomes are real-valued and represent the sigmoid function parameters μ, λ, l, c as well as the gains α, γ that appear in Eqns (6.5) and (6.6) respectively. Each parameter in the chromosome forms a gene. All genes in a chromosome are initialized randomly from values that belong in pre-specified intervals. In this way an initial population is established. Each chromosome corresponds to a specific HONN structure with unspecified weight values. These weights are determined in the HONN training phase that utilizes (6.6).

In the sequel, all chromosomes are evaluated with the aid of a properly selected fitness function. We aim at highly separable classes in the feature space. Two classes $i\,,j$ are called highly separable if the following properties hold:

- P_1. The intra-distance between any two features in a class, is minimum.
- P_2. The inter-distance between any two classes i and j is maximum.
- P_3. The overlap of class i and class j is minimum.

The intra-distance measures the compactness of a class. Hence, the area E_i each class occupies serves as a logical measure. Notice that the requirement of having minimum intra-distance becomes significant as the number of classes grows, which further leads to improving the generalization of the procedure. To measure the inter-distance, we first determine the centers c_i, c_j of the classes i, j respectively, and then calculate the inter-distance as the Euclidean distance $\|c_i - c_j\|$. Satisfaction of P_1, P_2 and P_3 is equivalent of having low variance, minimum overlapping classes, whose centers are distant in the feature space.

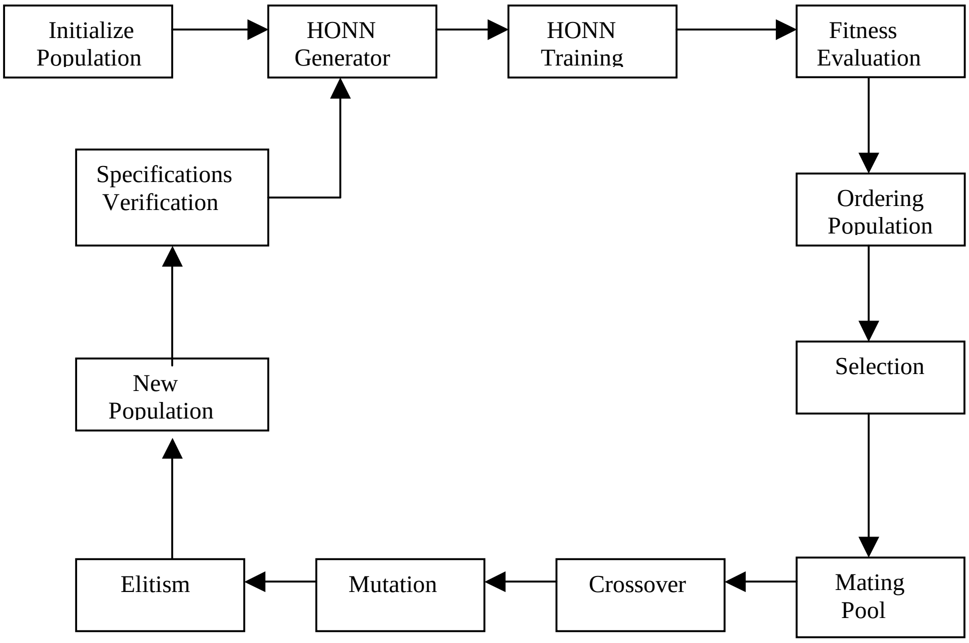


Figure 6.9 The proposed genetic algorithm in block diagram form.

Let v_{ij} denote the overlap of class *i* with class *j*. Minimizing the fitness function:

$$F = \sum_{i,j\ i\neq j} \left[\left(\frac{E_i + E_j}{\|c_i - c_j\|^k} \right)^p + v_{ij} \right] \tag{6.9}$$

where *p*, *k* are some positive constants, obviously leads to the simultaneous satisfaction of the separability conditions P_1 - P_3, thus yielding highly separable classes in the feature space. Prior to calculating (6.9), the border of each class should be determined.

After evaluation, each chromosome is ordered according to its fitness value. A percentage (e.g., the top 35%) are selected to serve as the candidate parents, thus forming the mating pool, while the rest are disregarded.

Since the chromosomes are real-valued, real-valued crossover and mutation operations are applied. First, a ***crossover probability*** $p_c \in [0,1]$ is selected. Consequently, for each gene, a real number $r_c \in [0,1]$ is randomly selected. If $r_c > p_c$ the parents exchange the values of their corresponding *i*-genes. Similar to the crossover, a ***mutation probability*** $p_m \in [0,1]$ is first defined. Consequently, a real number $r_m \in [0,1]$ is randomly selected. If $r_m < p_m$ then the mutation operation is applied to a randomly selected gene of the chromosome. Let *v* be the value of the selected gene. A random perturbation is added on *v* according to the formula:

$$v' = v + \Delta v = v + 0.5rv \qquad (6.10)$$

where $r \in$ [-0.5, 0.5] is also randomly selected. In this way the real-valued mutation operation is performed.

According to the elitism operation, the best fit chromosome (i.e., the one with the highest fitness value) is copied to the next generation. Hence, the search is directed towards the currently best solution. Elitism is the last step of our search. In the sequel, a new population is established and a new generation begins. The algorithm terminates whenever a chromosome is found to possess a fitness value lower than an a priori defined threshold or whenever a pre-specified number of generations has been reached. The algorithm termination conditions are checked in the specifications verification phase, which is depicted in Figure 6.9.

Overall, we select all sructural parameters except HONN order using the GA (guidelines for the selection of HONN order are presented in [137]). The learning parameters of the HONN are determined through (6.6). The operational parameters p_c, p_m , *p* and *k*, as well as the stopping criteria (fitness threshold and maximum number of generations), are defined by the user through a trial and error procedure. In our case we have selected $p_c = 0.6$, $p_m = 0.3$, *p* = 0.5 and *k* = 0.2. The GA terminates if the maximum number of 1500 generations is reached.

6.4.4 Feature Reduction and Classification

In this research, the KLT is used to de-correlate and reduce the dimensionality of feature vectors, disjoint class spaces in the new (reduced) feature space and aid the classifiers in performing accurate discrimination. The KL transformation projects the *N* HONN-weight features to the *K* most important directions. In essence, the KL transform projects feature vectors on the directions that best preserve class properties. Two different forms of the KLT are studied. In the first only one KL transformation matrix (1 KLT) is created for the entire data set, whereas in the second one KLT matrix is created for each class (each displacement). Thus, the first approach computes the most significant directions of the entire problem space and preserves directions where the data set expresses the largest diversion. In the second approach [148], [149] each individual class is represented by its most significant directions. For each vector in class i, only its projection on to the most significant directions of class i

is preserved for classification. For each class, this approach preserves only the directions that best characterize the shape of its boundary and discards the rest. Thus, it encompasses class specific characteristics and uses them to better isolate and discriminate classes by avoiding class mixing in irrelevant directions. The theoretical background of the multiple KLT approach is given in [148], whereas its application as a general analytic tool is established in [149] and is presented in Subsection 4.1.5.2.

Pattern classification may be modelled using a Bayesian approach, assuming a Gaussian distribution of the training set in each class. In our application, a Bayesian-distance classifier is used to implement the classifier. Notice that the classifier operates on the feature vectors extracted by the preceding HONN network and that any available classification scheme may be utilized as a candidate classifier.

6.5 Results

In this section we present and compare classification results obtained by the two proposed approaches. Notice that the classification results obtained from either of the proposed approaches are only quantized measurements of lead displacements. We should emphasize that the process of Bayesian estimation presented in Chapter 5 can be effectively applied as to derive accurate displacement estimates for the entire component from these displacement measurements, which are viewed as individual observations from many cites of the component (its individual leads).

6.5.1 Classification Results using reduced dynamic-range processing

The reduced dynamic-range approach developed in Section 6.3 is now tested on Monte Carlo simulated images from four-sided QFP components. A total of 120 lead samples per class of the lead displacement is obtained resulting in totally 1560 samples for the 13 classes. We consider the Hamming network trained and tested for 7 and 5 classes. The first case involves pixel displacements {-6,-4,-2,0,+2,+4,+6} whereas the second case considers classes {-6, -3, 0, +3, +6}. These two cases study the ability of the classifier to discriminate classes in the feature space separated by 2 and 3 pixels apart, respectively. We do not consider all 13 classes {–6,…, -1, 0, +1,…, +6} or lead displacement per 1 pixel , since all these classes are hardly separable in the 13 dimensional feature space defined. Our approach is tested on 120 lead samples per type of the lead displacement resulting in totally 840 samples for the 7-class testing set and the 600 samples for the 5-class testing set correspondingly. The testing process follows a jack-knifing scheme [9], where all but one-sample feature vectors are used for training and the last one is used for testing. This process is repeated for all samples, leaving one out in every cycle. The overall classification rates from this jack-knifing process approximate the true classification probabilities of the classifier tested.

Using the jack-knife process we obtain the classification probabilities (%) for the 7-class and 5-class feature sets as depicted on Tables 6.2 and 6.3, respectively. Notice that all feature vectors are pre-labeled, so that classification statistics are easily computed for the classifier under consideration. Each row depicts the actual class of the labeled feature vectors, whereas each column indicates the class assigned by the classifier. Thus, each diagonal cell of table indicates the classification success rate for the corresponding class, whereas the rest of the cells in the row illustrate the misclassification rates.

Table 6.2 Classification probabilities (%) of the Hamming-distance classifier for 7 classes

-6 pixels shift	-4 pixels shift	-2 pixels shift	0 pixels shift	2 pixels shift	4 pixels shift	6 pixels shift
85.00	14.17	0.83	0.00	0.00	0.00	0.00
0.00	84.17	15.83	0.00	0.00	0.00	0.00
0.00	1.67	77.50	20.00	0.00	0.00	0.83
0.00	0.00	0.00	92.50	0.83	6.67	0.00
0.00	0.00	0.00	5.00	82.50	12.50	0.00
0.00	0.00	0.00	0.83	8.33	82.50	8.33
0.00	0.00	0.00	1.67	3.33	7.50	88.33

Table 6.3 Classification probabilities (%) of the Hamming-distance classifier for 5 classes

-6 pixels shift	-3 pixels shift	0 pixels shift	3 pixels shift	6 pixels shift
86.67	10.00	3.33	0.00	0.00
0.00	79.17	20.00	0.00	0.83
0.00	0.83	95.00	4.17	0.00
0.83	0.00	5.00	92.50	1.67
0.00	0.00	1.67	5.00	93.33

These classification results illustrate the ability of the Hamming classifier to separate displacement classes. As expected, the results for the 5-classes assignment are more accurate than the 7-classes case, where the larger lead-displacement differences between two successive classes are also reflected in the feature-vector differences. Nevertheless, due to the preserved correlation between AMs of neighboring classes, the discrimination ability is limited even in the 5-classes assignment. For instance, a portion of test leads from class {-6} is diffused to classes {-3} and {0}. Moreover, the percentage of correct classification differs significantly among classes. The latter problem can be alleviated with more extensive testing, but the former is an inherent limitation attributed to dimensionality reduction.

6.5.2 Classification results using reduced input-dimension processing

Similar to Subsection 6.5.1, we consider the Bayesian distance classifier trained and tested for 7 and 5 classes. Again, a total of 840 and 600 sample-leads are used for testing the classifier on seven and five classes, respectively. To approximate the unknown projection function the following HONN structure is used:

$$y = \mathbf{W}^T \mathbf{S}(x) = \sum_{i=1}^{3} w_i s_1^i(x) + w_4 s_2^4(x) + \sum_{i=5}^{8} w_i s_3^{(i-4)}(x) + \sum_{i=9}^{12} w_i s_4^{(i-8)}(x) \tag{6.11}$$

with

$$s_1(x) = \frac{0.9571}{1+e^{-35.703(x-0.076)}} + 0.2245, \qquad s_2(x) = \frac{0.3838}{1+e^{-0.3598(x-1.488)}} - 0.2607$$

$$s_3(x) = \frac{0.9625}{1+e^{-22.4438(x-0.7927)}} + 0.5625, \qquad s_4(x) = \frac{1.2906}{1+e^{-51.468(x-0.3287)}} - 0.3572 \;\; .$$

The HONN weights are updated according to:

$$\dot{w}_i = -0.000534 w_i + z s_1^i(x) \;,\; i = 1,2,3$$

$$\dot{w}_4 = -0.000756 w_4 + z s_2^4(x)$$

$$\dot{w}_i = -0.000825 w_i + z s_3^{(i-4)} \;\;,\; i = 5,6,7,8$$

$$\dot{w}_i = -0.000407 w_i \;,\; z s_4^{(i-8)} \;,\; i = 9,10,11,12 \;\; .$$

The parameter α that appears in (6.7) is fixed to $\alpha = 8.0913$.

The aforementioned in Subsection 6.4.3 genetic algorithm is used to estimate off-line the optimal structural parameters of the non-linear filter except HONN order.

For the displacement classification task, the Bayesian classifier is implemented that computes the distance from an unknown feature vector $\mathbf{x}$ to the *sample mean vector* $\mathbf{m}_i$ of each class and assigns the pattern to the class of minimum distance. Thus, $\mathbf{x}$ is assigned to class ω_i if $D_i < D_j$, for all $j \neq i$. The a priori class probabilities are set to 1/7 and 1/5 for the seven and five-class assignments, respectively. The KL transform is used to decorrelate the feature vectors by taking the projections of the *N*-dimensional HONN features to their *K* most important directions. In this study, the original dimension of N=13 is only reduced to K=11, signifying that the feature extraction process developed yields almost uncorrelated features.

The classification process is repeated for 840 and 600 cycles for the 7 and 5 classes, respectively, by leaving one test sample per cycle out of training. The approach using 1-KLT matrix utilises the 839×13-feature matrix to derive a single 13×11 transform-matrix for the entire training set. The testing vector is projected to the reduced feature space by this KLT matrix and classified according to the minimum distance scheme. In each cycle the jack-knifing process computes the corresponding KLT matrix, trains the classifier with 839 vectors and tests it with the feature vector left out of training. Alternatively, the classification approach using multiple KLT matrices derives a single transform matrix for each class based on the corresponding training set. Overall, within each cycle it derives 7 and 5 different KLT matrices for the seven and five class assignment, respectively. These matrices are then used to train the Bayesian classifier and derive the first and second order stochastic parameters (mean vector and covariance matrix) of each class. Subsequently the testing vector (within each cycle) is projected to all individual KLT matrices so that its distance measure from each particular class can be computed. The testing vector is classified to the class of the shortest distance. In the seven-class problem, for example, the testing vector is multiplied by 7 KLT class-matrices resulting in seven 1×11 vectors $\mathbf{x}_i,\ i=1,...,7$, which are then used as ***class-projected testing vectors*** (one specifically for each class) in the classifier. The initial feature vector is assigned to the class i based on the minimum class-specific distance, i.e. if $D(\mathbf{x}_i,\mathbf{m}_i) < D(\mathbf{x}_j,\mathbf{m}_j)\ \ \forall j=1,...,7\ , j \neq i$.

The probabilities of classification resulting from the jack-knife process are illustrated in Tables 6.4 to 6.7. More specifically, Tables 6.4 and 6.5 present the probabilities of the Bayesian-distance classifier for the 7-class case using 1 and 7 KLT matrices, respectively. The corresponding probabilities for the 5-class case are presented in Tables 6.6 and 6.7, respectively.

Table 6.4 Classification probabilities (%) of the Bayesian classifier on 7 classes (1 KLT matrix)

-6 pixels shift	-4 pixels shift	-2 pixels shift	0 pixels shift	2 pixels shift	4 pixels shift	6 pixels shift
85.83	12.14	2.03	0.00	0.00	0.00	0.00
11.25	67.83	16.13	3.07	1.23	0.00	0.49
0.26	8.72	80.17	10.21	0.64	0.00	0.00
0.00	11.31	0.89	65.64	18.92	3.08	0.16
1.12	0.00	0.00	16.04	73.17	4.60	5.07
0.00	0.00	0.00	3.03	4.26	90.83	1.88
1.54	0.00	0.78	2.01	1.78	1.49	92.4

Table 6.5 Classification probabilities (%) of the Bayesian classifier on 7 classes (7 KLT matrices)

-6 pixels shift	-4 pixels shift	-2 pixels shift	0 pixels shift	2 pixels shift	4 pixels shift	6 pixels shift
76.70	12.64	5.91	2.54	0.00	1.01	1.20
12.47	**80.83**	4.43	1.05	0.00	1.22	0.00
0.00	5.29	**86.07**	8.43	0.21	0.00	0.00
8.55	1.24	3.61	**68.64**	16.13	0.00	1.83
0.27	0.00	0.00	12.24	**83.30**	3.96	0.23
0.00	0.00	0.00	6.02	0.37	**90.00**	3.61
2.01	0.00	0.00	1.08	0.00	3.81	**93.10**

Table 6.6 Classification probabilities (%) of the Bayesian classifier on 5 classes (1 KLT matrix)

-6 pixels shift	-3 pixels shift	0 pixels shift	3 pixels shift	6 pixels shift
88.30	7.67	4.03	0.00	0.00
3.05	91.60	5.35	0.00	0.00
8.65	0.00	82.40	6.41	2.54
0.00	0.00	3.42	90.50	6.08
0.00	0.73	4.82	3.07	91.38

Table 6.7 Classification probabilities (%) of the Bayesian classifier on 5 classes (5 KLT matrices)

-6 pixels shift	-3 pixels shift	0 pixels shift	3 pixels shift	6 pixels shift
92.50	3.69	2.81	0.62	0.38
0.86	93.33	5.81	0.00	0.00
4.27	2.98	86.70	6.05	0.00
0.00	0.00	4.67	93.33	2.00
2.37	0.00	0.00	0.00	97.63

From the above classification results we conclude that the multiple KL approach is able to discriminate classes better that the single KL projection. Moreover, as expected the classification results for the 5-classes assignment are more accurate than these for the 7-classes problem. The limitations due to information reduction are again clear and are attributed to similarities of the projection functions. In this case, the direction of displacement can be difficult. Notice for instance the classification of samples with displacement +6 in Table 6.5, which distributes samples even to the class of –6-pixels displacement. The effects are more evident in the 0-displacement class, whose samples are classified to almost the entire range of values.

6.5.3 Comparison of results

Comparing the results of the two proposed approaches based on approximate processing we can easily derive that none of them has an overall superior performance over the other. The information loss and the associated effects are different in the two approaches. In general, the reduced dynamic-range processing (topological features) is more effective in discriminating 0-displacement features, whereas the reduced input-dimension processing (projection features) is more efficient in classifying lead features reflecting actual displacements. Notice that the complementary information processed by the two algorithms can be efficiently merged within an information fusion scheme, as the proposed in Chapter 8 and [48], to drastically improve the classification probabilities for all classes under consideration. In order to further compare the results of conventional and approximate processing, we copy here the classification probabilities obtained in Chapter 5 for lead features extracted from the original grey-scale images (Table 6.8). It is evident that the results of any of the approximate processing approaches are slightly inferior to the results obtained from full-information images.

Table 6.8 Bayesian Classification Probabilities (%) of lead features extracted at pixel level [from table 5.6.b of Chapter 5]

-6 pixels shift	-3 pixels shift	0 pixels shift	+3 pixels shift	+6 pixels shift
97.75	2.24	0.00	0.00	0.00
2.25	**94.35**	3.10	0.00	0.28
0.00	2.82	**94.35**	1.69	1.12
0.00	0.00	2.57	**93.14**	4.28
0.58	0.00	0.00	0.87	**98.25**

With respect to time requirements, our feature extraction and classification approaches achieve the following performance using a fast Intel Core 2 Duo workstation. The ***pixel-based approach*** (***optical features***), used in Chapter 5, takes about 0.34 sec for processing an entire QFP chip of 120 leads. The ***reduced dynamic-range approach*** (***topological features***) requires 0.15 sec, less than half of the computation time of the conventional approach. Finally, the ***reduced input-dimension processing*** (***projection features***) requires about 0.22 sec for the entire QFP-120 component. Owing to the problem formulation, special attention has been given to the operation of the HONN-based feature extraction module, where a slow convergence of weights might decelerate the entire algorithm. Since the concatenation of leads over their pads is similar for the entire component, the corresponding projection functions to be approximated are all of the same form (for a single component). Thus, the weights of each lead can be initialized at the converged weights of the previously considered lead, highly boosting the performance of the HONN network by avoiding local minima and drastically accelerating its convergence. So, both data-space reduction algorithms achieve significant saving in computational time over the conventional pixel-based approach.

In comparison with existing industrial systems for PCB inspection, the proposed approaches can achieve better throughputs, even though it considers each lead separately. The results of our survey of industrial systems are outlined on Table 6.9. The performance data for commercial products have been obtained through the vendors' online available product datasheets. In our case, the speed of algorithms was mapped to throughput (cm^2/sec) by simulating performance on a 120-lead QFP component of roughly 3.3×3.3cm surface at a sampling resolution of 20 μm/pixel. The speed of each algorithm was estimated with respect to the chip's total lead area. The reported times refer to processing alone, without including the board placement/ adjustment times required by the mechanical operation of the production line.

Table 6.9 Inspection speed comparison

System	speed (cm^2/sec)	resolution (μm/pixel)
optical feature	**32.4**	**20**
reduced dyn range	**72.6**	**20**
reduced imput dim	**49.1**	**20**
Agilent Medalist SJ50 3	38.7	16
Orbotech Symbion P36	22-60	20
Viscom S3088	20-40	15

Overall, we may conclude that high abstraction features used in approximate processing are generally less descriptive than pixel-based features for classification purposes. With respect, however to computational complexity, the approximate processing can yield appreciable reduction at the cost of slightly inferior results.

Chapter 7

Combination of Multiple Classifiers for Post Placement Quality Inspection of Components: A Comparative Study

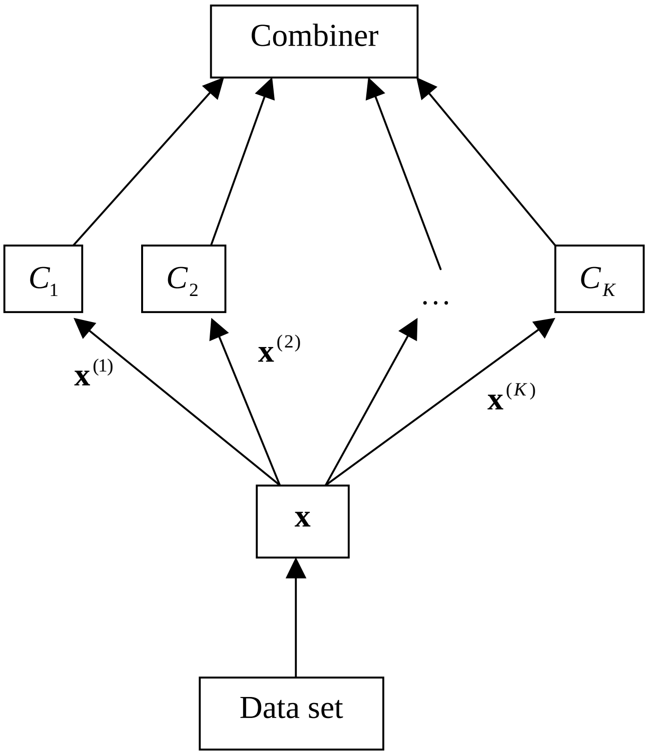
7.1 Introduction

One of the most exciting advances in pattern recognition over the last decade is represented by multiple classifier fusion. It addresses a serious drawback of the classical approach to designing a pattern recognition system and focuses on finding the best classifier by fusing complementary discriminatory information that primary classifiers may encapsulate is not tapped. Multiple expert fusion aims to make use of many different designs to improve the classification performance.

The combination of multiple classifiers has been intensively studied with the aim of overcoming the limitations of primary classifiers [20], [21], [186], [187]. Classifiers differing in feature representation, architecture, learning algorithm, or training data exhibit complementary classification behavior and the fusion of their decisions can yield higher performance than the best individual classifier. The performance of a multiple classifier system relies on both the complementariness of the participating classifiers and the combination method. Hence, the research efforts in this field have focused on either the generation of complementary classifiers or the combination of a given set of classifiers.

A starting point for grouping ensemble classifier methods can be sought in the ways of building the ensemble. The diagram in Figure 7.1 illustrates four approaches aiming at building ensembles of diverse classifiers [21].

Our work presented in this chapter is mainly focused on Approach A and contain details on different ways of combining the classifier decisions. The base classifiers $C_1,\ldots,C_K$ (Approach B), can be any of the models discussed in Chapter 4 along with classifiers not discussed in this Ph.D. thesis. Many ensemble paradigms employ the same classification model, for example, a decision tree and a neural network, but there is not evidence that this strategy is better than using different models [21]. The design of the base classifiers for the ensemble is partly specified within the ***bagging*** and ***boosting*** models [21] while designing the combiner is not coupled with a specific base classifier. At feature level (Approach C) different feature subsets $\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, \ldots, \mathbf{x}^{(K)}$ (i.e., distinct pattern representations) can be used for the primary classifiers. This topic is included in the work presented in Chapter 8. Finally, the data sets can be modified so that each classifier in the ensemble is trained on its own data set (Approach D). This approach has proved to be extremely successful owing to the bagging and boosting methods [21].



A. ***Combination level*****:**
Design different
combiners.

B. ***Classifier level:***
Use different
base classifiers.

C. ***Feature level:***
Use different
feature subsets.

D. ***Data level:***
Use different
data subsets.

Figure 7.1 Approaches to building classifier ensembles

There are generally two types of combination: ***classifier selection*** and ***classifier fusion*** [21], [188]. The presumption in classifier selection is that each classifier is "an expert" in some local area of the feature space. Classifier fusion assumes that all classifiers are trained over the entire feature space and are, thereby, considered as competitive rather than complementary. On the other hand, based on a given classifier set, the combination methods can be categorized according to the level of the individual classifiers outputs: ***abstract level*** (*class label*), ***rank level*** (*rank order*), and ***measurement level*** (*class scores*) [186]. The abstract level classifiers output only the class label, whereas the rank level classifiers output the rank for each class. The measurement level classifiers assign each class a measurement value to indicate the possibility that the input pattern pertains to the class. In principle, the class scores rovide richer information than the class label and the rank order and should give higher combination performance.

An important issue in combining multiple classifiers is the use of different feature sets or different training sets, randomly selected [187]. In addition, from the point of view of the ***input pattern representation***, there are basically two classifier combination scenarios [187]. In the first scenario, all the classifiers use the same represantation of the input pattern (***identical pattern representation***). In the second scenario, each classifier uses its own representation of the input pattern (***distinct pattern representation***) [187], [189]. In this case, the measurements extracted from the pattern are unique to each classifier, i.e, each individual classifier uses a different set of features.

A variety of schemes have been proposed for combining multiple classifiers. The most often used classifiers fusion approaches include the majority voting [186], [190], [191]; various rank-ordered rules, such as the sum rule (averaging), product-rule, max-rule, min-rule, median rule [20], [187]; the weighted combination (weighted averaging) [21], [192]; the Borda count [193], [195]; the Bayesian approach (naïve Bayes combination) [186], [194-195]; the Dempster–Shafer (D-S) theory of evidence [186], [196], [198-200]; the behavior–knowledge space method (BKS) [201-202]; the fuzzy integral [15], [169], [192], [209]; fuzzy templates [203]; decision templates [204]; the probabilistic schemes [20], [187], [189]; combination through order statistics [205], [206]; combination by a neural network [169], [207]. Overall a comparative table of various classifier combination strategies based on a few properties can be found in [208].

The objective of the presented work in this chapter is to test and compare multiple classifier fusion methods for improving the classification of the individual leads in component quality inspection. **This work has been accepted for publication in [48]**. Instead of using single statistical or neural classifiers as in the previous Chapters 5 and 6 we implement multi-modular classification systems that combine decisions from statistical and neural modules. Combining the power of the individual classifiers through multimodular architectures we improve the classification results and enhance the robustness of the overall classification system. We propose four representative schemes for soft fusion of multiple classifiers. The first approach uses the majority voting principle for fusion of multiple experts. The second scheme performs combination by using the naïve Bayes method. In the third approach, the outputs of multiple classifiers are combined using Dempster – Shafer theory of evidence. The last scheme involves the calculation of fuzzy integrals. All fusion methods, using identical pattern representations are considered. The features for classification are obtained directly from the lead images as has been presented in Subsection 4.2.6.4 (we use ***optical features*** as identical pattern representations). Following the process of quantized classification of individual leads, we can further proceed with the Bayesian estimation approach developed in Chapter 5 to accurately estimate component displacements based on the measurements from many individual leads, i.e. the quantized lead displacements.

7.2 Experimental Set up and Feature Extraction Process

In this research we use the experimental procedure presented in Subsection 5.1.2 of Chapter 5. So, for the purpose of presenting our results, QFP (Quad Flat Pack) SMD components with 120 leads (30 leads per side) are employed. The feature extraction process from lead images is identical to developed in Subsection 4.2.6.4. As in Chapters 5 and 6, we consider again quantized displacement estimations organized at multiples of a pixel displacement. The displacement classes considered are {-6, -3, 0, +3, +6} and {-6, -4, -2, 0, +2, +4, +6}, in pixel displacements over the lead's central position.

7.3 Multiple classifier combination methods

7.3.1 Formulation of the combined classifier problem

In this research, we assume that a small set of trained classifiers is available and we are interested in combining their outputs aiming at the highest possible accuracy.

Let $C = \{C_1, C_2, \ldots, C_K\}$ be a set of classifiers and $\Omega = \{\omega_1, \omega_2, \ldots, \omega_M\}$ be a set of class labels. Each classifier gets as input a feature vector $\mathbf{x} \in \mathbb{R}^n$. The classifier output is an *M*- dimensional vector $C_i(\mathbf{x}) = [c_{i,1}(\mathbf{x}),\ldots,c_{i,M}(\mathbf{x})]^T$, where $c_{i,j}(\mathbf{x})$ is the degree of “support” given by classifier $C_i,\ i = 1,\ldots,K$ to the hypothesis that $\mathbf{x}$ comes from class $\omega_j,\ j = 1,\ldots,M$. Without loss of generality we can restrict $c_{i,j}(\mathbf{x})$ within the interval $[0,\ 1]$ and call them “soft labels”, with 0 meaning “no support” and 1 implying “full support”, $i = 1,\ldots K,\ j = 1,\ldots M$ [202]. Most often $c_{i,j}(\mathbf{x})$ is an estimate of the posterior probability $P(\omega_i \mid \mathbf{x})$. The process of combining classifiers attempts to combine the *K* classifier outputs $C_1(\mathbf{x}),\ldots,C_K(\mathbf{x})$ as to obtain a soft label for $\mathbf{x}$, denoted $C(\mathbf{x}) = [\mu_1(\mathbf{x}),\ldots,\mu_M(\mathbf{x})]^T$, where $\mu_j(\mathbf{x})$ denotes the overall degree of support for ω_j given by the ensemble classifier.

If a crisp class label of $\mathbf{x}$ is needed, we can use the maximum membership rule, which assigns $\mathbf{x}$ to class ω_s iff,

$$c_{i,s}(\mathbf{x}) \ge c_{i,j}(\mathbf{x})\ \ \forall\, j = 1,\ldots,M \quad \text{for individual crisp labels and} \qquad (7.1)$$

$$\mu_s(\mathbf{x}) \ge \mu_l(\mathbf{x}),\ \forall l = 1,\ldots,M \quad \text{for the final crisp label} \qquad (7.2)$$

The minimum-error classifier is recovered from Eqn (7.2) when $\mu_i(\mathbf{x}) = P(\omega_i \mid \mathbf{x})$. In the following we introduce the four combination methods already mentioned in Section 7.1.

7.3.2 Majority Voting

Majority voting is a popular and easy to implement method [21], [184-186], [191]. The primary classifiers “vote” with their class labels and the class label with most votes is assigned to $\mathbf{x}$. Let $C_i(\mathbf{x}) = [c_{i,1}(\mathbf{x}),\ldots c_{i,M}(\mathbf{x})]^T \in [0,\ 1]^M$ be the output of classifier C_i for input $\mathbf{x}$. To assign a class vote to classifier C_i, we harden the classification decision by the *maximum membership* formula

$$\text{choose class } \omega_{\mathrm{k}} \iff C_{i,k}(\mathbf{x}) = \max_j \left\{ c_{i,j}(\mathbf{x}) \right\} \qquad (7.3)$$

By this rule, we can formulate the hardened classification decision of each C_i as the binary vector C_i^h (h stands for “hardened”) containing 1 at position k and 0 elsewhere, i.e.,

$$c_{i,j}^h(\mathbf{x}) = \begin{cases} 1, & \text{if } j = k \\ 0, & \text{otherwise} \end{cases} \tag{7.4}$$

The majority vote aggregation F_{maj} is given by

$$F_{\text{maj}} \equiv C(\mathbf{x}) = \left[c_1(\mathbf{x}), \ldots, c_M(\mathbf{x})\right]^T, \;\; c_j(\mathbf{x}) \;\in\; \{0,1\} \tag{7.5}$$

and

$$c_j(\mathbf{x}) = \begin{cases} 1, \;\text{if } \displaystyle\sum_{i=1}^{K} c_{i,j}^h(\mathbf{x}) = \max_j \sum_{i=1}^{K} c_{i,j}^h(\mathbf{x}) \\ 0, \;\text{otherwise} \end{cases} \tag{7.6}$$

The result is a binary vector with element 1 corresponding to the most supported class, and 0 elsewhere. In an equivalent formulation, the class label is assigned if the majority of K classifiers, i.e., at least ⌊K/2⌋ + 1 classifiers, vote for that class. More than one element in Eqn (7.6) with value 1 means a tie. To find a single class label for $\mathbf{x}$, ties are settled randomly.

7.3.3 Naïve Bayes Combination

Whereas the voting method only considers the result of each classifier, the approach of Bayesian formalism [21], [195] considers the error of each classifier. The “naïve Bayes” scheme assumes that the classifiers are mutually independent given a class label (conditional independence).

Consider the crisp class labels obtained from the K classifiers and let $L_1, \ldots, L_K$ be the class labels assigned to $\mathbf{x}$ by classifiers $C_1(\mathbf{x}), \ldots, C_K(\mathbf{x})$, respectively. Thus, for any input $\mathbf{x} \in \mathbb{R}^n$ to be classified, the K classifier outputs define a vector $\mathbf{L} = [L_1, \ldots, L_K] \;\in\; \Omega^K$. Denote by $P(L_j)$ the probability that classifier C_j labels $\mathbf{x}$

in class $L_j \in \Omega$. The conditional independence allows for the following representation

$$P(\mathbf{L} \mid \omega_k) = P(L_1, \ldots, L_K \mid \omega_k) = \prod_{i=1}^{K} P(L_i \mid \omega_k) \tag{7.7}$$

Then, the posterior probability needed to label $\mathbf{x}$ is given by

$$\begin{aligned} P(\omega_k \mid \mathbf{L}) &= \frac{P(\omega_k) P(\mathbf{L} \mid \omega_k)}{P(\mathbf{L})} = \\ &= \frac{P(\omega_k) \prod_{i=1}^{K} P(L_i \mid \omega_k)}{P(\mathbf{L})}, \qquad k = 1, \ldots, M \end{aligned} \tag{7.8}$$

Since the denominator does not depend on ω_k and can be ignored, the support for class ω_k by the set of classifiers can be computed as

$$\mu_k(\mathbf{x}) \propto P(\omega_k) \prod_{i=1}^{K} P(L_i \mid \omega_k) \tag{7.9}$$

The practical implementation of the naïve Bayes fusion on a data set S with cardinality *N* is explained below. Assuming *M* classes labeled 1 through *M*, the error for the *ith* classifier, $i = 1, \ldots, K$, can be represented by a two-dimensional ***confusion matrix*** as follows:

$$CM^i = \begin{pmatrix} a_{1,1}^i & \cdots & a_{1,M}^i \\ \vdots & \ddots & \vdots \\ a_{M,1}^i & \cdots & a_{M,M}^i \end{pmatrix} \tag{7.10}$$

For each classifier C_i, an $M \times M$ confusion matrix CM^i is calculated by applying C_i to the training data set *S*. The (*k*,*L*) th entry of this matrix, $a_{k,L}^i$ is the number of elements of the data set whose true class label was ω_k and were assigned to class ω_L by C_i. By N_L we denote the total number of elements of *S* that truly belong to class ω_L. Taking $a_{k,L_i}^i / N_k$ as an estimate of the probability $P(L_i \mid \omega_k)$, and N_k / N as an estimate of the prior probability of class ω_k, Eqn (7.9) is equivalently written as:

$$\mu_k(\mathbf{x}) \propto \frac{1}{N_k^{K-1}} \prod_{i=1}^{K} a_{k,L_i}^{i} \tag{7.11}$$

7.3.4 Multi-Classifier Combination based on Fuzzy Integral

Fuzzy integral has been reported to give excellent results as a classifier combiner [15],[21]. The philosophy of the fuzzy integral combiner is to measure the “strength” not only for each classifier alone but also for all the subsets of classifiers. Every subset of classifiers has a measure of strength that expresses how good this group of experts is for the given input $\mathbf{x}$. The ensemble support for class ω_j, $\mu_j(\mathbf{x})$, is obtained from the support values of individual classifiers $c_{i,j}(\mathbf{x}), i = 1,\ldots,K$, $j = 1,\ldots,M$, but also taking into account the competences of the groups of the various subsets of experts. The measure of strength of the subsets is expressed through a ***fuzzy measure*** [15], [39-41] denoted by *g*. Thus, to get $\mu_j(\mathbf{x})$ we “fuse” the support values $c_{i,j}(\mathbf{x}), i = 1,\ldots,K$, $j = 1,\ldots,M$ and g via fuzzy integral [192], [209].

7.3.4.1 Mathematical Background on fuzzy measures and fuzzy integrals

Stemming from the concept of fuzzy sets [41], the theory of fuzzy measures and fuzzy integrals was first introduced by Sugeno [15], [192], [209]. In fuzzy sets, a value $\mu_A(x)$ is assigned to each element *x* of the universal set *X* signifying its degree of membership to a particular set *A* with non-sharp (fuzzy) boundaries. A fuzzy measure is used to express an evaluation of a concept that is heavily subject to human perception. In mathematical terms, a fuzzy measure is a set function with monotonicity (often, but not always being additivity). Thus, a fuzzy measure assigns a value in the unit interval [0,1] to each crisp subset *A* of the universal set *X* signifiying the degree of evidence or belief that a particular element *x* belongs to this crisp subset. Consider, for instance, a group of people *X*. For fuzzy sets, the age of a person $x \in X$ is *known*, so that we consider x to be “Old” with a membership grade

$\mu_A(x)$, where *A* is a fuzzy set of “Old” people. For fuzzy mesures, on the other hand, the age of the person x $\in$ *X* is unknown but there is an indication (possibly by looking at that person) that (s)he belongs to the crisp subset A consisting of people with age 50 years old, with a measure of $g_x(A)$. Thus, in fuzzy sets a value is assigned to each element of the universal set signifying its degree of membership to a particular set with an unsharp boundary, while in fuzzy measures a value is assigned to crisp subset of the universal set signifying the degree of evidence or belief that a particular element belongs in the subset. In this form, fuzzy sets are used to address *vagueness* associated with the difficulty of making sharp or precise distinctions of objects in the world, whereas fuzzy measures are used to solve *ambiguity* associated with making a choice between two or more alternatives.

Based on the notion of a fuzzy measure, a fuzzy integral is a function with monotonicity which is used for aggregating information from multiple sources with respect to a fuzzy measure.

Sugeno fuzzy measure

Let *X* be the universe of discourse and 2^X be the power set of *X* (i.e all crisp subsets A in X) . Then a set function $g:\ 2^X \rightarrow\ [0,1]$, which assigns a number in the unit interval [0, 1] to each crisp subset of *X*, is defined as a fuzzy measure if it satisfies the following three axioms:

- **Axiom 1** (Boundary conditions): $g(\varnothing)=0,\ g(X)=1.$
- **Axiom 2** (Monotonicity): For every pair of crisp sets

  $A, B \in 2^X$, if $A \subseteq B$, then $g(A) \leq g(B)$.
- **Axiom 3** (Continuity): For every sequence $\left(A_i \in 2^X \mid i \in \mathbb{Z}^+\right)$ of measurable subsets of X , if either $A_1 \subseteq A_2 \subseteq \cdots$ or $A_1 \supseteq A_2 \supseteq \cdots$ (i.e. , the sequence is monotonic), then $\lim\limits_{i\to\infty} g(A_i) = g\left(\lim\limits_{i\to\infty} A_i\right)$, where $\mathbb{Z}^+$ is the set of all positive integers.

For a crisp subset $A \in 2^X$, $g(A)$ represents the degree of evidence, or our belief, that a given element $x \in X$ (which has not been previously located in any crisp subset of X) belongs to the crisp subset *A*. In general, the fuzzy measure of the union of two disjoint subsets cannot be directly computed from the fuzzy measures of the subsets. For this purpose, Sugeno [41], [209] introduced the so called, λ-fuzzy measure. The motivation for defining this measure is that the specification of a fuzzy measure *g* requires the knowledge of $g(A)$ for all subsets A in X. In order to reduce the quantity of primary data, an extra axiom can be added to Axioms 1-3 of fuzzy measures, which allows the calculation of $g(A)$ from$\left\{g(\{x\}) \mid x \in A\right\}$. The λ-fuzzy measure allows the computation of the fuzzy measure of the union of two disjoint subsets directly from the fuzzy measures of the subsets. Sugeno proposed the decomposable λ-fuzzy measure, satisfying the following additional Axiom 4 known as the λ-rule:

- **Axiom 4**: $g(A \cup B) = g(A) + g(B) + \lambda g(A) g(B),$
  $A, B \subset X$ and $A \cap B = \varnothing,$ with λ>-1

When $\lambda = 0$, the λ- fuzzy measure becomes a probability measure [209]. In general, the value of λ can be determined from the properties of the λ-fuzzy measure.

Let $X = \{x_1, x_2, \ldots, x_K\}$ be a finite set (a set of committee members in our case). If the ***fuzzy density*** of the λ-fuzzy measure is defined as a function $g_i : x_i \in X \rightarrow [0,1]$ such that $g_i = g(\{x_i\}),\ i = 1, \ldots, K$, then the λ-fuzzy measure of X can be obtained in a closed form as [209] :

$$g(X) = \sum_{i=1}^{K} g_i + \lambda \sum_{i_1=1}^{K-1} \sum_{i_2=i_1+1}^{K} g_{i_1} g_{i_2} + \ldots + \lambda^{K-1} g_1 g_2 \cdots g_K \tag{7.12}$$

If $\lambda \neq 0$, (7.12) can be rewritten as

$$g(X) = \frac{1}{\lambda}\left[\prod_{i=1}^{K}(1 + \lambda g_i) - 1\right] \tag{7.13}$$

If $g(X) = 1$, the constant λ can be determined by solving the folowing equation:

$$\lambda + 1 = \prod_{i=1}^{K}(1 + \lambda g_i) \tag{7.14}$$

It has been prooved [209] that for a fixed set of fuzzy densities g_i' s, $i = 1, ..., K,\ 0 < g_i < 1,$ there exists a unique root of $\lambda > -1$ and $\lambda \neq 0$ of Eqn (7.14). Also from Eqn (7.14) it can be seen that if the values of g_i are known, then λ can be readily computed. A possible interpretation of a fuzzy density, g_i, can be given by the grade of importance, or the degree of belief in the single attribute x_i, for the overall evaluation of the system.

When g is the λ-fuzzy measure, the values of $g(A_i)$, $A_i \subset X$, and $A_i = \{x_i, ..., x_K\}$, can be computed recursively as follows:

$$g(A_1) = g(\{x_1\}) = g_1 \ , \tag{7.15}$$

$$g(A_i) = g_i + g(A_{i-1}) + \lambda g_i g(A_{i-1}), \text{ for } 2 \le i \le K \tag{7.16}$$

Choquet integral

Let g be a fuzzy measure on *X*. The ***discrete Choquet integral*** *C_g(.)* of a function $f : X \to \mathbb{R}^+$ with respect to g is defined as:

$$C_g\left[f(x_1),\ldots,f(x_K)\right] = \sum_{i=1}^{K}\left[f(x_i) - f(x_{i-1})\right]g(A_i), \tag{7.17}$$

where indices i have been permuted so that

$0 \le f(x_1) \le \ldots \le f(x_K) \le 1$, $A_i = \{x_i, \ldots, x_K\}$ and $f(x_0) = 0$.

There are numerus interpretations of the meaning of fuzzy integrals. A fuzzy integral can be understood as a fuzzy expectation [209], the maximal grade of aggrement between two opposite tendencies [209], or the maximal grade of aggreement between the objective evidence and the expectation [209]. In this work, a fuzzy integral is considered as a maximum degree of belief (for a class or an object) obtained from the fusion of several objective evidences, where the respective importance of multiple attributes is subject to fuzzy measures.

7.3.4.2 Multi-classifier Fusion by Choquet fuzzy integral

We adopt Sugeno's λ-fuzzy measure and assign the initial fuzzy densities g_i , $i = 1,\ldots,K$, as the degrees of importance of classifiers $C_i, i = 1,\ldots,K$, based on their performance on the testing data. For each tested feature vector **x**, let $c_{1,j},\ldots,c_{K,j}$ be the support from classifiers $C_1,\ldots,C_K$ for class ω_j, respectively, obtained from the confusion matrices CM^i, $i = 1,\ldots,K$ of classifiers C_i , $i = 1,\ldots,K$. For every class ω_j, the overall degree of support is computed as follows. We first assign the initial fuzzy densities g_i , $i = 1,\ldots,K$:

$$g_i = c_{i,j},\ i = 1,\ldots,K \tag{7.18}$$

The value of λ needed for the calculation and integration of the fuzzy measure *g(.)* is obtained as the unique real root greater than –1 of Eq. (7.14). Once $g_1,\ldots,g_K$ are set and λ is found, the computation of the fuzzy integral for classifier fusion for class ω_j proceeds with the following algorithm:

a. For a given $\mathbf{x}$, sort the support values $c_{1,j}(\mathbf{x}), c_{2,j}(\mathbf{x}),\ldots,c_{K,j}(\mathbf{x})$, for class ω_j, to obtain $c_{i_1,j}(\mathbf{x}), c_{i_2,j}(\mathbf{x}),\ldots,c_{i_K,j}(\mathbf{x})$, with $c_{i_1,j}(\mathbf{x})$ being the highest degree of support, and $c_{i_K,j}(\mathbf{x})$ the lowest one.

b. Arrange the fuzzy densities correspondingly, i.e., $g_{i_1},\ldots,g_{i_K}$ and set
$$g(1) = g_{i_1}$$

c. For k classifier combinations, $k = 2$ to K, calculate recursively the λ-fuzzy measures: .
$$g(k) = g_{i_k} + g(k-1) + \lambda g_{i_k} g(k-1)$$

d. Calculate the overall degree of support for class ω_j by the Choquet fuzzy integral:
$$\mu_j(\mathbf{x}) = c_{i_1,j}(\mathbf{x}) + \sum_{k=2}^{K} \left[c_{i_{k-1},j}(\mathbf{x}) - c_{i_k,j}(\mathbf{x})\right] g(k-1)$$

The above algorithm is repeated for all classes $\omega_j, j = 1,\ldots,M$. The final output class for the combined classifier is selected as the one with the highest integrated value, i.e. as in Eqn (7.2).

The support for ω_j, $\mu_j(\mathbf{x})$, can be thought of as a compromise between the competence (represented by the fuzzy measure *g*) and the evidence (repesented by the support values $c_{1,j}(\mathbf{x}), c_{2,j}(\mathbf{x}),\ldots,c_{K,j}(\mathbf{x})$). Notice that the fuzzy measure vector $[g(1),\ldots,g(K)]^T$ might be different for each class and is specific for the current vector **x**.

7.3.5 Multiclassifier combination based on Dempster – Shafer theory of evidence

The Dempster – Shafer theory of evidence [41], [185-186], [196-197] also known as the theory of belief functions, is a generalization of the Bayesian theory for subjective probability. This theory is more flexible than Bayesian when our knowledge is incomplete and we have to deal with uncertainty and ignorance. Whereas the Bayesian theory requires the assignment of probabilities for each question of interest, belief functions allow us to assign degrees of belief for one question based on probabilities for a related question. These degrees of belief may or may not have the mathematical properties of probabilities; how much they differ from probabilities will depend on how closely the two questions are related.

7.3.5.1 Mathematical Background on Dempster – Shafer theory of evidence

In this section we introduce the basic concepts of the ***Dempster – Shafer (D-S) theory of evidence*** [41], [186], [196],

Let Θ be a set of mutually excaustive and exlusive atomic hypotheses, $\Theta = \{\theta_1, \ldots, \theta_M\}$, referred to as the ***frame of discernment***. A subset $A = \{\theta_{i_1}, \ldots, \theta_{i_q}\} \subset \Theta$ represents a hypothesis denoting the disjunction $\theta_{i_1} \cup \ldots \cup \theta_{i_q}$. Each element $\theta_i \subset \Theta$ corresponds to a one-element subset $\{\theta_i\}$, called a singleton. Let 2^{Θ} denote the power set of Θ, i.e., the set of all possible subsets of Θ, so that each subset $A \subset \Theta \in 2^{\Theta}$.

The D-S theory uses a numeric value in the interval [0, 1] inclusive to indicate belief in a hypothesis (subset) $A \subset \Theta$ based on the occurrence of an evidence *e*. This value, conventionally denoted by *Bel*(*A*), indicates the degree to which the evidence *e* supports the hypothesis *A*. The value of Bel(*A*) is calculated from another function called ***basic probability assignment*** (bpa), which represents the individual impact of each evidence on the subsets of Θ. A bpa, denoted *m(.)*, is a generalization of a probability density function. It assigns values in [0, 1] to each and every element of 2^{Θ} (i.e., each subset of Θ, instead of each element of Θ as in probability theory) such that the numeric values sum up to 1.

A function *m* is called a basic probability assignment if:

$$m:2^{\Theta}\rightarrow\left[0,1\right],\ m\left(\varnothing\right)=0,\ \text{and}\ \sum_{A\subseteq\Theta}m\left(A\right)=1 \tag{7.19}$$

The quantity $m\left(A\right)\in\left[0,1\right],\ A\in 2^{\Theta}$, is called *A*'s basic probability and is interpreted as the degree of evidence in support of some element of Θ belonging to the set *A,* but the evidence does not extend to any particular subset of *A*. Whereas probability theory assigns a measure of probability to atomic hypotheses θ_i, *m(A)* represents the ***belief*** of a (no necessarily atomic) hypothesis *A*. Instead of probability, *m(A)* is a measure of support we are willing to assign to a composite hypothesis *A* at the expense of support $m\left(\theta_i\right)$ of atomic hypotheses θ_i. If for the frame of discernment Θ we set $m\left(\theta_i\right)\neq 0$ for all θ_i and *m(A)*=0 for all $A\neq\theta_i$, the formulation resembles that of probability theory with $\sum_{i=1}^{M}m\left(\theta_i\right)=1$ and $m\left(\theta_i\right)$ may be regarded as a probability of θ_i.

Every set $A\in 2^{\Theta}$ for which *m*(*A*) > 0 is called a ***focal element*** of *m*. When Θ is finite, *m* can be fully characterized by a list of its focal elements *A* with the corresponding values *m*(*A*). It is interesting to point out that basic probability assignments are not fuzzy measures and are characterized by the following particular features:

- *m*(*A*) is the portion of the total belief commited exactly to *A*, which cannot be further subdivided among the subsets of *A* and does not include the portions of the total belief commited to subsets of *A*.
- The singletons $\{\theta_i\},\ i=1,...,M$ are only parts of sets in 2^{Θ}. Thus, it is possible that $\sum_{i=1}^{M}m\left(\theta_i\right)<1$. Moreover, since θ_i and $\neg\theta_i$ (*i.e.* $\Theta-\theta_i$) are only two elements of 2^{Θ}, it is possible that $m\left(\theta_i\right)+m\left(\neg\theta_i\right)<1$. This feature of singletons defies the basic axioms of Bayesian formalism and, in other words, the bpa supplies an incomplete probabilistic model.
- When *A* is the only focal element in 2^{Θ}, we have *m*(Θ) = 1-*m*(*A*). In general, *m*(Θ) absorbs the unassigned portions of the total belief after commitment of belief to various proper subsets of Θ.

Since a subset *A* represents the disjunction of all the elements in *A*, the truth of $B \subseteq A$ implies truth of *A*, i.e., all the evidence committed to exactly one subset of *A* will also support *A*. Hence, a ***belief function*** *Bel*(*A*) is defined by:

$$\begin{aligned} &Bel: \ 2^{\Theta} \to [0,1] \\ &Bel(A) = \sum_{B \subseteq A} m(B) \ \ \forall A \subseteq \Theta \end{aligned} \tag{7.20}$$

The belief function is a special type of fuzzy measure that satisfies Axioms 1 – 3 of Subsection 7.3.4.1 and the axiom of ***subadditivity*** [41]. We may consider the belief function as a generalization of the probability function. When $A = \theta_i$ is a singleton (atomic hypothesis), then $Bel(A) = Bel(\theta_i) = m(\theta_i)$. Furthermore, when *A* = *Θ*, *Bel*(*Θ*)=1.

If m_1 and m_2 are basic probability assignments on Θ, their combination or ***orthogonal sum*** for a nonempty set $A \subseteq \mathbf{2}^{\Theta}$, is defined as:

$$m(A) = m_1 \oplus m_2(A) = S^{-1} \sum_{B \cap C = A} m_1(B) \cdot m_2(C), \text{ where } B, C \subseteq \Theta \tag{7.21}$$

and $$S = \sum_{B \cap C \neq \varnothing} m_1(B) \cdot m_2(C), \ \ m(\varnothing) = 0 \tag{7.22}$$

Obviously, the combination rule may be generalized to combine multiple evidences. Since there is one-to-one correspodence between *Bel* and *m,* the orthogonal sum of belief functions is defined in the same way as: $Bel\ (.) = Bel_1 \oplus Bel_2\ (.)$.

Special kinds of *Bel*-functions are appropriate for representing evidence. These functions are called ***simple*** and ***separable support functions***. *Bel(.)* is a simple support function if there exists an $F \subseteq \Theta$ called the focus of *Bel(.),* such that *Bel*(*Θ*)=1 and

$$Bel(A) = \begin{cases} s, & \text{if } F \subseteq A \text{ and } A \neq \Theta \\ 0, & \text{otherwise} \end{cases} \tag{7.23}$$

where *s* is called *Bel*'s ***degree of support***.

A separable support function is either a simple support function or an orthogonal sum of simple support functions. Separable support funcions are very useful for combining evidences from several sources. If *Bel*(.) is a simple support function with focus $F \neq \Theta$, then $m(F) = s,\ m(\Theta) = 1 - s,$ and $m(.) = 0$ elsewhere. Let *F* be a focus for two simple support functions with degrees of support s_1 and s_2, respectively. If

$Bel = Bel_1 \oplus Bel_2$ then $m(F) = 1-(1-s_1)(1-s_2)$, $m(\Theta) = (1-s_1)(1-s_2)$, and m is 0 elsewhere.

7.3.5.2 Evidence Combination Method

In the context of measurement-level classifier combination, a method for evidence combination is presented in [196] and is also adopted here.

Let $\mathbf{x}$ be an input vector and K be the number of different classifiers, $C_i(\mathbf{x}),\ i = 1,\ldots,K$. It is also assumed that each classifier produces an output vector $\mathbf{y}_i \in \mathbb{R}^M$, $\mathbf{y}_i = C_i(\mathbf{x})$, where M is the number of classes. Suppose that for each classifier C_i and each candidate class j , a computed value $e_j(\mathbf{y}_i)$ represents some measurement of evidence for the proposition "$\mathbf{y}_i$ belongs to class *j*". In terms of the Dempster – Shafer theory, these values could be combined according to the theory and the class with the highest evidence is chosen. Thus, the important values $e_j(\mathbf{y}_i)$ need be defined and computed.

Let $\{\mathbf{t}_j\}$ be a subset of the training data corresponding to a class *j*. Let $\mathbf{r}_{i,j}$ be the mean vector for a set $\{C_i(\mathbf{t}_j)\}$ for each classifier C_i and each class *j*, representing a reference vector for that class *j*. The support function $\phi(\mathbf{r}_{i,j},\mathbf{y}_i)$ for class *j* and each classifier C_i is denoted by $c_{i,j} = \phi(\mathbf{r}_{i,j},\mathbf{y}_i)$, where ϕ can be obtained by using the Euclidean distance between $\mathbf{r}_{i,j}$ and $\mathbf{y}_i$:

$$c_{i,j} = \phi(\mathbf{r}_{i,j},\mathbf{y}_i) = \frac{\left(1+\left\|\mathbf{r}_{i,j}-\mathbf{y}_i\right\|^2\right)^{-1}}{\sum_{k=1}^{M}\left(1+\left\|\mathbf{r}_{i,k}-\mathbf{y}_i\right\|^2\right)^{-1}} \tag{7.24}$$

The value of this function is between 1 and 0 with the maximum when the output vector coincides with a reference vector. Now the function ϕ can be transformed into evidence $e_j(\mathbf{y}_i)$.

Consider a frame of discernment $\Theta = \{\theta_1,\ldots,\theta_M\}$, where θ_j is the hypothesis that "$\mathbf{y}_i$ belongs to class *j*" For every classifier C_i and class *j* , $c_{i,j}$ can represent evidence for hypothesis θ_j (*pro* θ_j), and all remaining $c_{i,k}$, $k \neq j$, can represent evidence *contra* θ_j (*pro* $\neg\theta_j$). We can use $c_{i,j}$ as a degree of support for a simple support function with focus θ_j. This yields the basic probability assignment

$$m_j\left(\theta_j\right) = c_{i,j} \quad \text{and} \quad m_j\left(\Theta\right) = 1 - c_{i,j} \tag{7.25}$$

In a similar manner, $c_{i,k}$, $k \neq j$ are degrees of support for simple support functions with a common focus $\neg\theta_j$. The combination of this simple support function with focus $\neg\theta_j$ is a separable support function with the degree of support $1 - \prod_{k \neq j}\left(1 - c_{i,k}\right)$. The corresponding basic probability assignment is

$$m_{\neg j}\left(\neg\theta_j\right) = 1 - \prod_{k \neq j}\left(1 - c_{i,k}\right) \tag{7.26}$$

and

$$m_{\neg j}\left(\Theta\right) = 1 - m_{\neg j}\left(\neg\theta_j\right) = \prod_{k \neq j}\left(1 - c_{i,k}\right) \tag{7.27}$$

Combining our knowledge about θ_j we obtain the evidence $e_j\left(\mathbf{y}_i\right) = m_j \oplus m_{\neg j}\left(\theta_j\right)$ for class j and classifier i:

$$e_j\left(\mathbf{y}_i\right) = \frac{c_{i,j}\prod_{k \neq j}\left(1 - c_{i,k}\right)}{1 - c_{i,j}\left[1 - \prod_{k \neq j}\left(1 - c_{i,k}\right)\right]} \tag{7.28}$$

Finally, evidences for all classifiers may be combined according to the ***Dempster's rule of combination*** (also called the ***orthogonal sum***) to obtain a measure of confidence for each class *j* for the feature vector **x** :

$$e_j\left(\mathbf{x}\right) = e_j\left(\mathbf{y}_1\right) \oplus e_j\left(\mathbf{y}_2\right) \oplus \ldots \oplus e_j\left(\mathbf{y}_K\right) \tag{7.29}$$

$e_j\left(\mathbf{y}_i\right)$, after an appropriate normalization, can be considered as Bayesian evidence function with nonzero basic probability assignments only on atomic hypotheses. Hence, equivalent to Eq. (7.29) we can write $e_j\left(\mathbf{x}\right) = S\prod_{i=1}^{K} e_j\left(\mathbf{y}_i\right)$, where S is a normalizing constant. Now we assign class k to the feature vector $\mathbf{x}$ if $e_k(\mathbf{x}) = \max_{j=1}^{M}\{e_j(\mathbf{x})\}$.

7.4 Experimental results

7.4.4 Classification Results obtained from Primary Classifiers based on Identical Pattern Representations

The Bayes classifier, MLP neural network classifier and LVQ neural network classifier are well-established and quite successful techniques in pattern recognition. They are employed for the primary classification task of individual leads in our component displacement estimation. The theoretical background of these classifiers has been introduced in Chapter 4.

In this research we present two types of results. The first one deals with simulated data operating in an external leave-one-out validation scheme. The results presented show the average accuracies attained for each lead displacement through this recursive cross validation scheme. The second type of results refers to testing on the real data. The training of primary classifiers is performed on the entire set of simulated data, whereas testing is performed on the completely independent set of real images. For the generation of simulated data we use the *Monte Carlo simulation* process [171-174], in order to generate lead samples with appropriate size and intensity distributions for trainining the classifiers, as has been mentioned in Subsection 5.4.4.

For testing with real images, a set of 20 real component images are kindly provided from the actual placement environment of Philips, The Netherlands. Ten actual boards with different shifts are provided, with two images from each case. Each individual case is controlled by the placement machine and conveys the limited accuracy of placement. The sides of each component are located and the individual lead areas are extracted. These images are used for testing of our developed algorithms; the training stage of classifiers is performed with the simulated data.

In order to facilitate a soft-level combination of classifier outcomes, the responses of MLP and LVQ neural networks are normalized by employing the ***softmax*** method [9], [21] and are used as estimates of the posterior probabilities of the classes.

The primary classifiers are trained for 5 and 7 classes. The first case involves classes {-6,-3, 0,+3,+6} whereas the second case considers training on classes {-6,-4,-2,0,+2,+4,+6}. These two cases study the ability of the classifiers to discriminate classes in the feature space separated by 3 and 2 pixels apart, respectively. We do not consider training on all 13 classes, since all these classes are not separable in the 12 dimensional feature space defined. To overcome the problem of statistical significance of the results caused by the rather small data set (for a quite high dimensional feature space) we apply a jack-knifing validation process. This process sweeps along all sample vectors and every time extracts one sample out of the training data. It trains the classifier with all other vectors and classifies the extracted vector that has not been seen by the classifier.

Regarding the design of the primary classifiers, the LVQ neural network architecture was defined by the feature vector size, training set size and output class mapping. In particular for use with 12 geometric (optical) features the LVQ input layer consisted of 12 neurons. In accordance to LVQ theory the hidden competitive layer contained neurons, equal to the number of training set cases. In the output layer for 5 classes (2 pixel shift precision) 5 output neurons were needed. Accordingly discrimination of 7 classes required 7 output neurons. The model was trained for 1000 epochs with a learning parameter a=0.09. The MLP neural network was designed with 50 hidden layer neurons and 5 or 7 neurons depending on the required output classes. The input layer was as above defined by the dimensionality of the feature vector. As stated before, 12 features per lead formulate the feature vector that forms the input to each classifier.

The classification rates of primary classifiers on 5 classes of simulated lead-images are shown in Table 7.1. Table 7.2 presents the classification rates of individual classifiers on 7 classes. From the classification results of primary classifiers, we can initially conclude that the Bayes classifier provides better results than the MLP and LVQ classifiers on the 5-classes case. However, competing performances of Bayes and MLP classifiers are observed on the 7-classes case. Furthermore, as can be observed in Tables 7.1, 7.2, the discrimination between different classes becomes easier as we move to larger displacement intervals; the distinction of 3-pixel difference in Table 7.1 is more efficient than that of 2-pixel difference in Table 7.2. Overall, we observe a large variance of each classifier’s performance along the classes of interest.

Table 7.1 Classification rates of Primary classifiers on 5 classes using Monte Carlo simulated images

	Input	**- 6 pixels**	**-3 pixels**	**0 pixels**	**+ 3pixels**	**+6 pixels**
	Simulated	**shift**	**shift**	**shift**	**shift**	**shift**
Classifier	**Bayes**	97.75	94.35	94.35	93.14	98.25
	MLP	95.32	93.87	92.28	95.83	97.63
	LVQ	93.27	90.67	78.24	95.42	94.78

Table 7.2 Classification rates of Primary classifiers on 7 classes using Monte Carlo simulated images

	Input	**- 6 pixels**	**- 4 pixels**	**- 2 pixels**	**0 pixels**	**+2 pixels**	**+4 pixels**	**+6 pixels**
	Simulated	**shift**	**shift**	**shift**	**shift**	**shift**	**shift**	**shift**
Classifier	**Bayes**	93.00	81.82	75.72	85.87	77.80	85.87	91.86
	MLP	91.18	80.46	80.36	79.74	82.41	86.83	88.27
	LVQ	83.47	76.42	78.90	57.06	80.40	74.35	81.16

In the sequel we test the primary classifiers on the set of 20 real component images from the actual placement environment. The testing set consists of 120 lead-images obtained from the components of the corresponding class. The classification rates of primary classifiers on 5 classes for the real lead-images are shown in Table 7.3, whereas Table 7.4 presents the classification rates of individual classifiers on 7 classes. As we observe by comparing the results for real and simulated data, there is a small decrease (ranging from 0.30 to 1.30 in different classes) in classification rates for the real data, which are used as an independent test set. Nevertheless, the results on real data are only slightly inferior to those from cross validation, indicating the robustness of developed techniques in realistic operation

Table 7.3 Classification rates of Primary classifiers on 5 classes using real images

	Input	**- 6 pixels**	**-3 pixels**	**0 pixels**	**+ 3pixels**	**+6 pixels**
	Simulated	**shift**	**shift**	**shift**	**shift**	**shift**
Classifier	**Bayes**	96.43	93.61	93.17	92.08	97.33
	MLP	94.56	93.19	91.84	94.42	96.37
	LVQ	92.73	91.24	79.66	94.77	94.19

Table 7.4 Classification rates of Primary classifiers on 7 classes using real images

	Input	**- 6 pixels**	**- 4 pixels**	**- 2 pixels**	**0 pixels**	**+2 pixels**	**+4 pixels**	**+6 pixels**
	Simulated	**shift**	**shift**	**shift**	**shift**	**shift**	**shift**	**shift**
Classifier	**Bayes**	91.87	80.26	74.66	85.26	77.32	85.42	91.03
	MLP	90.33	79.74	79.23	78.44	81.92	86.05	87.49
	LVQ	83.16	77.28	79.57	56.59	79.63	73.94	80.71

7.4.5 Results of Combined Classifiers using Identical Pattern Representations

The methods of Section 7.3 are used here to combine the three primary classifiers (Bayes, MLP, LVQ), using different methodologies but operating on the same feature sets (optical). For such a three-classifier combination case, the combining scheme based on majority voting (MV) assigns classification to one class if two or three classifiers produce this same class. Otherwise, the input pattern is rejected. To apply the naïve Bayes (NB) combination method, the conditional probabilities $P(L_i \mid \omega_k)$, $i = 1,2,3$, $k = 1,\ldots,5$ or $k = 1,\ldots,7$, are obtained from the resulting confusion matrices of individual classifiers on the training set. In a same manner, to fuse the results using the Choquet fuzzy integral (CFI) , the initial fuzzy densities $g_i, i = 1,2,3$, are computed from the resulting confusion matrices of individual classifiers on the training set. The above three combination methods are performed in the context of abstract-level combination. The Dempster-Shafer (D-S) fusion is performed in the context of measurement-level classifier combination. Thus, the training of the D-S combiner is performed based on the presented method in Subsection 7.3.5.2.

The classification results obtained from the above four combiners using identical (optical) features based on simulated lead-images are presented in tables 7.5 and 7.6 on 5 and 7 classes, respectively. As we observe from these tables, all combination classifiers achieve better performance than any individual classifier used for fusion. Examining deeper their performance, we conclude that the naïve Bayes and the Dempster – Shafer combiners achieve better overall performance than the other schemes, with the naïve Bayes reaching the best performance of all combining classifiers employed. The largest improvement achieved by the combined classifiers over the best individual classifier performance is also depicted in Tables 7.5 and 7.6 for the naïve Bayes scheme. In fact, maximum improvement (3.98 %) is achieved by this fusion approach for the class of –3 pixels shift on the 5 class formulation. The advantage of naïve Bayes combiner over the others fusion schemes, along with the advantage of primary Bayes classifier over the others individual classifiers, cannot be generalized. The ranking of classification schemes observed in this application is partially attributed to the stochastic properties of the data set, supporting the assumption that the distribution of our experimental data follows the normal (Gaussian) distribution.

Table 7.5 Classification rates of Combining Classifiers on 5 classes using identical (optical) features based on simulated images

	Input	**- 6 pixels**	**-3 pixels**	**0 pixels**	**+ 3pixels**	**+6 pixels**
	Simulated	**shift**	**shift**	**shift**	**shift**	**shift**
Combiner	**MV**	98.25	95.13	94.78	96.41	98.43
	NB	**99.20**	**98.33**	**97.21**	**98.84**	**99.87**
		(>1.45)	**(>3.98)**	(>2.86)	(>3.01)	(>1.62)
	CFI	98.34	95.89	95.44	96.90	98.79
	D-S	98.67	97.71	96.63	97.56	99.36

Table 7.6 Classification rates of Combining Classifiers on 7 classes using identical (optical) features based on simulated images

	Input	- 6 pixels	- 4 pixels	- 2 pixels	0 pixels	+2 pixels	+4 pixels	+6 pixels
	Simulated	**shift**	**shift**	**shift**	**shift**	**shift**	**shift**	**shift**
Combiner	**MV**	94.12	82.53	81.05	86.38	83.44	87.85	92.48
	NB	**96.82**	**85.49**	**83.87**	**88.79**	**85.21**	**90.35**	**95.53**
		(>3.82)	(>3.67)	(>3.51)	(>2.92)	(>2.80)	(>3.52)	(>3.67)
	CFI	95.27	83.76	82.64	87.30	84.37	88.54	93.86
	D-S	95.79	84.88	83.26	88.14	84.69	89.87	94.64

In the sequel we derive the classification results using the four combination schemes based on the classifiers Bayes, MLP and LVQ employing real images, given in Tables 7.3 and 7.4. These results are presented in Tables 7.7 and 7.8 on 5 and 7 classes, respectively. By comparing these results with Tables 7.5 and 7.6, we can detect a small decrease (ranging from 0.30 to 1.30 in different classes) in classification rates from the case of testing simulated data, which can be attributed to small differences in the formation of the training and the testing data. Nevertheless, by comparing them with the results of individual classifiers on real image data (Tables 7.3 and 7.4), we observe a consistent increase of the success rate achieved by any fusion methodology.

Table 7.7 Classification rates of Combined Classifiers on 5 classes using identical (optical) features based on real images.

	Input	- 6 pixels	-3 pixels	0 pixels	+ 3pixels	+6 pixels
	Simulated	**shift**	**shift**	**shift**	**shift**	**shift**
Combiner	**MV**	97.16	94.42	93.56	95.73	98.04
	NB	**97.70**	**96.97**	**95.56**	**97.61**	**98.61**
		(>1.27)	**(>3.36)**	(>2.39)	(>2.84)	(>1.28)
	CFI	96.95	94.88	95.00	96.49	98.40
	D-S	97.26	95.76	94.63	96.15	98.27

Table 7.8 Classification rates of Combined Classifiers on 7 classes using identical (optical) features based on real images

	Input	**- 6 pixels**	**- 4 pixels**	**- 2 pixels**	**0 pixels**	**+2 pixels**	**+4 pixels**	**+6 pixels**
	Simulated	**shift**	**shift**	**shift**	**shift**	**shift**	**shift**	**shift**
Combiner	**MV**	92.65	80.69	79.94	85.90	82.73	86.89	91.61
	NB	**95.33** **(>3.46)**	**83.50** (>3.24)	**82.76** (>3.19)	**87.84** (>2.58)	**84.29** (>2.37)	**89.33** (>3.28)	**94.44** (>3.41)
	CFI	93.69	81.80	81.43	86.45	83.67	87.53	92.68
	D-S	94.23	82.98	80.94	87.30	83.77	88.48	93.61

7.4.6 Comparison between primary and combining multiple classifiers

The classifier ensemble's high accuracies can be partially attributed to the diversity of the three primary classifiers. It is the author's opinion that an additional improvement can be achieved in the 7-class case by enriching the primary classifier's pool. This would require a very careful choice of additional classifiers that would contribute to the ensemble's diversity, if possible. The 5-class case is less likely to benefit since the obtained accuracies are already nearly maximized. Such a refinement might also render 1-pixel resolution shift estimation (13 classes) manageable. In any case, one has to keep in mind that model complexity should not outweigh possible minimal gains and that results have to be extended to other datasets.

The incresead computational complexity of fusion in a real time inspection system was also a factor considered. The overhead in a multiple classification process of this type is additive. This problem is addressed in three ways towards minimizing this overhead. Firstly the number of classes is kept to a minimum required for quality inspection by quantizing the output displacements. Secondly, the features used were chosen so that no intensive image processing or costly transformations are involved in their computation. Thirdly, a minimal primary classifier pool is used whist maintaining a decent misclassification rate.

It should also be noted that the classification problem under consideration presents a special nature with both negative and positive aspects. The primary classifiers's soft outputs are mostly concentrated around the distinct class labels, which can create mapping difficulties for certain combined classifiers. From a different point of view, classifiers in this application area can benefit from certain symmetries and prior knowledge inherent to the problem. Limiting the displacements to one axis (for the corresponding component side) reduces the degrees of freedom in problem specification and classifier design. Additionally, the shape and size areas is roughly known or can be easily inferred for any new dataset and thus geometry metrics can be used reliably.

Chapter 8

Combining Multiple Classifiers using Reduced Dimensionality Distinct Pattern Representations for Post Placement Quality Inspection of Components

8.1 Introduction

As we have mentioned in the previous chapter, from the point of view of the ***input pattern representation***, there are basically two classifier combination approaches [187]. In the first approach, all the classifiers use the same represantation of the input pattern (***identical pattern representation***). In the second approach, each classifier uses its own representation of the input pattern (***distinct pattern representation***) [187], [210-216] . In this case, the measurements extracted from the pattern are unique to each classifier, i.e. each individual classifier uses a different set of features. In Chapter 7, we have tested and compared multiple classifier fusion methods for improving the classification of the individual leads in component quality inspection, based on identical pattern representations (optical features). The objective of our research presented in this chapter is to fuse decisions from primary classifiers, which operate on distinct pattern representations. **This research has been accepted for publication in [48]**.

The methods used the combine the various levels of base classifier output generally fall into two categories namely, ***fixed rules*** and ***trained rules*** (i.e. ***nontrainable combiners*** and ***trainable combiners***) [21], [188], [214]. Fixed rules are static in that their form and parameters do not change as a result of the output produced by the base classifiers. As such, they are simple, have low time and memory requirements and are well suited to groups of classifiers that exhibit similar performances and make uncorellated errors. Fixed rules include the Dempster – Shafer theory of evidence [186], [196], [198-200], the sum-rule (averaging), product-rule, max-rule, min-rule, median rule [187], the majority vote rule [186], etc. On the other hand, trained rules adapt their parameters to the outputs of the base classifiers and as such are more suitable for combining classifiers that have varying levels of performance and make correlated errors. These rules generally have high memory and computational needs and impose strict requirements on the quality and size of the training sets. Trained rules include various trainable combiners (called ***meta-classifiers*** or ***meta-learners***), including statistical combiners [21], [194-195] neural networks [207], fuzzy integral combiners [15], [169], [192], [209] and typically, the weighted combination (weighted average) [21], [192].

In this work we elaborate on two schemes for distinct pattern representations. In the former scheme we use only reduced dimensionality features (i.e., ***topological*** and ***projection features***), whereas the latter enriches the topological and projection features with optical ones, in order to improve the classification rates and robustness across all lead-displacement classes. In the abovementioned former scheme for distinct pattern representations we use the quantized classification of lead displacements based on reduced dynamic-range and input-dimension processing of lead images. The classification task of individual leads is executed via two different primary classifiers. The first classifier is a Hamming neural network classifier based on reduced dynamic-range processing (topological features). The second classifier is a Bayesian distance classifier based on input-dimension processing (projection features). Both aforementioned classifiers have been developed and tested in Chapter 6. Finally, in the latter scheme for distinct patterns we enrich the pool of primary classifiers with a Bayes classifier operating on ***optical features***. This classifier has been developed and tested in Chapter 5. Thus, the goal of this research is to fuse decisions from the aforementioned three classifiers that are based on distinct pattern representations.

The motivation for exploring the combination issue is to improve performance of classification task of individual leads based on distinct pattern representations. Kittler [20], [187] provides a theoretical basis of many existing classifier combination schemes for fusing the decisions of multiple experts, each employing a different, distinct pattern representation. In this research, we adopt and explore this theoretical framework, in order to design and test non-trainable classifier fusion schemes. The fixed combination rules are motivated in this work from the completely different nature of the feature sets used by our primary classifiers, justifying the assumption of complementary and uncorrelated classification results.

8.2 Experimental Framework and Reduced Dimensionality Feature Extraction Processes

In this research we use the experimental framework presented in Subsection 5.1.2. So, for the purpose of presenting our results, QFP (Quad Flat Pack) SMD components with 120 leads (30 leads per side) are employed. The Hamming distance classifier uses the reduced dynamic-range processing introduced in Section 6.3 whereas the Bayesian distance classifier uses the HONN based feature extraction process (i.e. reduced input-dimension processing) presented in Section 6.4. Finally, the Bayes classifier operating on optical features has been presented in Section 5.5. As in Chapters 5, 6 and 7 we consider again quantized displacement estimations organized at multiples of a pixel displacement. The displacement classes considered are {-6, -3, 0, +3, +6} and {-6, -4, -2, 0, +2, +4, +6}, in pixel displacements over the lead's central position.

8.3 Combining Multiple Classifiers based on Distinct Pattern Representations

In this section, the problem of combining different classifiers using distinct pattern representations is addressed for the classification (shift-estimation) of individual lead images.

8.3.1 Non-trainable Combination Schemes for Identical Pattern Representations

The term "nontrainable" implies that the combiner has no extra parameters that need to be trained, i.e. the ensemble is ready for operation as soon as the primary classifiers are trained [21]. Simple non-trainable combiners calculate the overall support for class ω_j, $\mu_j(\mathbf{x})$, using only the supports $c_{1,j}(\mathbf{x}), c_{2,j}(\mathbf{x}), ..., c_{K,j}(\mathbf{x})$ from classifiers $C_1, C_2, ..., C_K$, respectively, by

$$\mu_j(\mathbf{x}) = F\left[c_{1,j}(\mathbf{x}), c_{2,j}(\mathbf{x}), ..., c_{K,j}(\mathbf{x})\right], \tag{8.1}$$

where F is a ***combination function***. The class label of $\mathbf{x}$ is found as the index of the maximum $\mu_j(\mathbf{x})$. The combination function F can be chosen in many different ways. Some popular choices are:

- ***Simple mean (average)*** , (F = average):

$$\mu_j(\mathbf{x}) = \frac{1}{K}\sum_{i=1}^{K} c_{i,j}(\mathbf{x}) \tag{8.2}$$

- ***Minimum/maximum/median*** (F = minimum/maximum/median). For instance, in the maximum case:

$$\mu_j(\mathbf{x}) = \max_{i=1}^{K}\left\{c_{i,j}(\mathbf{x})\right\} \tag{8.3}$$

- ***Product*** (F = product):

$$\mu_j(\mathbf{x}) = \prod_{i=1}^{K} c_{i,j}(\mathbf{x}) \tag{8.4}$$

8.3.2 Combination Rules for Distinct Pattern Representations

The case of distinct pattern representation poses an additional burden to the design of the combiners, since the sources of information (features) are quite inhomogeneous. Nevertheless, based on a Bayesian framework for relating the available information, similar simple rules can be derived for the combination of the corresponding classifiers. Assume that K classifiers are available, each representing the given pattern by a distinct feature vector. We consider K conditionally independent feature subsets (distinct pattern representations). Each subset generates a part of the feature vector (i.e., ***distinct feature vector***), $\mathbf{x}^{(i)}$, so that $\mathbf{x} = \left[\mathbf{x}^{(1)}, \mathbf{x}^{(2)}, ..., \mathbf{x}^{(K)}\right]^T$, $\mathbf{x} \in \mathbb{R}^n$. Notice that there is an one-to-one correspondence between each feature vector $\mathbf{x}^{(i)}$ and its underlying classifierC_i, $i = 1, \ldots, K$. From the assumed independence, the class-conditional probability density function (pdf) for class ω_j is a product of the class-conditional pdfs on each feature subset

$$p\left(\mathbf{x} \mid \omega_j\right) = \prod_{i=1}^{K} p\left(\mathbf{x}^{(i)} \mid \omega_j\right) \tag{8.5}$$

The class conditional probabilities are given by:

$$p\left(\mathbf{x}^{(i)} \mid \omega_j\right) = \frac{P\left(\omega_j \mid \mathbf{x}^{(i)}\right) p\left(\mathbf{x}^{(i)}\right)}{P\left(\omega_j\right)} \tag{8.6}$$

whereas the posterior probability using the entire information on **x** is

$$P\left(\omega_j \mid \mathbf{x}\right) = \frac{P\left(\omega_j\right) p\left(\mathbf{x} \mid \omega_j\right)}{p\left(\mathbf{x}\right)} = \frac{P\left(\omega_j\right)}{p\left(\mathbf{x}\right)} \prod_{i=1}^{K} p\left(\mathbf{x}^{(i)} \mid \omega_j\right) \tag{8.7}$$

Substituting Eq. (8.6) into Eq. (8.7) we obtain:

$$P\left(\omega_j \mid \mathbf{x}\right) = P^{(1-K)}\left(\omega_j\right) \prod_{i=1}^{K} P\left(\omega_j \mid \mathbf{x}^{(i)}\right) \frac{\prod_{i=1}^{K} p\left(\mathbf{x}^{(i)}\right)}{p\left(\mathbf{x}\right)} \tag{8.8}$$

The last fraction does not depend on any class label, so that it can be ignored when calculating the overall support $\mu_j\left(\mathbf{x}\right)$ for class ω_j. Taking the classifier output $c_{i,j}\left(\mathbf{x}^{(i)}\right)$ as the estimate of $P\left(\omega_j \mid \mathbf{x}^{(i)}\right)$ and estimating the prior probabilities for the classes from the data, the support for ω_j is calculated as the ***product combination rule*** [21]:

$$P\left(\omega_j \mid \mathbf{x}\right) \propto P^{(1-K)}\left(\omega_j\right) \prod_{i=1}^{K} P\left(\omega_j \mid \mathbf{x}^{(i)}\right) \tag{8.9}$$

so that we can assign

$$\mu_j\left(\mathbf{x}\right) = \hat{P}^{(1-K)}\left(\omega_j\right) \prod_{i=1}^{K} c_{i,j}\left(\mathbf{x}^{(i)}\right) \tag{8.10}$$

Kittler et al. [187] take the above formula further to derive the sum combination rule. Suppose that classifiers $C_i,\ i = 1,...,K$, only slightly improve on the accuracy of the classification decision. In other words, the posterior probabilities differ only by a small fraction $\Delta_{j,i}$, $j = 1,...,M,\ i = 1,...,K$ from the prior probabilities $P\left(\omega_j\right),\ j = 1,...,M$, where $\left|\Delta_{j,i}\right| \ll 1$. Thus:

$$P\left(\omega_j \mid \mathbf{x}^{(i)}\right) = P\left(\omega_j\right)\left(1 + \Delta_{j,i}\right) \tag{8.11}$$

Substituting in Eq. (8.9), we obtain

$$P\left(\omega_j \mid \mathbf{x}\right) \propto P\left(\omega_j\right) \prod_{i=1}^{K} \left(1 + \Delta_{j,i}\right) \tag{8.12}$$

Expanding the product and ignoring all terms of order higher than two with respect to $\Delta_{j,i}$, we obtain [21]:

$$P\left(\omega_j \mid \mathbf{x}\right) \propto P\left(\omega_j\right)\left(1 - K\right) + \sum_{i=1}^{K} P\left(\omega_j \mid \mathbf{x}^{(i)}\right) \tag{8.13}$$

Taking the classifier output $c_{i,j}\left(\mathbf{x}^{(i)}\right)$ as the estimate of $P\left(\omega_j \mid \mathbf{x}^{(i)}\right)$ and estimating the prior probabilities for the classes from the data, the overall support $\mu_j(\mathbf{x})$ for class ω_j is calculated as the ***sum combination rule***:

$$\mu_j(\mathbf{x}) = \hat{P}\left(\omega_j\right)(1-K) + \sum_{i=1}^{K} c_{i,j}\left(\mathbf{x}^{(i)}\right) \tag{8.14}$$

For equal prior probabilities Eq. (8.9) reduces to:

$$P\left(\omega_j \mid \mathbf{x}\right) \propto \prod_{i=1}^{K} c_{i,j}\left(\mathbf{x}^{(i)}\right) \tag{8.15}$$

Furthermore, ignoring the constant term, the sum combination rule (8.14) can be viewed as the average a posteriori probability for each class over all the classifier outputs [187], so that we may assign:

$$\mu_j(\mathbf{x}) = \frac{1}{K}\sum_{i=1}^{K} c_{i,j}\left(\mathbf{x}^{(i)}\right) \tag{8.16}$$

The aforementioned combination rules (8.10) and (8.14) constitute the fundamental schemes for combining classifiers, each representing the given pattern by a distinct feature vector. Some additional nontrainable fusion strategies can be developed from these rules by considering the inequalities:

$$\prod_{i=1}^{K} c_{i,j}\left(\mathbf{x}^{(i)}\right) \le \min_{i=1}^{K}\left\{c_{i,j}\left(\mathbf{x}^{(i)}\right)\right\} \le \frac{1}{K}\sum_{i=1}^{K} c_{i,j}\left(\mathbf{x}^{(i)}\right) \le \max_{i=1}^{K}\left\{c_{i,j}\left(\mathbf{x}^{(i)}\right)\right\} \tag{8.17}$$

The relationship (8.17) suggests that the product and sum combination rules can be approximated by their upper or lower bounds, as appropriated above.

Starting from (8.14) and approximating the sum by the maximum of the support values $c_{i,j}\left(\mathbf{x}^{(i)}\right)$, the overall support $\mu_j(\mathbf{x})$ for class ω_j is calculated as the ***max combination rule***

$$\mu_j(\mathbf{x}) = \hat{P}\left(\omega_j\right)(1-K) + K\max_{i=1}^{K}\left\{c_{i,j}\left(\mathbf{x}^{(i)}\right)\right\} \tag{8.18}$$

which under the assumption of equal prior probabilities reduces to

$$\mu_j(\mathbf{x}) = \max_{i=1}^{K}\left\{c_{i,j}\left(\mathbf{x}^{(i)}\right)\right\} \tag{8.19}$$

Starting from (8.10) and bounding the product of support values $c_{i,j}\left(\mathbf{x}^{(i)}\right)$ from above, the overall support $\mu_j\left(\mathbf{x}\right)$ for class ω_j reduces to the ***min combination rule***

$$\mu_j\left(\mathbf{x}\right)=\widehat{P}^{(1-K)}\left(\omega_j\right)\min_{i=1}^{K}\left\{c_{i,j}\left(\mathbf{x}^{(i)}\right)\right\} \tag{8.20}$$

which under the assumption of equal prior probabilities reduces to

$$\mu_j\left(\mathbf{x}\right)=\min_{i=1}^{K}\left\{c_{i,j}\left(\mathbf{x}^{(i)}\right)\right\} \tag{8.21}$$

8.4 Experimental Results

8.4.1 Classification Results obtained from Primary Classifiers based on Distinct Pattern Representations

In this section the Hamming neural network classifier operating on topological features and Bayesian classifier operating on projection features are employed for the primary classification task of individual leads in our component displacement estimation.

In this research, as in Chapter 7, we present two types of results. The first one deals with Monte Carlo simulated data operating in an external leave-one-out validation scheme. The results presented show the average accuracies attained for each lead displacement through this recursive cross validation scheme. The second type of results refers to testing on the real data. The training of primary classifiers is performed on the entire set of simulated data, whereas testing is performed on the completely independent set of real images.

The *reduced dynamic-range approach* developed Section 6.3 of Chapter 6 is tested on simulated images from four-sided QFP components. A total of 120 lead samples per class of the lead displacement are obtained, resulting in totally 1560 samples for the 13 classes. We consider the ***Hamming neural network classifier*** trained and tested for 5 and 7 classes. The first case involves pixel displacements {-6,-4,-2,0,+2,+4,+6} whereas the second case considers classes {-6, -3, 0, +3, +6}. A total of 600 and 840 sample-leads are used for training and testing the classifier on five and seven classes, respectively. These two cases study the ability of the classifier to discriminate classes in the feature space separated by 3 and 2 pixels apart, respectively.

The *reduced input-dimension processing* approach developed in Section 6.4 of Chapter 6 is also tested on simulated images from QFP components. The ***Bayesian distance classifier*** is trained and tested for 5 and 7 classes. As before, a total of 600 and 840 sample-leads are used for testing the classifier on five and seven classes, respectively.

In order to facilitate a soft-level combination of classifier outcomes, the responses of Hamming neural network are normalized by employing the ***softmax*** method [9], [21] and are used as estimates of the posterior probabilities of the classes.

The classification rates of primary classifiers operating on the reduced dimensionality features on 5 and 7 classes are illustrated in Tables 8.1 and 8.2, respectively. The classification results of the Bayes classifier operating on the optical features are also presented in these tables. For the Hamming-distance classifier as expected, the results for the 5-classes assignment are more accurate than the 7-classes case, where the larger lead-displacement differences between two successive classes are reflected more clearly in the feature-vector differences. Nevertheless, due to the preserved correlation between AMs of neighboring classes, the discrimination ability is limited even in the 5-classes assignment. Moreover, the percentage of correct classification differs significantly among classes. The latter problem can be alleviated with more extensive testing, but the former is an inherent limitation attributed to dimensionality reduction. From the classification results of the Bayesian classifier operating on projection features we again conclude that the classification results for the 5-classes assignment are more accurate than these for the 7-classes problem. The limitations due to information reduction are again clear and are attributed to similarities of the projection functions. In this case, the direction of displacement poses an additional difficulty in its correct identification and classification.

Comparing the results of the two proposed primary classifiers based on reduced dimensionality processing, we can easily derive that none of them has an overall superior performance over the other. The information loss and the associated effects are different in the two approaches. In general, the reduced dynamic-range processing is more effective in discriminating 0-displacement features, whereas the reduced input-dimension processing is more efficient in classifying lead features reflecting actual displacements. The major benefit of reduced dimensionality processing (with either of the proposed versions) is the significant reduction of the algorithmic computational complexity, by avoiding processing the entire information content of the grayscale image. In order to preserve this benefit and still retain high success rates in classification, we consider decision fusion approaches as to improve the performance of the overall quality inspection system.

Table 8.1 Classification rates of Primary classifiers on 5 classes using Monte Carlo simulated images

	features type	**Classifier type**	- 6 pixels shift	-3 pixels shift	0 pixels shift	+ 3pixels shift	+6 pixels shift
Classifier	topological	Hamming	86.67	79.17	95.00	92.50	93.33
	projection	Bayes	92.50	93.33	86.70	93.33	97.63
	optical	Bayes	97.75	94.35	94.35	93.14	98.25

Table 8.2 Classification rates of Primary classifiers on 7 classes using Monte Carlo simulated images

	features type	**Classifier type**	- 6 pixels shift	- 4 pixels shift	- 2 pixels shift	0 pixels shift	+2 pixels shift	+4 pixels shift	+6 pixels shift
Classifier	topological	Hamming	85.00	84.17	77.50	92.50	82.50	82.50	88.33
	projection	Bayes	76.70	80.83	86.07	68.64	83.30	90.00	93.10
	optical	Bayes	93.00	81.82	75.72	85.87	77.80	85.87	91.86

In the sequel we test the primary classifiers on the set of 20 real component images from the actual placement environment. The testing set consists of 120 lead-images obtained from the components of the corresponding class. The classification rates of primary classifiers on 5 classes for the real lead-images are shown in Table 8.3, whereas Table 8.4 presents the classification rates of individual classifiers on 7 classes. As we observe by comparing the results for real and simulated data, there is a small decrease (ranging from 0.30 to 1.30 in different classes) in classification rates for the real data, which are used as an independent test set. Nevertheless, the results on real data are only slightly inferior to those from cross validation, indicating the robustness of developed techniques in realistic operation.

Table 8.3 Classification rates of Primary classifiers on 5 classes using real images

	features type	**Classifier type**	- 6 pixels shift	-3 pixels shift	0 pixels shift	+ 3pixels shift	+6 pixels shift
Classifier	topological	Hamming	85.83	78.56	94.13	92.06	92.79
	projection	Bayes	91.46	93.76	87.24	92.95	97.18
	optical	Bayes	96.43	93.61	93.17	92.08	97.33

Table 8.4 Classification rates of Primary classifiers on 7 classes using real images

	features type	**Classifier type**	- 6 pixels shift	- 4 pixels shift	- 2 pixels shift	0 pixels shift	+2 pixels shift	+4 pixels shift	+6 pixels shift
Classifier	topological	Hamming	84.25	83.79	76.63	91.16	81.59	81.37	87.75
	projection	Bayes	75.68	79.96	84.75	67.60	82.45	89.26	92.53
	optical	Bayes	91.87	80.26	74.66	85.26	77.32	85.42	91.03

8.4.2 Results of Combined Classifiers using Distinct Pattern Representations

In this section, the combination rules derived in Subsection 8.3.2 are used to combine the two primary classifiers (Hamming classifier and Bayesian classifier), using distinct pattern representations for individual leads classification. The quite diverse nature of information handled by each approach justifies the assumption of class conditional independence (at least approximately) for the distinct representations used by the individual classifiers. Four different combination rules are tested under the assumption of equal priors and their results are compared. Each combiner uses the outcomes of primary classifiers as estimates of a posterior class probability, in a soft-level combination manner.

The classification results obtained from the above four combination rules are presented in Tables 8.5 and 8.6 for 5 and 7 classes, respectively. As we observe from these tables, the sum combination rule achieves better performance than any individual classifier alone with the exception of the class of –3 pixels shift on the 5 class formulation and class of 0 pixels shift on the 7 class formulation. The max combination rule follows closely in performance, whereas the worst results are achieved when using the product and min combination rules. The results are in close agreement with the findings of [187], based on a theoretical error sensitivity analysis, where the sum combination rule is found to be much more resilient to estimation errors of the posterior probabilities $P\left(\omega_j \mid \mathbf{x}^{(i)}\right)$than the product combination rule. In particular, the product combiner is oversensitive to classification estimates close to zero. Presence of such estimates from one classifier has the effect of veto on that particular class, regardless the outcome of other classifiers.

We should further emphasize that fusion may not improve the classification results for each and every lead displacement compared to the individual classifiers, but it rather improves the overall classification ability for all lead-shifts examined. Even though fusion increases the classification accuracy for lead shifts where individual classifiers generally lag in performance, there are a few cases where one or the other individual classifier (based on topological or projection features) by chance achieves extremely high accuracy. The results of primary classifiers show a large variance of performance across the lead displacements, as in Tables 8.1 and 8.2 or 8.3 and 8.4 for simulated and real data, respectively. From these results, we cannot claim that one individual classifier, either Hamming based on topological or Bayes based on projection features, surpasses the other in performance. Each one attains maximum performance by chance at some specific lead displacement. We cannot generalize such results of individual classifiers due to the limited number of available data. Notice that this large variation is reduced by the fusion approaches. Thus, fusion using distinct, reduced-content representations not only boost the overall classification performance, but also makes the overall classification performance more consistent across all lead-displacements examined.

Table 8.5 Classification rates of Combined Classifiers on 5 classes using distinct features (topological & projection) based on simulated images

	Input Simulated	- 6 pixels shift	-3 pixels shift	0 pixels shift	+ 3pixels shift	+6 pixels shift
Combination Rule	**Product**	83.08	78.25	84.23	88.76	89.68
	Sum	**94.64**	**90.68**	**95.67**	**97.42**	**99.73**
	Max	90.78	87.14	91.62	93.46	96.09
	Min	84.33	79.04	85.59	89.94	90.81

Table 8.6 Classification rates of Combined Classifiers on 7 classes using distinct features (topological & projection) based on simulated images

	Input Simulated	-6 pixels shift	- 4 pixels shift	- 2 pixels shift	0 pixels shift	+2 pixels shift	+4 pixels shift	+6 pixels shift
Combinat ion Rule	**Product**	71.24	75.81	74.41	70.93	76.13	78.95	83.76
	Sum	**85.08**	**86.63**	**86.12**	**85.91**	**87.38**	**90.53**	**94.97**
	Max	81.59	83.43	82.36	81.22	83.87	87.11	91.55
	Min	72.09	76.88	75.23	71.64	77.39	79.83	85.03

Considering the classification of real data, the results of these four combination rules are presented in Tables 8.7 and 8.8 for the 5 and 7 class formulations, respectively. We recall that the individual classifiers used at first level are the Hamming neural network operating on topological features and the Bayes classifier operating on projection features extracted from the set of real images considered. As can be observed in Tables 8.7 and 8.8, the sum combiner again achieves overall better results, but there is a small decrease (ranging from 0.30 to 1.30 in different classes) in classification rates in comparison with Tables 8.5 and 8.6 for the simulated data.

Table 8.7 Classification rates of Combined Classifiers on 5 classes using distinct features (topological & projection) from real images

	Input	- 6 pixels	-3 pixels	0 pixels	+ 3pixels	+6 pixels
	Simulated	shift	shift	shift	shift	shift
Combination Rule	Product	81.89	77.34	85.00	87.93	89.44
	Sum	**93.33**	**91.38**	**94.61**	**92.77**	**94.55**
	Max	89.28	87.94	90.47	94.38	95.35
	Min	83.09	78.38	85.79	88.94	90.55

Table 8.8 Classification rates of Combined Classifiers on 7 classes using distinct features (topological & projection) from real images

	Input	-6 pixels	- 4 pixels	- 2 pixels	0 pixels	+2 pixels	+4 pixels	+6 pixels
	Simulated	shift	shift	shift	shift	shift	shift	shift
Combination Rule	Product	69.97	75.14	73.06	69.54	75.54	77.45	82.94
	Sum	**84.68**	**85.98**	**85.12**	**84.75**	**86.08**	**89.63**	**93.95**
	Max	80.41	82.77	81.31	86.10	81.67	86.01	91.24
	Min	70.85	76.00	73.98	70.04	77.03	77.86	84.93

In general, the classification scores achieved using reduced dimensionality features are inferior to those obtained using full image information (optical features) as has been mentioned in Chapter 6. Furthermore, the combination of topological and projection features in a distinct representation fusion scheme also lags in performance to the combination of classifiers trained with optical features alone in Subsection 7.4.2. This is expected since all feature sets are obtained from the same primary source (original lead images), so that the information captured by topological and projection features does not add much to the information already conveyed by optical features. Furthermore, the primary data in reduced content representation (1-bit edge images and 1-D projections) are inter-related, rendering the corresponding features (topological and projection) not quite independent. At this stage we do not perform any feature selection process, since we are focusing on the nature of primary data (edges and projections) and the information conveyed in these forms, rather than the nature of features. Nevertheless, it is worth mentioning that reduced dimensionality features using just a portion of information available can still attain acceptable results, especially through the employment of fusion. Reduced dimensionality features have the benefit of summarizing the required information for adequate shift detection in a compact format that can significantly reduce processing time. It is the author's opinion that such features should be used when balance is required between speed and effectiveness. In addition, the particular reduced dimensionality features possess conceptual attributes that can instigate further speed-up and improvement in component inspection systems. More specifically, the topological features (edges) may be used for appropriate modeling of the component placement process and can be directly obtained from a number of commercial cameras, eliminating the need of preprocessing. The projection features on the other hand may eventually enable the use of faster and cheaper line sensors instead of area cameras for component inspection.

Elaborating on the use of distinct feature representations and its potential in increasing accuracy and robustness for all classes of lead displacements, we further consider a combination of optical, topological and projection features. We define the resulting distinct features set (i.e., *optical* & *topological & projection features)* as ***distinct features-2***. This set of features captures information form many different aspects of the problem and contains features that are more likely to be independent than the set used before employing only topological and projection features. The quite diverse nature of information handled by each approach justifies the assumption of class conditional independence (at least approximately) for the distinct representations used by the individual classifiers. Motivated by good results of the sum combination rule we also use it as a fusion rule on the Bayes classifier with optical features, Hamming classifier with topological features and Bayes classifier with projection features. The classification rates based upon Monte Carlo simulated and real images are presented in Tables 8.9 and 8.10 on 5 and 7 classes, respectively. We observe that the sum combination rule achieves better performance than any individual classifier alone based on distinct features-2, with the exception of the class of 0 pixels shift on the 7 class formulation. It improves the results of the first level Bayes classifier and derives quite uniform results across all classes. Comparing this distinct feature combination with the one in Subsection 7.4.2 of Chapter 7 using identical pattern representation, we can claim that the former achieves comparable and at cases (7-class formulation) even better performance than the latter. This result further supports the potential of the distinct representation scheme, requiring however further investigation on the appropriate selection of distinct features, which is out of the scope of this research.

Table 8.9 Classification rates of Sum Combination Rule on 5 classes using distinct features-2 (optical & topological & projection) on simulated and real images

Sum Combination Rule	- 6 pixels shift	-3 pixels shift	0 pixels shift	+ 3pixels shift	+6 pixels shift
Simulated Images	98.37	97.14	96.27	97.45	99.73
Real Images	97.49	95.53	95.04	95.52	98.92

Table 8.10 Classification rates of Sum Combination Rule on 7 classes using distinct features-2 (optical & topological & projection) on simulated and real images

Sum Combination Rule	-6 pixels shift	- 4 pixels shift	- 2 pixels shift	0 pixels shift	+2 pixels shift	+4 pixels shift	+6 pixels shift
Simulated Images	93.65	88.25	86.92	92.70	87.53	91.46	95.19
Real Images	92.45	86.66	85.44	92.28	86.18	90.27	94.12

With respect to time requirements, as we have mentioned in Chapter 6, the tested approaches achieve the following performance using a fast Intel Core 2 Duo workstation. The optical feature approach takes about 0.34 sec for processing an entire QFP chip of 120 leads. The reduced dynamic-range approach requires 0.15 sec, less than half of the computation time of the conventional approach. Finally, the reduced input-dimension processing requires about 0.22 sec for the entire QFP-120 component.

High abstraction features are generally less descriptive than pixel-based features for classification purposes. With respect however to computational complexity, the distinct features in cooperation with a fusion scheme can yield appreciable reduction at the cost without compromising the effectiveness of inspection.

8.4.3 Comparison between the two Fusion Schemes Presented in Chapters 7 and 8

In Chapters 7 and 8, we tested several combination methods for soft fusion of the outputs of multiple classifiers. The aim is to improve the performance of primary classifiers used for individual lead-image classification in post-placement quality inspection of components. Two different schemes of classifier fusion are considered. The first one refers to identical feature representations, where the primary classifiers operate on the same feature set. The second scheme uses distinct feature representations, where each of the primary classifiers operates on a different set of features. Comparing the classification results of the proposed combined classifiers, we can derive that all combiners have better prformance that any individual classifier alone. In addition to, it is verified that both the naïve Bayes and the Dempster – Shafer combiners on identical feature representations achieve better overall performance, with the naïve Bayes reaching the best performance improvement over the primary classifiers. The combiners based on distinct feature representations present lower overall performance and higher variability of their results. This is expected due to reduced content of information exploited. Despite that, their performance is still better than that of most primary classifiers, showing a good potential for accelarating the inspection process when speed can be balanced against effectiveness.

According to market studies [231], the PCB inspection field is in need of reliable systems in order to sustain growth as component densities get higher. Use of exhaustive solder paste inspection helps reduce the contribution from the print process to solder joint defects, in-turn saving money by reducing the cost of scrap with minimal cost to rework (i.e. wash boards) and with no penalty in solder joint reliability [232]. Some companies claim this number to be as high as 80% of their overall defect Pareto chart [233]. Furthermore, the total misclassification cost in an automated optical inspection system is the product of the production volume, cost-per-defective PCB and accuracy. Taking into account the ranges of the first two variables it is evident that even a minor, yet consistent, improvement in classification accuracy is translated to amplified profits.

Overall, classifier fusion can contribute to the visual solder-joint inspection domain by improving accuracy and speed. One of the conditions under which fusion is favorable is the high diversity in features and primary classifier outputs. Evaluation of a number of diversity metrics indicated that using distinct representations (different feature sets) of leads, in most cases leads to a reduction in the correlation between the outputs of individual classifiers. This is attributed to the reduced correlation in the input vectors of distinct information content. Since this is a desirable feature in fusion, a further research is required to establish the effects of combining truly different input representations besides exploiting different attributes of the same primary source of information (as with the use of the same optical images to obtain the different features sets). Fusion at different levels (measurements, features, and outputs) can then be evaluated overall.

Chapter 9

Conclusions and Future Research

In this Ph.D. thesis, we have developed a variety of hybrid intelligent multi-classifier approaches for machine vision systems targeted towards ***multi-lead surface mounted devices (SMDs) post-placement quality inspection*** using classical statistical and soft computing pattern recognition techniques. The purpose of this final chapter is to sum up the achievements of the work described in this thesis, to discuss its limitations and to address the directions of our future research.

A novel framework to inspect the post placement quality of SMDs based on 2D images of the SMD under investigation, has been proposed in Chapter 5. The problem of estimating the quality measures has been reduced to that of computing the lead shifts along their trans-axial direction. The shift of each lead is estimated from area characteristics (geometric measurements), on the corresponding lead image, through a classification approach. The classifier is fed with the measurements of area characteristics of each lead and produces a classification of the lead shift to several classes. Subsequently, individual-lead classification results are used within our Bayesian estimation setup, to estimate the total component displacement and rotation with respect to its ideal position, (i.e., the central position of its pad area). Having these estimates, we can easily derive the three quality measures of interest for either the entire component or its individual leads. Notice that the composite component displacement is computed in this way under super-resolution scales. Thus, even if the classifier outputs discrete classes, the component displacement is computed with real accuracy. The developed algorithm has been tested with success on real PCB images, in order to assess its potential.

In Chapter 6, we have considered two approaches to overcome the computational complexity of classical machine vision quality inspection of SMDs on a PCB. The first employs associative memories to implement the reduced information content in image intensity levels. The idea is to compare the edge structure of a lead image with that of stored fundamental patterns. The second scheme compresses the data space by considering only a directional projection function of the data. A non-linear filter based on high order neural networks is used to encode the characteristics of each projection function. Both methodologies are tested on real industrial PCB images. The quality of inspection slightly deteriorates while the computational time is significantly reduced, when compared to classical (full information) visual inspection techniques.

In Chapter 7, we have tested four combination methods for soft fusion of the outputs of multiple classifiers designed on the basis of optical features. The aim is to improve the performance of primary classifiers used for individual lead-image classification in post-placement quality inspection of components. A Bayes classifier and two well-known neural network classifiers, i.e. the MLP and LVQ classifiers, have been employed as baseline classifiers in this work for the discrimination of lead shifts in the examined images. In the sequel, the fusion of classifiers is achieved based on four approaches. The first combination approach uses the majority voting principle. The second scheme performs fusion by using Bayes naïve combiner. The third combination scheme involves the computation of fuzzy integrals. In the last scheme, the outputs of multiple classifiers are combined using Dempster – Shafer theory of evidence. Comparing the classification results of the proposed combined classifiers, we can conclude that each and every combiner achives better performance than any individual classifier alone. In addition, it is verified that both the naïve Bayes and the Dempster – Shafer combiners achieve better overall performance, with the naïve Bayes reaching the best performance improvement of 3.98 %.

Finally, in Chapter 8, a machine vision system is investigated for quality inspection of SMDs on a PCB based on the fusion of two individual classifiers that have been developed in Chapter 6, each representing the given pattern by a distinct feature vector. We have further investigated on a theoretical framework to derive nontrainable combination rules based on distinct pattern representations. From our experimental results it has been shown that multiple classifiers based on distinct pattern representations can be used to improve the robustness of post placement quality inspection systems.

A future research can be addressed towards five directions:

- **Rough Set Theory** instead of proposed methods in this thesis can be used for feature extraction and reduction of dimensionality purposes [217-218]. The ability to handle imprecise and incosistent information has become one of the most important requirements for a feature extraction system. Many theories, techniques and algorithms have been developed to deal with the analysis of imprecise or inconsistent information. The most succesful ones are based on fuzzy set theory and the Dempster-Shafer theory of evidence. Rough Set Theory, which was introduced by Pawlak [219], is a new mathematical tool that can be employed to handle uncertainty and vagueness. It focuses on the discovery of patterns in inconsistent data and can be used as the basis to perform formal reasoning under uncertainty, machine learning and rule discovery.

- **Independent Component Analysis (ICA)** [220-221] can be also employed for feature extraction and dimensionality reduction purposes [222] in our future work. The ICA is a well-established statistical signal / data processing technique that aims at decomposing a set of multivariate signals into a base of statistically independent data (or vectors/streams) with the minimal loss of information content. ICA generalizes the technique of Principal Component Analysis (PCA) and, like PCA has proven a useful tool for finding structure in data [220]. The main two appreciated uses of ICA are the linear *blind source separation* and the *data representation and visualization* [223].
- **Fuzzy Lattice Neurocomputing (FLN) classifiers** [224 -225] can be used for classification purposes instead of the utilized primary classifiers in this thesis. Several FLN classifiers have been presented in the literature. The most popular among them is the σ-FLNMAP classifier [224-225], which forms a synergy of two σ-FLN modules, namely σ-FLN$_a$ and σ-FLN$_b$ module interconnected via the MAP field F^{ab}. More specifically, the σ-FLNMAP is a lattice domain extension of the fuzzy-ARTMAP neural network [226]. In addition, the σ-FLNMAP is a promising candidate for majority-voting classification [227]. The idea is to train an ensemble of σ-FLNMAP modules, namely *voters*, using a different permutation of the training data per module; finally a testing datum is classified to the category that receives the majority vote in the ensemble. Hence, the *Voting σ*-FLNMAP classifier emerges [228], which may be regarded as a panel of experts each inducing its own set of rules from the training data.
- **Classifier combination based on confidence transformation**. The conversion of classifier outputs to crisp class label or rank order simplifies combination but loses useful information, deteriorating the combination

performance. In essence, the classifier outputs should be transformed to uniform measures that have similar scales. Preferably, the transformed measures represent the degree of confidence of decision, like the class posterior probability or likelihood [229]. In the context of classifier combination, the transformation of classifier outputs into confidence measures has been addressed in [214], [229]. A confidence transformation method is the combination of a scaling function and a confidence type [214]. The scaling function shifts and re-scales the classifier output to a moderate range such that the outputs of different classifiers are comparable. The re-scaled output is transformed to confidence measure using an activation function corresponding to one confidence type (e.g. log-likelihood (linear), likelihood (exponential), sigmoid, or evidence) [214].

- An additional line of research that is being considered is the use of **Support Vector Machines** (SVM) as an alternative method of classifier fusion. A specific SVM kernel which operates on the combined classifiers’ feature space has been designed for this purpose, but is still in primary stages of development.

Appendix A

Specific Cases of Maximum-Likelihood Parameter Estimation

A.1 The Gaussian Case: Unknown μ

To see how maximum-likelihood methods apply to specific cases, suppose that the samples are drawn from a multivariate normal population with mean $\boldsymbol{\mu}$ and covariance matrix $\boldsymbol{\Sigma}$. For simplicity, consider first the case where only the mean is unknown [9], [11]. Under this condition, we consider a sample point $\mathbf{x}_k$ and find

$$p(\mathbf{x}_k \mid \boldsymbol{\mu}) = \frac{1}{(2\pi)^{N/2}\left|\boldsymbol{\Sigma}\right|^{1/2}} \exp\left[-\tfrac{1}{2}(\mathbf{x}_k - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu})\right]$$

For n available samples $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_n$ we have

$$L(\boldsymbol{\mu}) \equiv \ln\prod_{k=1}^{n} p(\mathbf{x}_k \mid \boldsymbol{\mu}) = -\frac{n}{2}\ln\left[(2\pi)^N\left|\boldsymbol{\Sigma}\right|\right] - \frac{1}{2}\sum_{k=1}^{n} (\mathbf{x}_k - \boldsymbol{\mu})^T \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}) \qquad \text{(A.1)}$$

Taking the gradient with respect to $\boldsymbol{\mu}$ we obtain

$$\frac{\partial L(\boldsymbol{\mu})}{\partial \boldsymbol{\mu}} \equiv \begin{bmatrix} \dfrac{\partial L}{\partial \mu_1} \\ \dfrac{\partial L}{\partial \mu_2} \\ \vdots \\ \dfrac{\partial L}{\partial \mu_N} \end{bmatrix} = \sum_{k=1}^{n} \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \boldsymbol{\mu}) \qquad \text{(A.2)}$$

We see from Eqn (4.26) of Chapter 4 and Eqn (A.2) that the maximum-likelihood estimate for $\boldsymbol{\mu}$ must satisfy

$$\sum_{k=1}^{n} \boldsymbol{\Sigma}^{-1}(\mathbf{x}_k - \hat{\boldsymbol{\mu}}_{ML}) = 0 \qquad \text{(A.3)}$$

Multiplying by Σ and rearranging, we obtain

$$\hat{\boldsymbol{\mu}}_{ML} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{x}_k \qquad \text{(A.4)}$$

That is, the ML estimate of the unknown population mean, for the Gaussian densities, is just the arithmetic average of the training samples – the *sample mean*.

A.2 The Gaussian Case: Unknown μ and Σ

In the more general (and more typical) multivariate normal case, neither the mean $\boldsymbol{\mu}$ nor the the covariance matrix $\boldsymbol{\Sigma}$ is known. Thus, these unknown parameters constitute the components of the parameter vector $\boldsymbol{\theta}$. Consider first the univariate case with $\theta_1 = \mu$ and $\theta_2 = \sigma^2$. Here the log-likelihood of a single point is

$$L(\boldsymbol{\theta}) \equiv \ln p(x_k \mid \boldsymbol{\theta}) = -\frac{1}{2}\ln 2\pi\theta_2 - \frac{1}{2\theta_2}(x_k - \theta_1)^2 \tag{A.5}$$

and its derivative is

$$\frac{\partial L(\boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \frac{\partial \ln p(x_k \mid \boldsymbol{\theta})}{\partial \boldsymbol{\theta}} = \begin{bmatrix} \dfrac{1}{\theta_2}(x_k - \theta_1) \\ -\dfrac{1}{2\theta_2} + \dfrac{(x_k - \theta_1)^2}{2\theta_2^2} \end{bmatrix} \tag{A.6}$$

Applying Eq. (4.35) to the full log-likelihood leads to the conditions

$$\sum_{k=1}^{n} \frac{1}{\hat{\theta}_2}\left(x_k - \hat{\theta}_1\right) = 0 \tag{A.7}$$

and

$$-\sum_{k=1}^{n} \frac{1}{\hat{\theta}_2} + \sum_{k=1}^{n} \frac{\left(x_k - \hat{\theta}_1\right)^2}{\hat{\theta}_2^2} = 0 \tag{A.8}$$

where $\hat{\theta}_1$ and $\hat{\theta}_2$ are the maximum-likelihood estimates for θ_1 and θ_2, respectively. By substituting $\hat{\mu} = \hat{\theta}_1$ and $\hat{\sigma}^2 = \hat{\theta}_2$ and doing a little rearranging, we obtain the following maximum-likelihood estimates for μ and σ^2:

$$\hat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k \tag{A.9}$$

and

$$\hat{\sigma}^2 = \frac{1}{n}\sum_{k=1}^{n} (x_k - \hat{\mu})^2 \tag{A.10}$$

While the analysis of the multivariate case is basically very similar, considerably more manipulations are involved. Just as we would predict, however, the result is that the maximum-likelihood estimates for $\boldsymbol{\mu}$ and $\boldsymbol{\Sigma}$ are given by

$$\hat{\boldsymbol{\mu}} = \frac{1}{n}\sum_{k=1}^{n}\mathbf{x}_k \tag{A.11}$$

and

$$\hat{\Sigma} = \frac{1}{n}\sum_{k=1}^{n}\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}\right)\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}\right)^T \tag{A.12}$$

Thus, once again we find that the maximum-likelihood estimate for the mean vector is the sample mean. The maximum likelihood estimate for the covariance matrix is the arithmetic average of the *n* matrices $\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}\right)\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}\right)^T$. Because the true covariance matrix is the expected value of the matrix $\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}\right)\left(\mathbf{x}_k - \hat{\boldsymbol{\mu}}\right)^T$, this is also a very satisfying result.

Appendix B

Image Feature Extraction Algorithms

B.1 Boundary-based features

Boundary-based features describe the boundary information. Once the objects are separated from the background by segmentation, their boundary coordinates can be used to extract external features, such as perimeter, curvature, signature, bending energy, Fourier descriptors, 2-D transform coefficient features (e.g. Fourier, Haar, Hadamard, or Wavelet transform), image codes (e.g. run code, chain codes), etc [22 – 27].

- **Perimeter**

The pixel distance around the circumference of the object. To accurately compute this, where a boundary pixel contacts its neighbor vertically or horizontally, the pixel distance is 1 unit. Where a pixel contacts a neighbor diagonally, the pixel distance is the square root of 2, or 1.414 units. The result is a measure of object boundary length.

- **Curvature**

The curvature scalar descriptor (also called *boundary straightness*) finds the ratio between the total number of boundary pixels (length) and the number of boundary pixels where the boundary direction changes significantly. The smaller the number of direction changes the straighter the boundary. The evaluation algorithm is based on the detection of angles between line segments positioned *b* boundary pixels from the evaluated boundary pixel in both directions. The angle need not be represented numerically; rather, relative position of line segments can be used as a property. The parameter *b* determines sensitivity to local changes of the boundary direction. Curvature computed from the chain code can be found in [22-27], and the tangential border representation is also suitable for curvature computation.

- **Signature**

The signature of a object (or region) may be obtained as a sequence of normal contour distances The normal contour distance is calculated for each boundary element as a function of the path length. For each border point *A*, the shortest distance to an opposite border point *B* is sought in a direction perpendicular to the border tangent at point *A*. For example the *polar radii signature* of an object shows the relationship between the distance from its centroid to points along its boundary as a function of angle. This is an orientation-invariant feature which allows an object to be compared with a standard prototype by cyclically shifting the signature of one with respect to the other in steps while checking for the best match [22-23]. For a circular object the graphical signature would simply be a horizontal line with the ordinate corresponding to the radius of the circle.

- **Bending energy**

The bending energy (BE) of a border (curve) may be understood as the energy necessary to bend a rod to the desired shape, and can be computed as a sum of squares of the border curvature *c*(*k*) over the border length *L*.

$$\mathrm{BE} = \frac{1}{L}\sum_{k=1}^{L} c^2(k)$$

Bending energy can easily be computed from Fourier descriptors using Parseval's theorem [22]. To represent the border, Freeman's chain code or its smoothed version may be used.

- **Chord distribution**

A line joining any two points of the region boundary is a chord, and the distribution of lengths and angles of all chords on a contour may be used for shape description. Let $b(x, y) = 1$ represent the contour points, and $b(x, y) = 0$ represent all other points. The chord distribution can be computed as

$$h(\Delta x, \Delta y) = \sum_i \sum_j b(i, j) b(i + \Delta x, j + \Delta y) dxdy$$

B.2 Region-based features

We can use boundary information to describe a region, and shape can be described from the region itself [22-27]. A large group of shape description techniques is represented by heuristic approaches, which yield acceptable results in description of simple shapes. Region area, rectangularity, elongation, direction, compactness, etc., are examples of these methods. Unfortunately, they cannot be used for region reconstruction and do not work for more complex shapes. Other procedures based on region decomposition into smaller and simpler sub-regions must be applied to describe more complicated regions, then sub-regions can be described separately using heuristic approaches. Objects are represented by a planar graph with nodes representing sub-regions resulting from region decomposition, and region shape is then described by the graph properties [22-27]. There are two general approaches to acquiring a graph of sub-regions: The first one is region thinning leading to the *region skeleton*, which can be described by a graph. The second option starts with the *region decomposition* into sub-regions, which are then represented by nodes, while arcs represent neighborhood relations of sub-regions. It is common to stipulate that sub-regions be convex.

B.2.1 Simple scalar region descriptors

A number of simple heuristic shape descriptors exist which relate to statistical feature description. These methods are basic and are used for description of sub-regions in complex regions, and may then be used to define graph node classification [22-27].

- **Area**

The simplest and most natural property of a region is its area. The area is computed as the total number of pixels inside, and including, the object boundary. The result is a measure of object size. The area measure usually does not include object hole areas. Assuming that labeling has identified regions, the *algorithm of calculated area in quadtrees* may be used. If the region is represented by the (anti-clockwise) Freeman chain code, *the algorithm of region area calculation* from Freeman 4-connectivity chain code representation provides the region.

- **Compactness**

Compactness is a ratio based on the area and perimeter measures of an object. The result is a measure of object roundness or compactness, given as a value between 0 and 1. The greater the ratio, the rounder the object. If the ratio is equal to 1, the object is a perfect cycle. As the ratio decreases from 1, the object departs from a circular form. Compactness is given by

$$Compactness = \frac{4\pi \times Area}{(Perimeter)^2}$$

- **Euler's number**

Euler's number ϑ (sometimes called genus or the Euler-Poincaré characteristic) describes a simple, topologically invariant property of the object. It is based on *S*, the number of contiguous parts of an object, and *N*, the number of holes in the object (an object can consist of more than one region, otherwise the number of contiguous parts is equal to one. Then

$$\vartheta = S - N$$

Special procedures to compute Euler's number can be found in [22].

- **Projections**

Horizontal and vertical region projections $p_h(i)$ and $p_v(j)$ are defined as

$$p_h(i) = \sum_j f(i,j) \qquad p_v(j) = \sum_i f(i,j)$$

Region description by projections is usually connected to binary image processing. Projections can serve as a basis for definition of related region descriptors; for example, the width (height) of a region with no holes is defined as the maximum value of the horizontal (vertical) projection of a binary image of the region.

- **Major Axis (or Principal Axis)**

The Major axis is the (x, y) endpoints of the longest line that can be drawn through the object. The major axis endpoints (x_1, y_1) and (x_2, y_2) are found by computing the pixel distance between every combination of border pixels in the object boundary and finding the pair with the maximum length.

- **Major Axis Length**

The Major Axis Length is the pixel distance length between the major axis endpoints. The result is a measure of object length.

$$Major\ Axis\ Length = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

where (x_1, y_1) and (x_2, y_2) are the major axis endpoints.

- **Major Axis Angle**

The Major Axis Angle is the angle between the major axis and the *x*-axis of the image. The angle can range from 0º to 360º. The result is a measure of object orientation.

$$Major\ Axis\ Angle = \tan^{-1}\left(\frac{(y_2 - y_1)}{(x_2 - x_1)}\right)$$

- **Minor Axis**

The Minor Axis is the (x, y) endpoints of the longest line that can be drawn through the object while maintaining perpendicularity with the major axis. The minor axis endpoints (x_1, y_1) and (x_2, y_2) are found by computing the pixel distance between the two border pixel endpoints.

- **Minor Axis Width**

The Minor Axis Width is the pixel distance length between the minor axis endpoints. The result is a measure of object width.

$$Minor\ Axis\, Width = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}$$

where (x_1, y_1) and (x_2, y_2) are the minor axis endpoints.

- **Aspect Ratio (or Eccentricity)**

The Aspect Ratio is the ratio of the width of the minor axis to the length of the major axis. This ratio is computed as the minor axis width distance divided by the major axis length distance. The result is a measure of object elongation, given as a value between 0 and 1. If the ratio is equal to 1, the object is roughly square or circularly shaped. As the ratio decreases from 1, the object becomes more elongated.

$$Aspect\ Ratio = \frac{Minor\ Axis\,Width}{Major\ Axis\ Length}$$

- **Elongation (Bounding Box Area)**

Elongation is a ratio between the length and width of the region-bounding rectangle (*bounding box area*). This is the rectangle of minimum area that bounds the shape, which is located by turning in discrete steps until a minimum is located. This criterion cannot succeed in curved regions, for which the evaluation of elongation must be based on maximum region thickness. Elongation can be evaluated as a ratio of the region area and the square of its thickness. The maximum region thickness (holes must be filled if present) can be determined as the number of erosion steps [22] that be applied before the region totally disappears. If the number of erosion steps is *d*, elongation is then

$$Elongation = \frac{Area}{(2d)^2}$$

- **Rectangularity**

Let F_k be the ratio of region area and the area of a bounding rectangle, the rectangle having the direction *k*. The rectangle direction is turned in discrete steps, and rectangularity measured as a maximum of this ratio F_k :

$$\mathrm{Re}\,c\,\tan gularity = \underset{k}{ma}\,x(F_k)$$

The direction need only be turned through one quadrant. Rectangularity assumes values from the interval (0,1], with 1 representing a perfectly rectangular region. Sometimes, it may be more natural to draw a bounding triangle; a method for similarity evaluation between two triangles called *sphericity* [22-23].

- **Direction**

Direction is a property, which makes sense in elongated regions only. If the region is elongated, *direction* is the direction of the longer side of a minimum bounding rectangle. If the shape moments are known, the direction θ can be computed as

$$\theta = \frac{1}{2}\tan^{-1}\left(\frac{2\mu_{11}}{\mu_{20} - \mu_{02}}\right)$$

It should be noted that elongation and rectangularity are independent of linear transformations, translation, rotation, and scaling. Direction is independent on all linear transformations, which do not include rotation. Mutual direction of two rotating objects is rotation invariant.

B.2.2 Moments

Region moment representations interpret a normalized gray-level image function as a probability density of a 2D random variable [22-23]. Properties of this random variable can be described using statistical characteristics – *moments*. Assuming that non-zero pixel values represent regions, moments can be used for binary or gray-level region description. A moment of order (*p*+*q*) is depended on scaling, translation, rotation, and even on gray-level transformations and given by

$$m_{pq} = \sum_i \sum_j i^p j^q f(i, j)$$

where i, j are the pixel coordinates.

Translation invariance can be achieved if we use the central moments,

$$\mu_{pq} = \sum_i \sum_j (i - x_c)^p (j - y_c)^q f(i, j)$$

where x_c, y_c are the coordinates of the region’s center of gravity (*centroid*), which can be obtained using the following relationships:

$$x_c = \frac{m_{10}}{m_{00}}$$

$$y_c = \frac{m_{01}}{m_{00}}$$

In the binary case, m_{00} represents the region area.

B.2.3 Convex hull

A region *R* is convex if and only if for any two points $x_1, x_2 \in R$, the whole line segment $x_1 x_2$ defined by its end points x_1, x_2 is inside the region *R*. The convex hull of a region is the smallest convex region *H* that satisfies the condition $R \subseteq H$. The convex hull has some special properties in digital data, which do not exist in the continuous case. For instance, concave parts can appear and disappear in digital data due to rotation, and therefore the convex hull is not rotation invariant in digital space. The convex hull can be used to describe region shape properties and can be used to build a tree structure of region concavity.

The region convex hull construction algorithm [170] can define a discrete convex hull. This algorithm has complexity $\mathrm{O}(n^2)$. The *simple polygon convex hull detection* algorithm [170] describes a more efficient approach.

Appendix C

Otsu's Thresholding Method

In this Appendix, we review the Otsu method for selecting optimal image threshold [152], [169]. For simplicity, we consider a single thresholding problem here. This corresponds to the most frequent requirement in image thresholding, to separate an image into two classes, foreground and background. Generalization of the formulation to multilevel thresholding problems is discussed in Subsection 4.2.6.2 of Chapter 4.

An image can be represented by a 2D gray-level intensity function $I(x,y)$. The value z of $I(x,y)$ is the gray-level (or the ***pixel intensity value***), ranging from 0 to $L-1$, where L is the number of distinct gray-levels.

In the case of single thresholding, the pixels of an image are divided into two classes $C_1 = \{0,1,...,T\}$ and $C_2 = \{T+1,\ T+2,...,\ L-1\}$, where T is the ***threshold value***. C_1 and C_2 are normally corresponding to the foreground (objects of interest) and the background.

We make use of the following notations:

- $P_1 = \Pr\{C_1\} =$ a priori probability of class C_1
- $P_2 = \Pr\{C_2\} =$ a priori probability of class C_2
- $p_1(z) = \Pr\{z \mid C_1\} =$ probability density function of gray-level z in C_1
- $p_2(z) = \Pr\{z \mid C_2\} =$ probability density function of gray-level z in C_2
- $h(z)$ is the normalized histogram function which represents the percentage of pixels having gray-level z over the total number of pixels of the image.
- the probability density function of all pixel gray levels in the image, corresponding to the normalized pixel intensity histogram, is $p(z) = P_1 p_1(z) + P_2 p_2(z)$.

Otsu has developed a thresholding method based on discriminant analysis which maximizes some measures of class separability [152], [169]. One of the measures is

$$J_{OT_1}(T)=\frac{P_1(T)P_2(T)\left[m_1(T)-m_2(T)\right]^2}{P_1(T)\sigma_1^2(T)+P_2(T)\sigma_2^2(T)} \tag{C.1}$$

where

$$P_1(T)=\Pr\{C_1\}=\sum_{z=0}^{T}h(z) \tag{C.2}$$

$$P_2(T)=\Pr\{C_2\}=\sum_{z=T+1}^{L-1}h(z)=1-P_1 \tag{C.3}$$

$$m_1(T)=\sum_{z=0}^{T}z\Pr\{z\,|\,C_1\}=\frac{1}{P_1}\sum_{z=0}^{T}zh(z) \tag{C.4}$$

$$m_2(T)=\sum_{z=T+1}^{L-1}z\Pr\{z\,|\,C_2\}=\frac{1}{P_2}\sum_{z=T+1}^{L-1}zh(z) \tag{C.5}$$

$$\sigma_1^2(T)=\sum_{z=0}^{T}\left[z-m_1(T)\right]^2\Pr\{z\,|\,C_1\}=\frac{1}{P_1}\sum_{z=0}^{T}\left[z-m_1(T)\right]^2h(z) \tag{C.6}$$

$$\sigma_2^2(T)=\sum_{z=T+1}^{L-1}\left[z-m_2(T)\right]^2\Pr\{z\,|\,C_2\}=\frac{1}{P_2}\sum_{z=T+1}^{L-1}\left[z-m_2(T)\right]^2h(z) \tag{C.7}$$

In the above equations, C_1 and C_2 are dependent of T and contain pixels with gray values in $[0,T]$ and $[T+1,L-1]$ respectively.

To maximize the ***criterion function*** in Eq. (C.1), the means of the two classes should be as well separated as possible and the variances in both classes should be as small as possible. This is similar to the ***Fisher criterion*** for pattern classification [9].

The optimal threshold value T_{OT}^* can be determined by searching for the value in the range $[0,L-1]$ so that $J_{OT}(T)$ is the maximum. That is,

$$T_{OT}^*=\arg\max_{0\le T\le L-1}J_{OT_1}(T) \tag{C.8}$$

Otsu has pointed out that the criterion function J_{OT_1} is equivalent to the following alternative criterion function [152]

$$J_{OT_2}(T)=\frac{\sigma^2}{P_1(T)\sigma_1^2(T)+P_2(T)\sigma_2^2} \tag{C.9}$$

where $$\sigma^2=\sum_{z=0}^{L-1}\left[z-m(T)\right]^2h(z) \tag{C.10}$$

where $$m = \sum_{z=0}^{L-1} zh(z) = P_1 m_1(T) + P_2 m_2(T) \qquad \text{(C.11)}$$

In this Ph.D. thesis, for Otsu's method, we make use of the following criterion function

$$J_{OT}(T) = \frac{P_1(T)\sigma_1^2(T) + P_2(T)\sigma_2^2(T)}{\sigma^2} \qquad \text{(C.12)}$$

This is simply the inverse of $J_{OT_2}(T)$. Now the optimal threshold is the gray – level at which the criterion function $J_{OT}(T)$ is minimum, similar to the other two criterion functions (C.1) and (C.9). That is,

$$T_{OT}^* = \arg \min_{0 \le T \le L-1} J_{OT}(T) \qquad \text{(C.13)}$$

In order to generalize to multilevel thresholding problems (i.e. four-level Othu's algorithm), the aforementioned Otsu's method can be formulated using the following generalized equation:

$$J(T) = \sum_{z=0}^{L-1} h(z) c(z,T) = \sum_{z=0}^{T} h(z) c_1(z,T) + \sum_{z=T+1}^{L-1} h(z) c_2(z,T) \qquad \text{(C.14)}$$

where $c(z,T)$ can be considered as the cost to pixels with gray level z when the threshold is set at value *T*. The cost function is split into two parts, $c_1(z,T)$ and $c_2(z,T)$, which provide different weights for pixels in two classes.

Based on the definitions of $P_1(T), P_2(T), \sigma_1(T), \sigma_2(T)$ in Eqs (C.2) to (C.7) and the definition of σ in Eqs (C.10) to (C.11), we can rewrite the modified criterion function as

$$J_{OT}(T) = \sum_{z=0}^{T} h(z) \frac{\left[z - m_1(T)\right]^2}{\sigma^2} + \sum_{z=T+1}^{L-1} h(z) \frac{\left[z - m_2(T)\right]^2}{\sigma^2} \qquad \text{(C.15)}$$

Thus, the cost functions are

$$c_{OT}^{(1)}(z,T) = \frac{\left[z - m_1(T)\right]^2}{\sigma^2} \quad \text{if } z \le T \qquad \text{(C.16)}$$

and

$$c_{OT}^{(2)}(z,T) = \frac{\left[z - m_2(T)\right]^2}{\sigma^2} \quad \text{if } z > T \qquad \text{(C.17)}$$

Appendix D

Proofs for Bayesian Displacement Estimator

D1. Approximation of conditional integral

From the central value theorem we conclude that for the continuous variable x in the interval $[a_i, b_i]$, there exists a certain value $x_0 \in [\alpha_i, b_i]$ that describes the integral in the form:

$$\int_{a_i}^{b_i} e^{-\frac{(x-s)^2}{2\sigma^2}} dx = (b_i - a_i) e^{-\frac{(x_0-s)^2}{2\sigma^2}} \qquad \text{(D.1)}$$

Our task is to find, or approximate, this value x_0. Eqn (D.1), is re-written:

$$\int_{a_i}^{x_0} \left[e^{-\frac{(x-s)^2}{2\sigma^2}} - e^{-\frac{(x_0-s)^2}{2\sigma^2}} \right] dx = \int_{x_0}^{b_i} \left[e^{-\frac{(x_0-s)^2}{2\sigma^2}} - e^{-\frac{(x-s)^2}{2\sigma^2}} \right] dx \qquad \text{(D.2)}$$

In this form, the left hand side represents the area A at Figure D.1, while the right hand side describes the area B. Assuming linearity of the exponent function in the interval $[a_i, x_0]$ and $[x_0, b_i]$, the area A is equal to that of A' in Figure D.1. The same holds for the areas B, B'. Thus (D.2) can be re-written as:

$$\int_{a_i}^{x_0} \left[e^{-\frac{(a_i-s)^2}{2\sigma^2}} - e^{-\frac{(x-s)^2}{2\sigma^2}} \right] dx = \int_{x_0}^{b_i} \left[e^{-\frac{(x-s)^2}{2\sigma^2}} - e^{-\frac{(b_i-s)^2}{2\sigma^2}} \right] dx \qquad \text{(D.3)}$$

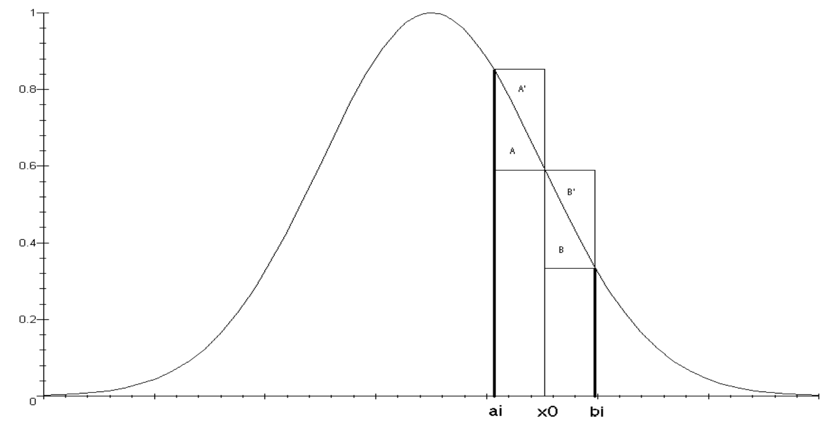


Figure D.1 Approximation of $\int_{a_i}^{b_i} e^{-\frac{(x-s)^2}{2\sigma^2}} dx$.

where now the left hand side represents the area A' and the right hand side the area B'. Hence,

$$\int_{a_i}^{b_i} e^{-\frac{(x-s)^2}{2\sigma^2}} dx = (x_0 - a_i) e^{-\frac{(a_i-s)^2}{2\sigma^2}} + (b_i - x_0) e^{-\frac{(b_i-s)^2}{2\sigma^2}} \tag{D.4}$$

Approximating the integral of the left hand side with the trapezoid rule we obtain:

$$\frac{b_i - a_i}{2}\left[e^{-\frac{(b_i-s)^2}{2\sigma^2}} + e^{-\frac{(a_i-s)^2}{2\sigma^2}} \right] = (x_0 - a_i) e^{-\frac{(a_i-s)^2}{2\sigma^2}} + (b_i - x_0) e^{-\frac{(b_i-s)^2}{2\sigma^2}}$$

or

$$\left(x_0 - \frac{a_i + b_i}{2} \right) e^{-\frac{(a_i-s)^2}{2\sigma^2}} = \left(x_0 - \frac{a_i + b_i}{2} \right) e^{-\frac{(b_i-s)^2}{2\sigma^2}} \tag{D.5}$$

Thus, approximately:

$$x_0 = \frac{a_i + b_i}{2} = l_i \tag{D.6}$$

since the quantized measurement assumes the central value of its interval.

D2. Statistics & Bounds of the Displacement Estimator

D2.1 Statistics of Quantized Classification *y*

Considering the case that $\alpha_i^{k,r} = P\left(y_k^r / \omega_i\right)\delta\left(i-r\right)$, with $\delta\left(i-r\right)$ a δ-function we obtain the estimator $\hat{s} = \frac{1}{K}\sum_{k=1}^{K} y_k^r$. The mean and variance of the estimator are therefore directly related with those of the quantized variable *y*. Let us consider the quantization process and its effect on the Gaussian distribution of *x*. For the mean of *y* for specific (fixed) s we have

$$\begin{aligned} E\left\{y\right\} &= \frac{1}{\sqrt{2\pi}\sigma}\sum_{r=-M}^{M} l_r \int_{\alpha_r}^{b_r} e^{-\left(x-s\right)^2/2\sigma^2} dx \\ &= \frac{1}{\sqrt{2\pi}\sigma}\sum_{r=-M}^{M} l_r \int_{-T/2}^{T/2} e^{-\left(l_r - s + \delta\right)^2/2\sigma^2} d\delta \end{aligned} \qquad \text{(D.7)}$$

where $\delta \in \left[-\frac{T}{2}, \frac{T}{2}\right]$.

Let now l_0 indicate the level of quantization that includes the mean s into its interval. We can interprete the mean in terms of its level as

$$E\left\{Y\right\} = \frac{T}{\sqrt{2\pi}\sigma}\sum_{n=-N_1}^{N_2}\left(l_0 + nT\right)\int_{-T/2}^{T/2} e^{-\left(l_0 - s + \delta - nT\right)^2/2\sigma^2}\frac{1}{T}d\delta \qquad \text{(D.8)}$$

Using the central value theorem to approximate the integral in (D.8), we obtain similar to Eqn (D.1):

$$E\left\{Y\right\} = \frac{T}{\sqrt{2\pi}\sigma}\sum_{n=-N_1}^{N_2}\left(l_0 + nT\right)e^{-\left(l_0 - s - nT\right)^2/2\sigma^2} \qquad \text{(D.9)}$$

at $\delta = 0$. Instead, we can preserve the form of (D.8) that can be interpreted as the mean value over a random variable δ, uniformly distributed. The mean value of *y* depends on the exact position of s within the interval $\left[l_0 - \frac{T}{2},\, l_0 + \frac{T}{2}\right]$. For the ensemble of the problem it is reasonable to assume that the actual displacement s will be located anywhere in this interval with equal probability. Notice that this assumption does not model a distribution of s in its entire range of definition, as in a map consideration. It rather models the error $l_0 - s$ in the interval $\left[-\frac{T}{2}, \frac{T}{2}\right]$ as to derive an approximation of the mean of the quantized estimator y irrespective of the position of s. Figure D.2 clarifies the aforementioned statements.

Consider $l_0 - s = \delta$, thus absorbing the uniformly distributed random variable into the displacement between the mean (s) and the midpoint (l_0) of the quantization interval that includes this mean. In this form:

$$\begin{aligned} E\{y\} &= \frac{T}{\sqrt{2\pi}\sigma} \sum_{n=-N_1}^{N_2} \underset{\delta}{E}\left\{\left(l_0 + nT\right) e^{-(\delta+nT)^2/2\sigma^2}\right\} \\ &= \frac{T}{\sqrt{2\pi}\sigma} \sum_{n=-N_1}^{N_2} \underset{\delta}{E}\left\{\left(s + \delta + nT\right) e^{-(\delta+nT)^2/2\sigma^2}\right\} \\ &= \frac{T}{\sqrt{2\pi}\sigma} \sum_{n=-N_1}^{N_2} \int_{-T/2}^{T/2} \left(s + \delta + nT\right) e^{-(\delta+nT)^2/2\sigma^2} \frac{1}{T} d\delta \\ &= \frac{1}{\sqrt{2\pi}\sigma} \int_{-\infty}^{+\infty} \left(s + z\right) e^{-z^2/2\sigma^2} dz = s \end{aligned} \tag{D.10}$$

with $z = \delta + nT$ in the entire range of definition of the Gausian distribution.

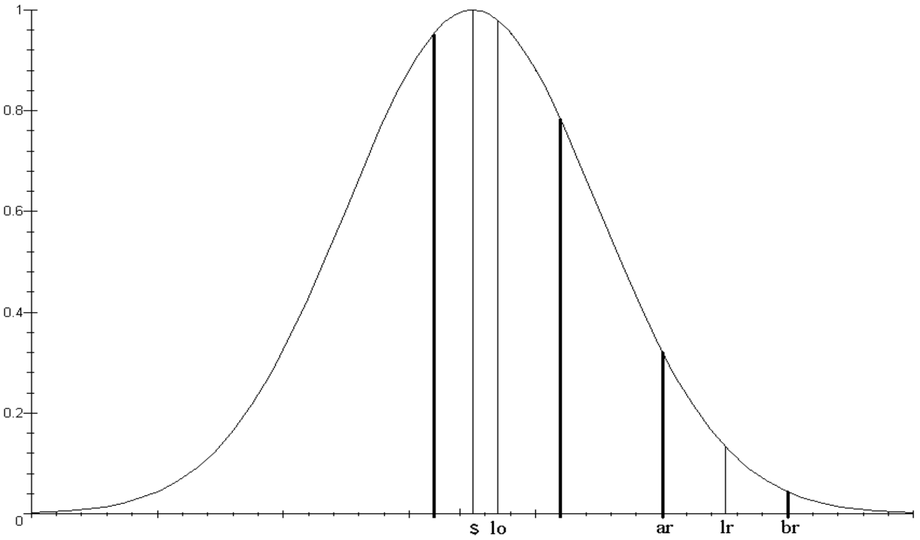


Figure D.2 The quantization process and its effect on the Gaussian distribution.

Similarly, for the variance of y we have:

$$\begin{aligned}\sigma_y^2 &= E\left\{(y-s)^2\right\} = \frac{1}{\sqrt{2\pi}\sigma}\sum_{r=-M}^{M}(l_r-s)^2\int_{\alpha_r}^{b_r} e^{-(x-s)^2/2\sigma^2}dx \\ &= \frac{T}{\sqrt{2\pi}\sigma}\sum_{r=-M}^{M}(l_r-s)^2\int_{-T/2}^{T/2} e^{-(l_r-s+\delta)^2/2\sigma^2}\frac{1}{T}dx \\ &\cong \frac{T}{\sqrt{2\pi}\sigma}\sum_{n=-N_1}^{N_2} \underset{\delta}{E}\left\{(\delta+nT)^2 e^{-(\delta+nT)^2/2\sigma^2}\right\} \\ &= \frac{1}{\sqrt{2\pi}\sigma}\int_{-\infty}^{+\infty} z^2 e^{-z^2/2\sigma^2}dz = \sigma^2\end{aligned} \tag{D.11}$$

D2.2 Statistics of the Estimator $\hat{s}$

The maximum likelihood estimator is given in the form :

$$\hat{s} = \frac{\sum_k\sum_i \alpha_i^{k,r} l_i}{\sum_k\sum_i \alpha_i^{k,r}}, \qquad \alpha_i^{k,r} = P\left(y_k^r/\omega_i\right) = P\left(l_r/\omega_i\right) \tag{D.12}$$

The form of this estimator is related more to the quantization levels l_r (classes), than to the observations y_k^r themselves. In other words, it is a function of the classes depending on the outcome of each observation. Therefore, to compute the statistics of this estimator we focus on the distribution of classes within the observations, rather than on the statistics of the observations themselves. This consideration shares a lot of similarities with the Monte Carlo approximation of a parameter's distribution. Hence, for large number of observations K we obtain:

$$\hat{s} \;=\; \frac{\sum_k \frac{1}{\mathrm{K}}\sum_i \alpha_i^{k,r} l_i}{\sum_k \frac{1}{\mathrm{K}}\sum_i \alpha_i^{k,r}} \;=\; \frac{\sum_{r=-M}^{M} P(\omega_r)\sum_{i=-M}^{M} P(l_r/\omega_i) l_i}{\sum_{r=-M}^{M} P(\omega_r)\sum_{i=-M}^{M} P(l_r/\omega_i)} \tag{D.13}$$

To simplify the computation, we assume that the classification error $P\left(l_r/\omega_i\right)$ is similarly distributed for all classes ω_r or levels l_r and the error for each class is symmetrically distributed around its own level. In other words, $P(l_r / \omega_i) = P(|r-i|)$ or $P(l_{i\pm n} / \omega_i) = P(l_n / \omega_0)$. Under these conditions

$$\sum_{r=-M}^{M} P(\omega_r) \sum_{i=-M}^{M} P(l_r / \omega_i) = \left(\sum_{i=-M}^{M} P(l_r / \omega_i) \right) \left(\sum_{r=-M}^{M} P(\omega_r) \right) = \sum_{r=-M}^{M} P(l_r / \omega_0) = 1$$

At the boarders, where $|i|$ or $|r|$ are large, we assume $P(\omega_r) \to 0$, in order to avoid symmetry problems. Consequently,

$$\begin{aligned}
\hat{s} &= \sum_{r=-M}^{M} P(\omega_r) \sum_{i=-M}^{M} P(l_r / \omega_i) l_i \\
&= \sum_{r=-M}^{M} P(\omega_r) \left[P(l_r / \omega_r) l_r + \sum_{n=1}^{N} (l_{r-n} + l_{r+n}) P(l_n / \omega_0) \right] \qquad \text{(D.14)} \\
&= P(l_0 / \omega_0) \sum_{r=-M}^{M} 2 l_r \frac{T}{\sqrt{2\pi}\sigma} e^{-(l_r - s)^2 / 2\sigma^2} + \sum_{n=1}^{N} P(l_n / \omega_0) \sum_{r=-M}^{M} 2 l_r \frac{T}{\sqrt{2\pi}\sigma} e^{-(l_r - s)^2 / 2\sigma^2}
\end{aligned}$$

and

$$E\{\hat{s}\} = s \left[P(l_0 / \omega_0) + 2 \sum_{n=1}^{N} P(l_n / \omega_0) \right] = s \sum_{r=-M}^{M} P(l_r / \omega_0) = s \qquad \text{(D.15)}$$

thus revealing an unbiased estimator. To be more precise, as in Section D2.1, the maximum likelihood estimator $\hat{s}$ for fixed s is almost unbiased; its small bias is a function of $|s - l_0|$. For $l_0 - s$ uniformly distributed in $\left[-\frac{T}{2}, \frac{T}{2} \right]$, the estimator $\hat{s}$ is unbiased.

Consider now the variance of the estimator for large number of observations. In general:

$$(\hat{s} - s)^2 = \frac{1}{\left[\sum_{i=-M}^{M} P(l_r / \omega_i) \right]^2} \left[\sum_{r=-M}^{M} P(\omega_r) \sum_{i=-M}^{M} (l_i - s) P(l_r / \omega_i) \right]^2 \qquad \text{(D.16)}$$

or

$$
\begin{aligned}
(\hat{s}-s)^2\left[\sum_{i=-M}^{M}P(l_r/\omega_i)\right]^2 &= \left[\sum_{i=-M}^{M}\sum_{r=-M}^{M}P(l_r/\omega_i)(l_i-s)P(\omega_r)\right]^2 \\
&=\left[\sum_{r=-M}^{M}P(l_r/\omega_i)(l_r-s)P(\omega_r)+\sum_{r=-M}^{M}P(\omega_r)\sum_{n=1}^{N}(l_{r-n}+l_{r+n}-2s)P(l_n/\omega_0)\right]^2 \\
&=\left[P(l_0/\omega_0)\sum_{r=-M}^{M}(l_r-s)P(\omega_r)+2\sum_{i=1}^{N}P(l_i/\omega_0)\sum_{r=-M}^{M}(l_r-s)P(\omega_r)\right]^2 \\
&=\left[\sum_{r=-M}^{M}(l_r-s)P(\omega_r)\right]^2\left[\sum_{i=-M}^{M}P(l_r/\omega_i)\right]^2
\end{aligned}
\qquad \text{(D.17)}
$$

Thus,

$$E\left\{(\hat{s}-s)^2\right\}=E\left\{\left[\sum_{r=-M}^{M}(l_r-s)P(\omega_r)\right]^2\right\} \qquad \text{(D.18)}$$

This is the same as the variance of the estimator when $\alpha_i^{k,r}=P(l_r/\omega_i)=\delta(r-i)$, which is equal to $\frac{1}{\mathrm{K}}\sigma^2$. In that case it is easily verified that $\hat{y}=\frac{1}{\mathrm{K}}\sum_k y_k^r$ with $\sigma_{\hat{y}}^2=\frac{1}{\mathrm{K}}\sigma^2$ and $(\hat{y}-s)^2=\left(\sum_{r=-M}^{M}P(\omega_r)(l_r-s)\right)^2$ with

$$E\left\{(\hat{y}-s)^2\right\}=E\left\{\left(\sum_{r=-M}^{M}P(\omega_r)(l_r-s)\right)^2\right\}=\frac{1}{\mathrm{K}}\sigma^2 \qquad \text{(D.19)}$$

From (D.18) and (D.19) we conclude $E\left\{(\hat{s}-s)^2\right\}=\frac{1}{\mathrm{K}}\sigma^2$, indicating a standard error $\frac{1}{\mathrm{K}}\sigma^2$ that is reducing with the number of observations considered.

References

[1] D. Kulkarni, *Computer Vision and Fuzzy-Neural Systems*, Prentice Hall, New Jersey,USA, 2001.

[2] E. N. Malamas, E. G. M. Petrakis, M.E. Zervakis, L. Petit, J-D Legat "A Survey on Industrial Vision Systems, Applications and Tools", *Image and Vision Computing*, vol. 21, no. 2, pp. 171-188, 2003.

[3] B. Batcelor and F. Waltz, *Intelligent Machine Vision-Techniques, Implementations and Applications*, Springer-Verlag, London, 2001.

[4] S. L. Bartlet, P. J. Besl, C. L. Cole, R. Jain, D. Mukherjee, K. D. Skifstad, "Automatic Solder Joint Inspection", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 10, no. 1, pp. 31-41, 1988.

[5] H-H Loh and M-S Lu, "Printed Circuit Board Inspection using Image Analysis", *IEEE Trans. on Industry Applications*, vol. 35, no. 2, pp. 426-432, 1999.

[6] S. P. Chan, "Robotic Assemply in Electronics Manufacturing Systems", in Industrial and Manufacturing Systems, vol. 4 of Neural Network Systems Techniques and Applications, Academic Press, USA, 1998.

[7] J. Tou and R. Gonzalez, Pattern Recognition Principles, Addison Wesley Publishing Company, USA, 1974.

[8] R. Schalkoff, Pattern Recognition: Statistical, Structural, and Neural Approaches , Wiley, USA, 1992.

[9] R. O. Duda, P. E. Hart, D. G. Stork, Pattern Classification, 2nd Edition, John Wiley & Sons, USA, 2001.

[10] C. Bishop, Neural Networks for Pattern Recognition, Oxford Univercity Press, New York, 1995.

[11] S. Theodoridis, and K. Koutroumbas, *Pattern Recognition*, third edition, Academic Press, USA, 2006.

[12] F. van der Heijden, *Image Based Measurement Systems – Object Recognition and Parameter Estimation*, Wiley, USA, 1994.

[13] F. van der Heijden, R.P.W. Duin, D. de Ridder, D.M.J. Tax, *Classification, Parameter Estimation, and State Estimation – An Engineering Approach using Matlab*, Wiley, UK, 2004.

[14] A. Ghosh, S.K.Pal, *Soft Computing Approach to Pattern Recognition and Image Processing*, vol. 53 of Series in Machine Perception and Artificial Intelligence, World Scientific, USA, 2002.

[15] P. Melin, O. Castillo, *Hybrid Intelligent Systems for Pattern Recognition using Soft Computing*, Springer Verlag, Germany, 2005.

[16] H. Bunke, and A. Kandel, *Hybrid Methods in Pattern Recognition*, vol. 47 of Series in Machine Perception and Artificial Intelligence, World Scientific, USA, 2002.

[17] J.-S. R. Jang, C.-T. Sun, E. Mizutani, Neuro-Fuzzy and Soft Computing, Prentice Hall, USA, 1997.

[18] D. Nauck, F. Klawonn, R. Kruse, Foundations of Neuro-Fuzzy Systems, Wiley, USA, 1997.

[19] M. Zervakis, S.K. Goumas, and G. A. Rovithakis, "A Bayesian Framework for Multi-lead SMD Post-Placement Quality Inspection", ***IEEE Transactions on Systems, Man and Cybernetics*** **-Part B: Cybernetics, vol. 34, no. 1, pp. 440-453, February 2004.**

[20] J. Kittler, “Multiple Classifier Systems” , in Soft Computing Approach to Pattern Recognition and Image Processing, vol. 53, Chapter 1 of Series in Machine Perception and Artificial Intelligence, World Scientific, USA, 2002.

[21] L.I.Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley-Interscience publication, New Jersey, USA, 2004.

[22] M. Sonka, V. Hlavac, R. Boyle, *Image Processing, Analysis, and Machine Vision,* Brooks/Cole Publishing Company, USA, 1999.

[23] J. C. Russ, *The Image Processing Handbook,* 3rd Edition, CRC Press, NY, 2000.

[24] J. R. Parker, *Algorithms for Image processing and Computer Vision,* Wiley, USA, 1997.

[25] B. Jahne, H. Haubecker, *Computer Vision and Applications,* Academic Press, USA, 2000.

[26] R. Jain, R. Kasturi, B. G. Schunck, *Machine Vision,* McGraw-Hill International Editions, Singapore, 1995.

[27] M. Seul, L. O’ Gorman, M. J. Sammon, *Practical Algorithms for Image Analysis - Description, Examples and Code,* Cambridge University Press, USA, 2000.

[28] S. K. Goumas, M.E. Zervakis, and G. A. Rovithakis, “Reduced Dimensionality Space for Post Placement Quality Inspection of Components based on Neural Networks”, in *Proc. ESSAN’ 2004, 12th European Symposium on Artificial Neural Networks*, pp. 275-280, April 28-30, 2004, Bruges, Belgium.

[29] J. M. Winograd, J. Ludwig, H. Nawab, et al, “Approximate Processing and Incremental Refinement Concepts”, in *Proc. IEEE 2nd Annual RASSP – Rapid Prototyping of Application Specific Signal processors Conf.*, Washington, D.C., July 1995.

[30] S. K. Goumas, G. A. Rovithakis and M. E. Zervakis, “A Bayesian Image Analysis Framework for Post Placement Quality Inspection of Components”, *Proceedings of 2002 IEEE International Conference on Image Processing - ICIP 2002*, pp. II-549-552, September 22-25, 2002, Rochester, New York, USA.

[31] R. A. Johnson, D. W. Wichern, *Applied Multivariate Statistical Analysis*, Prentice Hall, USA, 1998.

[32] A. S. Pandya, R. B. Macy, *Pattern Recognition with Neural Networks in C++*, CRC Press, USA, 1996.

[33] K. Fukunaga, *Introduction to Statistical Pattern Recognition*, Second Edition, Academic Press, USA, 1990.

[34] J. Schurmann, *Pattern Classification, a unified view of statistical and neural approaches*, Wiley, USA, 1996.

[35] R. P.W. Duin, F. Roli, D. de Ridder, “A Note on Core Research Issues for Statistical Pattern Recognition”, *Pattern Recognition Letters* 23 (2002) pp. 493-499.

[36] T. J. Ross, *Fuzzy Logic with Engineering Applications*, McGraw-Hill, USA, 1995.

[37] G. J. Klir, B. Yuan, *Fuzzy Sets and Fuzzy Logic, Theory and Applications*, Prentice-Hall, USA, 1995.

[38] W. Pedrycz, *Computational Intelligence – An Introduction*, CRC Press, USA, 1998.

[39] G. J. Klir, T. A. Folger, *Fuzzy Sets, Uncertainty, and Information*, Prentice-Hall, USA, 1988.

[40] G. J. Klir, *Uncertainty, and Information – Foundations of Generalized Information Theory*, Wiley-Interscience, USA, 2006.

[41] C.T. Lin and C.S. George Lee, *Neural fuzzy systems: a neuro-fuzzy synergism to intelligent systems*, Prentice-Hall Inc, NJ, USA 1996.

[42] S. Haykin, Neural Networks – A Comprehensive Foundation, Prentice-Hall, USA, 1994.

[43] C. G. Looney, *Pattern Recognition using Neural Networks*, Oxford University Press, New York, 1997.

[44] K. F. Man, K.S. Tang and S. Kwong, *Genetic Algorithms,* Springer-Verlag, Great Britain, 1999.

[45] Z. Michalewicz, *Genetic Algorithms + Data Structures = Evolution Programs*, Third, Revised and Extended Edition, Springer-Verlag, USA, 1999.

[46] S. K. Pal, P. P. Wang, *Genetic Algorithms for Pattern Recognition*, CRC Press, USA, 1996.

[47] K. F. Man, K.S. Tang and S. Kwong, “Genetic Algorithms: Concepts and Applications”, *IEEE Transactions on Industrial Electronics*, Vol. 43, No. 5, pp. 519-534, 1996.

[48] S. K. Goumas, I. N. Dimou, M. E. Zervakis, “Combination of Multiple Classifiers for Post Placement Quality Inspection of Components: A Comparative Study”, *Information Fusion*, Accepted for publication in the March 1, 2008.

[49] S. K. Goumas, M. E. Zervakis, A. Pouliezos, and G.S. Stavrakakis, “Intelligent On-Line Quality Control of Washing Machines using Discrete Wavelet Analysis Features and Likelihood Classification” *International Scientific Journal Engineering Applications of Artificial Intelligence, Elsevier Science Ltd.* Vol. 14, pp. 655-666, October 2001.

[50] N-M. Wanas, “Feature-Based Architectures for Decision Fusion”, *Ph.D. Thesis***,** University of Waterloo, Canada, 2003.

[51] J.Kittler, M.Hatef, R.P.W.Duin, and J.Matas, “On Combining Classifiers”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no.3, pp.226-239, March 1998.

[52] C. Pelekanakis, “Generation of PCB Images using Monte Carlo Simulation”, *Diploma Thesis*, Technical University of Crete, Greece, 2002.

[53] A. D. H. Thomas, M. G. Rodd, J.D. Holt, and C. J. Neill, “Real-time Industrial Visual Inspection: A Review” , *Real-Time Imaging*, vol.1, pp. 139-158,1995.

[54] R. T. Chin, “Automated Visual Inspection: 1981 to 1987”, *Computer Vision Graphics Image Processing*, vol. 41, pp. 346-381, 1988.

[55] T. S. Newman and A. K. Jain, “A Survey of Automated Visual Inspection”, *Computer Vision and Image Understanding*, vol. 61, pp. 231-262, 1995.

[56] J. Mahon, N. Harris, and D. Vernon, “Automated Visual Inspection of Solder Paste Deposition on Surface Mount Technology PCBs”, *Computers in Industry*, vol. 12, pp. 31-42, 1989.

[57] L. Zhang, A. Deghani, Z. Su, T. King, B. Greenwood, M. Levesley, “ Real-time Automated Visual Inspection System for Contaminant Removal from Wool”, *Real-Time Imaging* , vol.11, pp. 257-269, 2005.

[58] S. E. Umbaugh, *Computer Vision and Image Processing*, Prentice-Hall PTR, USA, 1998.

[59] R. L. Harvey, *Neural Network Principles*, Prentice-Hall International, USA, 1994.

[60] J. Mahon, "Automatic 3-D inspection of solder paste on surface mount printed circuit boards", *Journal of Materials Processing Technology*, vol. 26, pp. 245-256, 1991.

[61] D. Enke and C. Dagli, "Automated misplaced component inspection for printed circuit boards", *Computers Industrial Engineering*, vol. 33, Nos 1-2, pp. 373-376, 1997.

[62] O. Oyeleye and E. A. Lehtihet, "A Classification Algorithm and Optimal Feature Selection Methodology for Automated Solder Joint Inspection," *Journal of Manufacturing Systems*, vol. 17, pp. 251-262, 1998.

[63] S. K. Shah, "Probabilistic Multifeature/Multisensor Integration for Automatic Object Recognition" , PhD Thesis, The University of Texas at Austin, August 1998.

[64] C. Taylor, J. Graham, and D. Cooper, "System Architectures for nteractive Knowledge-Based Image Interpretation," *Philosophical Transactions of the Royal Society of London (A) Mathematical and Physical Sciences*, vol. 324, pp. 457-465, 1988.

[65] M. Moganti, F. Ercal, C. H. Dagli, and S. Tsunekawa, "Automatic PCB Inspection Algorithms: A Survey," *Computer Vision and Image Understanding*, vol. 63, pp. 287-313, 1996.

[66] S. M. Bhandarkar, T. D. Faust, and M. Tang, "CATALOG: A System for Detection and Rendering of Internal Log Defects Using Computer Tomography," *Machine Vision and Applications*, vol. 11, pp. 171-190, 1999.

[67] A. R. Jimenez, A. K. Jain, R. Ceres, and J. L. Pons, "Automatic Fruit Recognition: A Survey and New Results Using Range/Attenuation Images," *Pattern Recognition*, vol. 32, pp. 1719-1736, 1999.

[68] C. Taylor, J. Graham, and D. Cooper, "System Architectures for nteractive Knowledge-Based Image Interpretation," *Philosophical Transactions of the Royal Society of London (A) Mathematical and Physical Sciences*, vol. 324, pp. 457-465, 1988.

[69] R. T. Chin and C. A. Harlow, "Automated Visual Inspection: A Survey," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 6, pp. 557-573, 1982.

[70] B. C. Eduardo, "Review of Automated Visual Inspection 1983 to 1993-Part I: Conventional Approaches," *SPIE-Intelligent Robots and Computer Vision XII*, vol. 2055, pp. 128-158, 1993.

[71] B. C. Eduardo, "Review of Automated Visual Inspection 1983-1993-Part II: Approaches to Intelligent Systems," *SPIE-Intelligent Robots and Computer Vision XII*, vol. 2055, pp. 159-172, 1993.

[72] M. Shirvaikar, "Trends in Automated Visual Inspection", *J Real-Time Image Proc*, vol. 1, pp. 41-43, 2006.

[73] Y. Nakagawa, "Automatic visual inspection of solder joints on printed circuit boards", Proc. SPIE, Robot Vision, vol. 336, pp. 121-127, 1982.

[74] E. Trucco, A. Verri, *Introductory Techniques for 3-D Computer Vision*, Prentice Hall, USA, 1998.

[75] D. S. Jayas, J. Paliwal, N.S. Visen, "Multi-layer Neural Networks for Image Analysis of Agricultural Products", *J. agric. Engng Res*, vol. 77(2), pp. 119-128, 2000.

[76] D. Braggins, "Recent Developments in Image Capture for Machine Vision," *Sensor Review*, vol. 15, pp. 11-14, 1995.

[77] M. Loinaz and B. Ackland, "Video Cameras: CMOS Technology Provides On-Chip Processing," *Sensor Review*, vol. 19, pp. 19-26, 1999.

[78] M. Schanz, W. Brockherde, R. Hauschild, B. J. Hosticka, and A. Teuner, "CMOS Photosensor Arrays With On-Chip Signal Processing," presented at *Proc. Of European Solid State Circuits Conference (ESSCIRC'97)*, Southampton U.K., 1997.

[79] E. R. Fossum, "CMOS Image Sensors: Electronic Camera-On-A-Chip," *IEEE Transactions on Electron Devices*, vol. 44, pp. 1689-1698, 1997.

[80] K. Yamada, T. Nakano, and S. Yamamoto, "Robot Vision Robust to Changes of Lighting Conditions Using a Wide-Dynamic Range Visual Sensor," *Electrical Engineering in Japan*, vol. 120, pp. 34-40, 1997.

[81] K. Yamada, T. Nakano, and T. Yamamoto, "A Vision Sensor Having an Expanded Dynamic Range for Autonomous Vehicles," *IEEE Transactions on Vehicular Technology*, vol. 47, pp. 332-341, 1998.

[82] D. Braggins, "Creating the Right Image," *Sensor Review*, vol. 16, pp. 16-21, 1996. D. Litwiller, *CCD vs. CMOS: Facts and Fiction*: Laurin Publishing, 2001.

[83] C. Seibold, "Comparison of CMOS and CCD Image Sensor Technologies," : ECE 442 Senior Design, 2002.

[84] S. Fathnam and G. Slavenburg, "Processing the New World of Interactive Media: The Trimedia VLIW CPU Architecture," *IEEE Signal Processing Magazine*, vol. 15, pp. 108-117, 1998.

[85] S. Purcell, "The Impact of Mpact 2: The Mpact 2 VLIW Media Processor Improves Multimedia Performance in PCs," *IEEE Signal Processing Magazine*, vol. 15, pp. 102-107, 1998.

[86] R. Ghaffarian, "Chip Scale Package Issues", *Microelectronics Reliability*, vol. 40, pp. 1157-1161, 2000.

[87] S-M. Chang, J-H. Jou, A. Hsieh, T-H. Chen, C-Y. Chang, Y-H. Wang, C-M. Huang, "Characteristic Study of Anisotropic-Conductive Film for Chip-on-Film Packaging", *Microelectronics Reliability*, vol. 41, pp. 2001-2009, 2001.

[88] Y. Ousten, S. Mejdi, A. Fenech, J.Y. Deletage, L. Bechou, M.G. Perichaud, Y. Danto, "The Use of Impedance Spectroscopy , SEM and SAM Imaging for Early Detection of Failure in SMT assemblies", *Microelectronics Reliability*, vol. 38, pp. 1539-1545, 1998

[89] J. B. Zhang, "Computer-Aided Visual Inspection for Integrated Quality Control", *Computers in Industry*, vol. 30, pp. 185-192, 1996.

[90] S. K. Nayar A. C Sanderson, L. E. Weiss, and D.D. Simon, "Specular surface inspection using structured highlight and Gaussian Images", *IEEE Trans. Robotics Automation*, Vol. 6, No 2, pp. 108-218, 1990.

[91] J. H. Kim and H. S. Cho, "Neural network-based inspection of solder joints using a circular illumination", Image and Vision Computing, Vol. 13, No 6, 1995.

[92] R. Nameth, "X-ray inspection aids process control", *Electronic Production*, pp. 33-37, July 1989.

[93] R. J. K. Wassink, *Soldering in Electronics*, Ayr, U.K.: Electrochemical Publications, 1984.

[94] R. Vanzetti, A. C. Traub, and J. S. Ele, "Hidden Solder Joint Defects Detected by Laser Infrared System", *Proc. of the IPC 24th Annual Meeting*, pp. 1-15, Apr. 1981.

[95] M. Juha, *The Economics of Automated X-Ray Inspection for Solder Quality*, IRT Corp. Publishing, San Diego, 1986.

[96] C. A. Keely, "Solder Joint Inspection using Laser Doppler Vibrometry", *Hewlett- Packard Journal*, pp. 81-85, Oct. 1989.

[97] S. S. Chen, A Three-Dimensional Approach to Automatic Solder Joint Inspection, Hauppauge, NY: Robotic Vision Systems Inc., 1988.

[98] D. W. Capson, S. K. Eng, "A tiered-color illumination approach for machine inspection of solder joints", *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 10, no.3, pp. 387-393, 1988.

[99] T-H Kim, T-H Cho, Y. S. Moon, S. H. Park, "Visual Inspection System for the Classification of Solder Joints", Pergamon: *Pattern Recognition*, vol.32, pp. 565-575, 1999.

[100] Y. K. Ryu, H. S. Cho, "A Neural Network Approach to Extended Gaussian Image Based Solder Joint Inspection", *Pergamon: Mechatronics*, vol. 7, no. 2, pp. 159-184, 1997.

[101] K. W. Ko and H. S. Cho, "Solder Joints Inspection using a Neural Network and Fuzzy Rule-Based Classification Method", *IEEE Trans. on Electronics Packaging Manufacturing*, Vol. 23, no. 2, April 2000.

[102] S. Jagannathan, "Automatic Inspection of Wave Soldered Joints Using Neural Networks", *Journal of Manufacturing Systems*, vol. 16, no. 6, pp.389-398, 1997.

[103] R.A. Zoroofi, H. Taketani, S. Tamura, Y. Sato, K. Sekiya, "Automated Inspection of IC Wafer Contamination", *Pattern Recognition*, vol. 34, pp. 1307-1317, 2001.

[104] J-M. Zhang, R-M. Lin. M-J. Wang, "The Development of an Automatic Post-Sawing Inspection System using Computer Vision Techniques", *Computers in Industry*, vol. 40, pp. 51-60, 1999.

[105] M. Moganti and F. Ercal, "Segmentation of Printed Circuit Board Images into Basic Patterns", *Computer Vision and Image Understanding*, vol. 70, no. 1, pp. 74-86, 1998.

[106] M. Moganti and F. Ercal, "A Subpattern Level Inspection System for Printed Circuit Boards", *Computer Vision and Image Understanding*, vol. 70, no. 1, pp. 51-62, 1998.

[107] J.H. Kim, H.S. Cho, "Neural Network-based Inspection of Solder Joints using a Circular Illumination", *Image and Vision Computing*, vol. 13, no. 6, pp. 479-490, 1995.

[108] J.H. Kim, H.S. Cho, S. Kim, "Pattern Classification of Solder Joint Images Using a Correlation Neural Network, *Engineering Applications of Artificial Intelligence*, vol. 9, no. 6, pp. 655-669, 1996.

[109] E. Guerra, J.R. Villalobos, "A Three -Dimensional Automated Visual Inspection System for SMT assembly", *Computers and Industrial Engineering*, vol. 40, pp. 175-190, 2001.

[110] E. Guerra, A. Manriquez, D. Schwartz, J.R. Villalobos, "Three Dimensional Automated Visual Inspection of Surface Mounted Devices", *Computers and Industrial Engineering*, vol. 33, pp. 365-368, 1997.

[111] A. R. Rao, “Future Directions in Industrial Machine Vision: A Casse Study of Semiconductor Manufacturing Applications”, Image and Vision Computing, vol. 14, pp. 3-19, 1996.

[112] R. J. Mammone, *Artificial Neural Networks for Speech and Vision*, Chapman & Hall, UK,1994.

[113] Fausett L, *Fundamentals of Neural Networks: Architectures, Algorithms and Applications*, Prentice-Hall, USA, 1996.

[114] D. P. Mandic and J. A. Chambers, Recurrent Neural Networks for Prediction, Wiley, USA, 2001.

[115] J. C. Principe, N. R. Euliano, and W. C. Lefebvre, *Neural and Adaptive Systems: Fundamentals through Simulations*, Wiley, USA, 2000.

[116] B. D. Ripley, *Pattern Recognition and Neural Networks*, Cambridge University Press, UK, 1996.

[117] J. W. Kay and D. M. Titterington, *Statistics and Neural Networks: Advances at the Interface*, Oxford University Press, UK, 1999.

[118] S. Bandyopadhyay and S. Pal, “Pattern classification with genetic algorithms: incorporation of chromosome differentiation”, *Pattern Recognition Letters*, vol. 18, pp. 119-131, 1997.

[119] C. T. Leondes, *Industrial and Manufacturing Systems*, vol. 4 of Neural Network Systems Techniques and Applications, Academic Press, USA, 1998.

[120] C. T. Leondes, *Image Processing and Pattern Recognition*, vol. 5 of Neural Network Systems Techniques and Applications, Academic Press, USA, 1998.

[121] K. I. Diamantaras and S. Y. Kung, *Principal Components Neural Networks: Theory and Applications*, Wiley, USA, 1996.

[122] D. W. Ruck, S. K. Rojers, M. Kabrisky, M.E. Oxley, and B. W. Suter. “The multilayer perceptron as an approximation to a Bayes optimal dicriminant function”, *IEEE Transactions on Neural Networks*, 1 (4): 296-298, 1990.

[123] E. A. Wan, “Neural Network classification: A Bayesian interpretation”, *IEEE Transactions on Neural Networks*, 1 (4): 303-305, 1990.

[124] T. Kohonen, “Self-organizing map”, *Proc. IEEE*, vol. 78, no. 9, pp. 1464 – 1480, 1990.

[125] H. Demuth and M. Beale, *Neural Network Toolbox – For use with MATLAB – User Guide version 3.0*, The Mathworks Inc., MA, USA, 1997.

[126] T. M. Mitchell, *Machine Learning*, McGraw-Hill, USA, 1997.

[127] J. J. Hopfield, “Neural Networks and Physical Systems with Emergent Collective Computational Abilities”, *Proceedings of the National Academy of Scientists*, vol. 79, pp. 2554 – 2558, 1982.

[128] J. J. Hopfield, “Neurons with Graded Response Have Collective Computational Properties like Those of Two-State Neurons”, *Proceedings of the National Academy of Scientists*, vol. 81, pp. 3088 – 3092, 1984.

[129] J. A. Farrell, and A. N. Michel, “A Synthesis Procedure for Hopfield’s Continuous – Time Associative Memory”, *IEEE Transactions on Circuits and Systems*, vol. 37, no. 7, July 1990.

[130] A. N. Michel, J. A. Farrell, and H-F Sun, “Analysis and Synthesis Techniques for Hopfield Type Synchronous Discrete Time Neural Networks with Application to Associative Memory”, *IEEE Transactions on Circuits and Systems*, vol. 37, no. 11, November 1990.

[131] G. Yen and A. N. Michel, “A Learning and Forgetting Algorithm in Associative Memories: Results Involving Pseudo-Inverses”, *IEEE Transactions on Circuits and Systems*, vol. 38, no. 10, Oct. 1991.

[132] E. B. Kosmatopoulos, M. Polycarpou, M.A. Christodoulou, P.A. Ioannou, “High Order Neural Network Structures for Identification of Dynamical Systems”, *IEEE Trans. Neural Networks*, vol. 6, no. 2, pp. 422-431, 1995.

[133] G. A Rovithakis, “Robustifying Nonlinear Systems using High Order Neural Network Controllers”, *IEEE Trans. Automatic Control*, vol. 44, no.1 pp. 102-108, 1999.

[134] G. A. Rovithakis, “Tracking Control of Multi Input Affine Nonlinear Dynamical Systems with Unknown Nonlinearities using Dynamical Neural Networks”, *IEEE Trans. Systems, Man and Cybernetics-Part B: Cybernetics*, vol. 29, pp. 179-189, 1999.

[135] G. A. Rovithakis, Robust Neural Adaptive Stabilization of Unknown Systems with Measurement Noise”, *IEEE Trans. Systems, Man and Cybernetics-Part B: Cybernetics*, vol. 29, pp. 453-458, 1999.

[136] G. A. Rovithakis, and M. A. Christodoulou, *Adaptive Control with Recurrent High Order Neural Networks*, Springer-Verlag, London, 2000.

[137] G. A. Rovithakis, M. Maniadakis, M. Zervakis, G. Filippidis, G. Zacharakis, A. N. Katsamouris, and T. G. Papazoglou, “Artificial Neural Networks for Discriminating Pathologic From Normal Peripheral Vascular Tissue”, *IEEE Trans. on Biomedical Engineering*, vol. 48, no. 10, Oct. 2001.

[138] G. A. Rovithakis, M. Maniadakis, and M. Zervakis, “A Hybrid Neural Network/Genetic Algorithm Approach to Optimizing Feature Extraction for Signal Classification”, *IEEE Trans. on Systems, Man and Cybernetics*, accepted Nov. 8, 2002, in press.

[139] J. H. Holland, *Adaptation in Natural and Artificial Systems*, Cambridge, MA: MIT press, 1975.

[140] J. H. Holland, “Genetic Algorithms”, *Scientific American*, pp. 66-72, July, 1992.

[141] D. E. Goldberg, *Genetic Algorithms in Search, Optimization and Machine Learning*, Addison-Wesley, USA 1989.

[142] S. Bandyopadhyay, C. A. Murthy, and S. K. Pal, “Pattern Classification using Genetic Algorithms”, *Pattern Recognition Letters*, vol. 16, no. 8, pp. 801-808, 1995.

[143] S. Bandyopadhyay, and S. K. Pal, “Pattern Classification with Genetic Algorithms: Incorporation of Chromosome Differentiation”, *Pattern Recognition Letters*, 18 (1997) 119-131.

[144] M. Maniadakis, “Genetic Algorithms for Classification and Function Identification applications”, Master Thesis in Technical University of Crete, 1999.

[145] T. Back, U. Hammel, and H. Schwefel, Evolutionary computation: comments on the history and current state, *IEEE Transactions on evolutionary computation*, vol.1, no. 1, April 1997.

[146] E. L. Davis, *Handbook of Genetic Algorithms,* Van Nostrand Reinhold, New York, 1991.

[147] N. R. Pal, “Soft Computing for Feature Analysis”, *Fuzzy Sets and Systems*, 103 (1999) 201 – 221.

[148] R. Cappelli, D. Maio, and D. Maltoni, “Multispace KL for Pattern Representation and Classification”, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, vol. 23, no. 9, Sept. 2001.

[149] S. K. Goumas, M. E. Zervakis, and G. S. Stavrakakis, “Classification of Washing Machines Vibration Signals Using Discrete Wavelet Analysis for

Feature Extraction", *IEEE Trans. on Instrumentation and Measurement*, vol. 51, no. 3, June 2002.

[150] R. C. Gonzalez and R. E. Woods, *Digital Image Processing*, Addison Wesley, USA 1993.

[151] I. Pitas, *Digital Image Processing Algorithms and Applications*, Wiley, USA, 2000.

[152] N. Otsu, "A threshold selection method from gray-level histograms", *IEEE Trans. On Systems, Man, and Cybernetics*, vol. 9, pp. 62-69, 1979.

[153] J. Kittler, and J. Illingworth, "Minimum error thresholding", *Pattern Recognition*, vol. 19, no.1, pp. 41-47, 1986.

[154] S. S. Reddi, S. F. Rudin, H. R. Keshavan, "An optimal multiple threshold scheme for image segmentation, *IEEE Trans. On Systems, Man, and Cybernetics*, vol. 14, no. 4, pp. 661- 665, 1984.

[155] J. N. Kapur, P. K. Sahoo, and A.K.C. Wong, "A new method for gray-level picture thresholding using the entropy of the histogram", *Computer Vision, Graphics, and Image Processing*, vol. 29, pp. 273-285, 1985.

[156] G. P. Babu, M. N. Narasimha Murty, "Optimal thresholding using multi-state stochastic connectionist approach", *Pattern Recognition Letters*, vol. 16, pp. 11-18, 1995.

[157] Hui-Fuang Ng, "Automatic thresholding for defect detection", *Pattern Recognition Letters*, vol. 27, pp. 1644-1649, 2006.

[158] N. Papamarkos, and B. Gatos, "A new approach for multithreshold selection", *Computer Vision Graphics and Image Processing – Graphical Models and Image Processing*,vol. 56, pp. 357-370, 1994.

[159] N. Papamarkos, C. Strouthopoulos, I. Andreadis, "Multithresholding of color and gray-level images through a neural network technique", *Image and Vision Computing*, vol. 18, pp. 213 – 222, 2000.

[160] L. K. Huang, and M. J. Wang, "Image thresholding by minimizing the measure of fuzziness", *Pattern Recognition*, vol. 28, pp. 41-51, 1995.

[161] H. D. Cheng, J-R Chen and Jiguang Li, "Threshold selection based on fuzzy c-partition entropy approach", *Pattern Recognition*, vol. 37, no. 7, pp. 857-870, 1998.

[162] H. D. Cheng, C. H. Chen, H. H. Chiu, and Huijuan Xu, "Fuzzy Homogeneity Approach to Multilevel Thresholding", *IEEE Trans. on Image Processing*, vol. 7, no. 7, July 1998.

[163] K. Ramar, S. Arumugam, S. N. Sivanandam, L. Ganesan, D. Manimegalai, "Quantitive Fuzzy Measures for Threshold Selection", *Pattern Recognition Letters*, vol. 21, pp. 1-7, 2000.

[164] P. K. Sahoo, S. Soltani, A.K.C. Wong, "A survey of thresholding techniques", *Computer Vision, Graphics, and Image Processing*, vol. 41, pp. 233-260, 1988.

[165] M. Sezgin, B. Sankur, "Survey over image thresholding techniques, and quantitive performance evaluation", *Journal of Electronic Imaging*, vol. 13, no. 1, pp. 146-156, 2004.

[166] S. U. Lee, S. Y. Chung, and R. H. Park, "A comparative performance study of several global thresholding techniques for segmentation", *Computer Vision, Graphics and Image Processing*, vol. 52, no. 2, pp. 171-190, 1990.

[167] C. A. Glasbey, "An analysis of histogram-based thresholding algorithms", CVGIP: Graphical Models and Image Processing, vol. 55, no.2, pp. 171-190, 1990.

[168] T. Pavlidis, “Algorithms for shape analysis of contours and waveforms”, *IEEE Trans. On Pattern Analysis and Machine Intelligence*, vol.2, no.4, pp. 301-312, 1980.

[169] Z. Chi, H. Yan, and T. Pham, *Fuzzy Algorithms: With Applications to Image Processing and Pattern Recognition*, Advances in Fuzzy Systems – Applications and Theory vol. 10, World Scientific Publishing Co, Singapore, 1996.

[170] Joseph O’ Rourke, *Computational Geometry in C*, Cambridge University Press, USA, 1994.

[171] W. L. Martinez, Angel R. Martinez, *Computational Statistics Handbook with MATLAB*, Chapman & Hall / CRC, Florida, USA, 2002.

[172] G. S. Fishman, Monte Carlo: *Concepts, Algorithms and Applications*, Springer-Verlag, New York, USA, 1996.

[173] C. P. Robert, G. Casella, *Monte Carlo Statistical Methods*, Springer-Verlag, New York, USA, 1999.

[174] P. Bremaud, *Markov Chains, Gibbs Fields, Monte Carlo Simulation, and Queues*, Springer Science + Buiseness Media, New York, USA, 1999.

[175] S. H. Nawab and J. M. Winograd, “Approximate signal processing using incremental refinement and deadline-based algorithms,” in *Proc. IEEE Int. Conf. Acoustics, Speech, and Signal Processing*, vol.5, (Detroit), pp. 2857-2860, May 1995.

[176] C–M. Wang, S–C. Yang, P–C. Chung et al, “Orthogonal subspace projection-based approaches to classification of MR image sequences”, *Pergamon: Computerized Medical Imaging and Graphics*, vol. 25, pp. 465-476, 2001.

[177] R. Lotlikar and R. Kothari, “Adaptive linear dimensionality reduction for classification”, *Pergamon: Pattern Recognition*, vol. 33, pp. 185-194, 2000.

[178] S. Svensson, I. Nystrom, G. Sanniti di Baja, “Curve skeletonization of surface-like objects in 3D images guided by voxel classification”, *Elsevier: Pattern Recognition Letters*, vol. 23, pp. 1419-1426, 2002.

[179] Guang Zhe Xu, M. Kaneko and A. Kurematsu, “Synthesis of Facial Caricatures using Eigenspaces and its Applications to Communication”, *Proc. of 1999 International Workshop on Very Low Bitrate Video Coding (VLBV’99)*, pp. 192-195, Kyoto, Japan, Oct. 1999.

[180] P. Agouris, A. Stefanidis, and J. D. Carswell, “Sketch-Based Image Queries in Topographic Databases”, *Academic Press: Journal of Visual Communication and Image Representation*, vol. 10, pp. 113-129, 1999.

[181] Y. Y. Tang, H. D. Cheng, C.Y. Suen, “Transaction-ring-projection algorithm and its VLSI implementation”, *J. Pattern Recognition Artificial Intelligence*, vol.5, pp. 25-56, 1991.

[182] X. Xie, R. Sudhakar, H. Zhuang, “On Improving eye feature extraction using deformable templates”, *Pergamon: Pattern Recognition*, vol. 26, no. 8, pp. 1235-1243, 1994.

[183] M.Datcu, K.Seidel, “Bayesian Methods: Applications in Information Aggregation and Image Data Mining”, *Intl. Archives of Photogrammetry and Remote Sensing*, vol. 32, Villadoloid, Spain, June 1999.

[184] G.W.Ng, Intelligent Systems – *Fusion, Tracking and Control*, Research Studies Press Ltd, UK, 2003.

[185] X. E. Gross, *NDT Data Fusion*, Arnold, London, 1997.

[186] L.Xu, A.Krzyzak, and C.Y.Suen, “Methods of Combining Multiple Classifiers and Their Applications to Handwriting Recognition”, *IEEE Transactions on Systems, Man and Cybernetics*, vol. 22, no. 3, pp. 418-435, May/June 1992.

[187] J.Kittler, M.Hatef, R.P.W.Duin, and J.Matas, “On Combining Classifiers”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20,no. 3, pp.226-239, March 1998.

[188] L.I.Kuncheva, “Switching Between Selection and Fusion in Combining Classifiers: An Experiment” *IEEE Transactions on Systems, Man and Cybernetics*, Part B: Cybernetics, vol. 32, no. 2, pp. 146-156, April 2002.

[189] J. Kittler, A. Hojjatoleslami, T. Windeatt, “Strategies for combining classifiers employing shared and distinct pattern representations”, *Pattern Recognition Letters* 16 (1997) 1373-1377.

[190] L. Lam, C.Y.Suen, “Optimal combinations of pattern classifiers”, *Pattern Recognition Letters* 16 (1995) 945-954.

[191] L.I.Kuncheva, L.C.Jain, “Designing Classifier Fusion Systems by Genetic Algorithms”, *IEEE Transactions on Evolutionary Computation*, vol. 4, no. 4, September 2000.

[192] A.Verikas, A.Lipnikas, K. Malmqvist, M.Bacauskiene, and A.Gelzinis, “Soft combination of neural classifiers: A comparative study”, *Pattern Recognition Letters* 20 (1999) 429-444.

[193] T.K.Ho, J.Hull, and S.N. Srihari, “Decision combination in multiple classifier systems”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 16, no. 1, pp. 66-75, 1994.

[194] H.Altincay, “On naïve Bayesian fusion of dependent classifiers”, *Pattern Recognition Letters* 26 (2005) 2463 – 2473.

[195] E.Kim, W.Kim, and Y.Lee, “Combination of multiple classifiers for the customer’s purchase behavior prediction”, *Decision Support Systems* 34 (2002) 167-175.

[196] G. Rogova, “Combining the Results of Several Neural Networks Classifiers”, *Neural Networks*, vol. 7, no. 5, pp. 777-781, 1994.

[197] R. R. Yager, M. Fedrizzi, and J. Kacprzyk, *Advances in the Dempster – Shafer Theory of Evidence*, Wiley, USA, 1994.

[198] T. Denoeux, “A k-nearest neighbor classification rule based on Dempster-Shafer theory. *IEEE Transactions on Systems, Man and Cybernetics* Part B, vol. 25, no. 5, 1995.

[199] C. R. Parikh, M.J. Pont, and N.B. Jones, “Application of Dempster-Shafer theory in condition monitoring applications: A case study”, *Pattern Recognition Letters* 22 (2001) 777 –785.

[200] S. See Ng, and H. Singh, “Data equalisation with evidence combination for pattern recognition, *Pattern Recognition Letters* 19 (1998) 227 – 235.

[201] Y. S. Huang and C.Y.Suen, “A Method of Combining Multiple Experts for the Recognition of Unconstrained Handwritten Numerals”, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 17, no. 1, January 1995.

[202] C.A. Shipp and L.I.Kuncheva, “Relationships between combination methods and measures of diversity in combining classifiers”, *Information Fusion*, 3 (2002) 135-148.

[203] L. I. Kuncheva, J. C. Bezdek, M.A. Sutton, “On combining multiple classifiers by fuzzy templates”, in *ProceedingsNAFIPS’98*, Pensacola, FL, 1998, pp. 193-197.

[204] L.I.Kuncheva, “Using measures of similarity and inclusion for multiple classifier fusion by decision templates”, *Fuzzy sets and systems*, 122 (2001) 401- 407.

[205] H.J.Kang, K.Kim, and J.H.Kim, “Optimal approximation of discrete probability distribution with *k*-order dependancy and its application to combining multiple classifiers”, *Pattern Recognition Letters* 18 (1997) 515-523.

[206] H.J.Kang, K.Kim, and J.H.Kim, “A Framework for Probabilistic Combination of Multiple Classifiers at an Abstract Level”, *Engng Aplic.Artif. Intell.* Vol. 10, no. 4, pp. 379-385, 1997.

[207] M. Ceccareli, and A. Petrosino, “Multi-feature adaptive classifiers for SAR image segmentation”, *Neurocomputing*, 14 (1997) 345-363.

[208] D. Maltoni, D. Maio, A. K. Jain, S. Prabhakar, *Handbook of Fingerprint Recognition*, Springer-Verlag, USA, 2003.

[209] A. R. Mirhosseini, H.Yan, K-M Lam, and T.Pham, “Human Face Image Recognition: An Evidence Aggregation Approach” , *Computer Vision and Image Understanding*, vol 71, no. 2, pp. 213-230, August 1998.

[210] D.M.J. Tax, M. van Breukelen, R.P.W. Duin, and J. Kittler, “Combining multiple classifiers by averaging or by multiplying?”, *Pattern Recognition*, 33 (2000) 1475-1485.

[211] L. Rodriguez-Linares, C. Garcia-Mateo, J-L. Alba-Castro, “On Combining Classifiers for Speaker Authentication”, *Pattern Recognition*, 36 (2003) 347-359.

[212] D.J. Mashao, M. Skosan, “Combining Classifier Decisions for Robust Speaker Identification” , *Pattern Recognition*, 39 (2006) 147-155.

[213] A. Kumar, D. Zhang, “Personal Authentication using Multiple Palmprint Representation”, *Pattern Recognition*, 38 (2005) 1695 – 1704.

[214] C-L.Liu, “Classifier Combination based on Confidence Transformation”, *Pattern Recognition*, 38 (2005) 11-28.

[215] L.I. Kuncheva, “A Theoretical Study on Six Classifier Fusion Strategies”, *IEEE Trans. on Pattern Analysis and Machine Intelligence*, Vol. 24, no.2, February 2002.

[216] J. Czyz, J.Kittler, L. Vandendorpe, “Multiple Classifier Combination for Face-Based Identity Verification”, *Pattern Recognition*, 37 (2004) 1459 – 1469.

[217] L-Y Zhai, L-P Khoo, S-C Fok, “Feature Extraction using Rough Set Theory and Genetic Algorithms – An Application for the Simplification of Product Quality Evaluation”, *Computers & Industrial Engineering*, 43 (2002) 661-676.

[218] R.W. Swiniarski, A. Skowron, “Rough Set Methods in Feature Selection and Recognition”, *Pattern Recognition Letters*, xxx (2002) xxx.

[219] Z. Pawlak, “Rough Sets”, *International Journal of Computer and Information Sciences*, 11(5) (1982) 341-356.

[220] S. Roberts and R. Everson, *Independent Component Analysis: Principles and Practices*, Cambridge University Press, United Kingdom, 2001.

[221] A. Hyvarinen and E. Oja, “Independent Component Analysis: Algorithms and Applications”, *Neural Networks*, 13 (2000) 411-430.

[222] S. Akaho, “Conditionally Independent Component Analysis for Supervised Feature Extraction”, *Neurocomputing*, 49 (2002) 139-150.

[223] S. Simone, “Overview of Independent Component Analysis Technique with an Application to Synthetic Aperture Radar (SAR) imagery processing”, *Neural Networks*, 16 (2003) 453-467.

[224] V.G. Kaburlasos, *Towards a Unified Modeling and Knowledge-Representation based on Lattice Theory*, Springer-Verlag, The Netherlands, 2006.

[225] V.G. Kaburlasos, V. Petridis, “Fuzzy Lattice Neurocomputing (FLN) models”, *Neural Networks*, 13 (2000) 1145-1169.

[226] G.C. Anagnostopoulos, M. Georgiopoulos, “Category regions as new geometrical concepts in Fuzzy-ART and Fuzzy-ARTMAP, *Neural Networks*, 15(10) (2002) 1205-1221.

[227] L. Breiman, “Bagging Predictors”, *Machine Learning*, 24(2) (1996) 123-140.

[228] V.G. Kaburlasos, S. E. Papadakis, S. Kazarlis, “A genetically optimized ensemble of σ-FLNMAP neural classifiers based on non-parametric probability distribution functions. In: Proc Intl J Conf Neural Networks (IJCNN’03), Portland, OR, 2003, pp 426-431.

[229] J. Kittler, A framework for classifier fusion: it is still needed? in : F. J. Ferri, J. M. Inesta, A. Amin, P. Pudil (Eds.), Multiple Classifier Systems, Lecture Notes in Computer Science, vol. 1876, Springer, Berlin, 2000, pp. 52-66.

[230] B. S. Bennett, *Simulation Fundamentals*, Prentice Hall International, UK, 1995.

[231] “Growth Opportunities for the World SMT Inspection Equipment Markets”, Frost & Sullivan market survey report, May 2005.

[232] T. Lecklider , “PCB Inspection Outlook for 2005”, Evaluation Engineering, Dec 2004

[233] David M. Mendez, "An Integrated Test and Inspection Strategy," APEX Proceedings, 2000.