Technical Report

Pattern Recognition and Image Processing Group
Institute of Automation
Vienna University of Technology
Treitlstr. 3/1832
A-1040 Vienna AUSTRIA
Phone: +43 (1) 58801-8161
Fax: +43 (1) 5054668
E-mail: sab@prip.tuwien.ac.at

PRIP-TR-46 April 11, 1997

A Highly Adaptable Concept for Visual Inspection

Robert Sablatnig

Abstract

To be acceptable in industry, vision systems must be inexpensive, within the speed of the production-line flow, and very accurate. While visual inspection is high in potential, at present the design and implementation of automatic visual inspection systems is labor-intensive. In addition, most of the visual inspection systems are developed in isolation with no systematic approach. Increasing flexibility to allow the inspection of parts whose positions are less constrained is desirable.

This work aims to show a systematic automated visual inspection concept that separates the detection of primitives from the model-based analysis process. This separation is obtained by defining a general analysis graph for inspection, containing detail relations that represent detection algorithms. Together with an object-specific description, defined in a socalled description language, the analysis graph is instantiated. Existing pattern recognition software is re-used in the detection stage and therefore the use of any detection algorithm is possible without changing the analysis. The concept can be seen as a "recipe" for solving industrial applications, stating which kind of decision have to be made at which stage.

An industrial application of the concept, for which industrial constraints have to be considered, is shown in the example of an automated visual inspection system for analogue watermeters used for calibration. Results concerning time, accuracy, and reliability of the specific inspection task are given. The flexibility of the concept is demonstrated by testing the analysis process with the description of other instruments (a hygrometer and a clock), which is performed by adapting the analysis graph but without changing the detection algorithms.

Acknowledgments

This thesis would not have been possible without the guidance, advice and support of numerous people in our PRIP (Pattern Recognition and Image Processing) laboratory at the Vienna University of Technology, particularly I want to thank:

- **Prof. Walter Kropatsch**, my thesis supervisor, who has always shared his ideas with me and demonstrated to me what it means to work scientifically. His valuable suggestions and criticism have influenced this thesis to a large extend and made the completion of this work possible.

- **Horst Bischof**, for many thorough discussions on the dissertation topic, for carefully reviewing drafts of this thesis, and for his advice in everyday office life.

- **Aleš Leonardis**, for many discussions on computer vision and visual inspection, for his collaboration on generic algorithms, for providing the ExSel software, and for reviewing drafts of this thesis .

- **Christian Menard**, with whom I share the office for years, for his friendship, support and for cheering me up "*mit Rat und Tat* (advice and act)" when times were tough on me.

- **Amy Krois-Lindner**, for carefully proof-reading this thesis.

Without the persistent and effective work of our system administrator *Alexius Korzinek*, the completion of this work would not have been possible. Furthermore I would like to thank my students and colleagues, who have contributed to this work: *Michael Neuhauser, Volker Lainer, Norbert Brändle, Hannes Föttinger, Robert Brandner, Christian Hansen,* and *Christian Liska*. I am thankful to *Helmut Kofler* for many helpful suggestions regarding mathematical problems and *Bergitta Göbel* for solving all bureaucratic problems patiently and successfully. During writing my thesis I spent 3 months in the Netherlands at the Department of Computer Science at Leiden University. I thank *Nies Huijsmans, Jan Rekers,* and *Roderick Bloem* for their support and the productive atmosphere.

I would like to express my gratitude to my parents who have made my studies possible and who always encouraged and supported me in my goals. Last, but not least, special thanks to **Maria** who had a hard time during writing this thesis but showed always patience even when I did not have time for her.

Contents

Chapter 1

Introduction

In nature, vision systems are of paramount importance to survival. The eye-brain combination makes it possible to absorb, process and react to large amounts of information about the surroundings, all without any physical contact [VER91]. It has now been well over 30 years since several individuals and groups made concerted efforts to automate visual perception in the research discipline of *Computer vision* [NAL93]. Computer vision, sometimes also called image understanding or scene analysis, describes the automatic deduction of the structure and properties of a possibly dynamic three-dimensional (3-d) world from either a single or multiple two-dimensional (2-d) images of the world as a combination of image processing, pattern recognition, and artificial intelligence technologies [HAR91]. Computer vision describes a process that tries to recognize and locate position and orientation, as well as describe imaged objects in a 3-d environment, like the human visual system does. Since the description of the state of the physical world from inherently noisy and ambiguous images of the world is a complicated goal to be accomplished in a reliable, robust, and efficient manner, there have been attempts to solve this problem; however, there is no common solution available.

From the beginnings researchers tried to convert the results achieved in basic research into applications to prove that their algorithms work. The pattern recognition and more generally the computer vision field has the potential and promise to provide the technology to develop a variety of automated systems that are capable of operating under diverse conditions, delivering consistent results, working in environments not suitable for humans, and situations where human workers have to perform a repetitive, tiring and error pruning task [BHA94]. To fulfill the needs of industry and consumers, many different (often called "real-world") applications of computer vision were introduced, making work easier in the fields of navigation, manufacturing, quality control, remote sensing, cartography, target recognition and tracking, medical image analysis, document analysis to name a few. Generally, these applications are multidisciplinary and require a combination of science, engineering, and art, resulting in a challenging task to develop a successful computer vision application [BHA94].

1.1 What is Visual Inspection?

Historically, the first industrial application area was in the manufacturing industry because there was a strong desire to automate the production process and to control the final product. The application of computer vision is therefore called *machine vision*; the relationship between machine automation and computer vision is also represented in the choice of the term. The term was introduced in the mid 80ies; Batchelor, one of the pioneers in this field, first used the expression "*practical pattern recognition*" [BAT78] and changed it to Machine Vision in 1985 [BAT85]. Haralick defines a machine vision system as "*a system capable of ac-*

quiring one or more images of an object, capable of processing, analyzing and measuring various characteristics of the acquired images, and interpreting the results of measuring in such a way that some useful decision can be made about the object" [HAR91]. This global definition shows that there are various different applications in industry, making a further distinction between different application areas necessary. In general, however, there is little agreement how this subdivision should be made, as there is no unique definition that can be found in the literature (see for instance [BAT85,BAT92,BHA94,HAR91,HUN85,MAR92, VER91]), a definition of the terms from the author's point of view is necessary. The following hierarchical subdivision of machine vision tasks can be enumerated:

- *Locating:* Specific coordinates for the a priori known physical position of the object should be determined.

- *Recognition and Identification:* Characters, parts, components, etc. should be recognized, identified, classified, categorized, or sorted by assigning them to some category from the set of given categories. In some applications there is a further distinction between recognition and identification; identification is said to be a sub-process of recognition.

- *Measurement or Gauging:* Specific positions or dimensions of objects should be measured by one- or two-dimensional non-contact light sensitive sensors and in 3-d applications by structured light or other triangulation techniques.

- *Motion Estimation:* Moving objects should be detected and their position, velocity and moving direction should be determined. Equivalently, if the camera is moving, position, velocity and trajectory of the camera should be determined.

- *Guidance and Control:* Mobile or static robots, robot arms, vehicles, etc. should interact with their environment in combination with a vision system. This includes tracking, navigation and control which are often also called *Robot Vision*, too.

- *(Visual) Inspection:* Objects should be checked if they have been manufactured within permitted tolerances conforming to a given standard, functional or cosmetic defects should be detected, comparing measurements or images to preselected tolerance limits for quality inspection and/or sorting decisions.

There is no clear boundary between the enumerated tasks, this subdivision is structured hierarchically - from low machine vision tasks, like locating, to more sophisticated tasks like guidance and control. Motion estimation applications for instance may also use recognition and identification techniques, measurement operations will often result in an accept/reject decision which can be considered as inspection. However, there are applications that use only a limited number of pattern recognition techniques. Figure 1.1 shows the hierarchy of vision terms in a simplified diagram, all terms are embedded in pattern recognition, since all use its methods. Other image analysis tasks are not mentioned explicitly, other Computer Vision research areas are shown in an exemplary manner only.

It should be mentioned that the proposed definitions (like the subdivision of machine vision tasks) are not globally accepted since there exist different definitions. Class guidance and

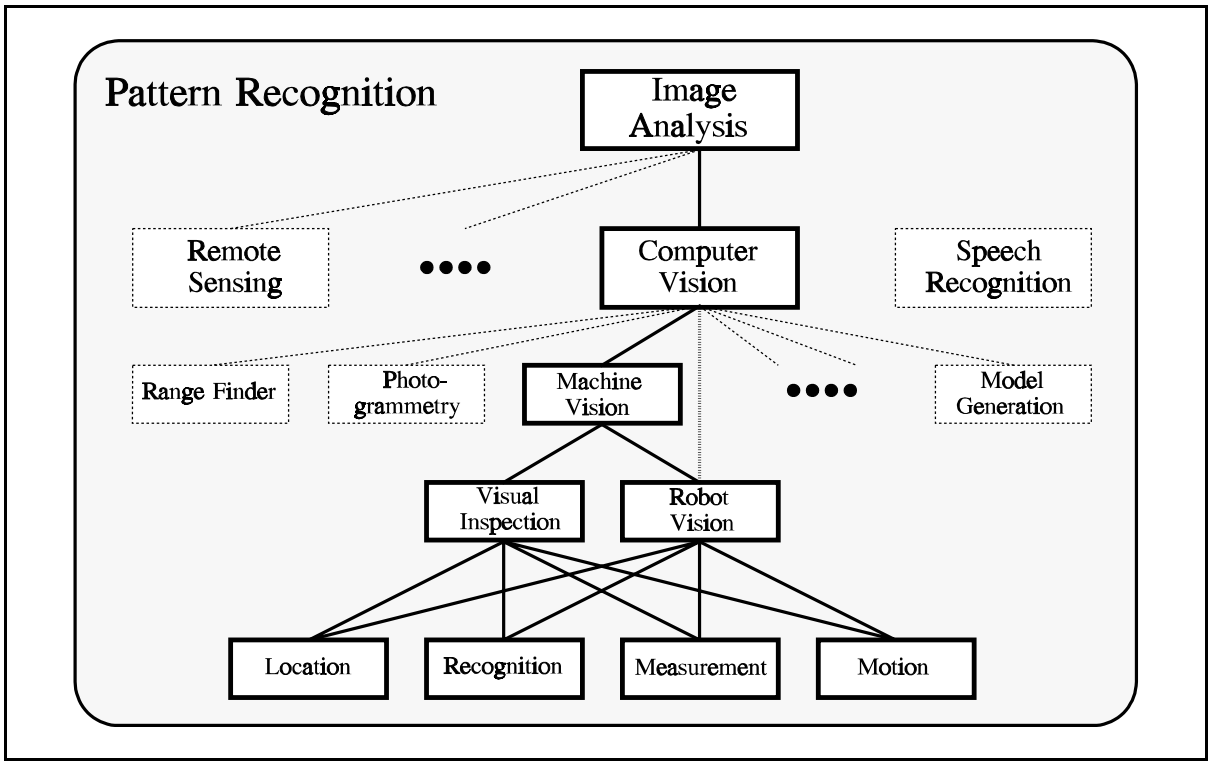


Figure 1.1 Hierarchical taxonomy of terms (simplified)

control is also called *Robot Vision*, which can be seen on the same level as Machine Vision. Moreover, in specific research and application areas terms are used with different meanings and therefore there exists a strong confusion in using the correct terms. The Society of Manufacturing Engineers (SME), for instance, uses the term Machine Vision instead of Visual Inspection; researchers in the Computer Vision field often do not distinguish between Computer- and Machine Vision, calling Machine Vision *Applications of Computer Vision*, the Society for Optical Engineering (SPIE) on the other hand, defines inspection as an application of Machine Vision and Robot Vision as a part of inspection. However, the taxonomy used in this dissertation is wide spread and accepted.

Currently the main application of machine vision lies in the **A**utomated **V**isual **I**nspection (in short called AVI) of industrial products and to a lesser extent in robot vision, because these applications increase productivity and improve product quality in the manufacturing process [BAT96]. The electronics industry is the most active one in applying computer vision techniques for AVI of their products like printed circuit boards, integrated circuit chips, photomasks etc., since significant technological advances in design and production of electronic assemblies have increased the speed of production while reducing physical size. Such production advances have complicated the existing human visual inspection process. Inspection systems have been retrofitted onto existing production lines and generally AVI systems now appear in every major industrial sector [BAT96].

Since there is a large variety of different inspection tasks, a subdivision into different distinct sub-categories is necessary for visual inspection, too. There are again different taxonomies in the relevant literature. However, there exist common contents within the sub-categories. Two major subdivisions can be identified:

- *Flaw Detection:* Flaw or defect detection refers to the detection and/or classification of an unexpected perturbation of an otherwise (locally) homogeneous surface. This category is characterized by the need to examine large surfaces, while the relative area of flaws is small. This kind of inspection systems is mainly used for web and steel inspection, where the objects move at high speed in the production line.

- *Model-based Inspection:* The inspection involves comparing the object under consideration with some object model which describes the relevant features of the object. There are bounds or tolerances specified in the model within which measurements taken from the object must lie, in order for the object to be acceptable. Due to the use of a (shape) model, this category is also called shape conformity checking.

There is a class which is a mixture between the two main-categories named flaw or *defect classification*. Here a model of defects is used to classify defects detected by the flaw detection system. Since flaw detection and defect classification use only very limited pattern recognition algorithms, these two sub-categories are not discussed in detail within this thesis.

Model-based inspection operations involve a comparison between an actual object (also called product or item), or part of one, and a relevant specification standard, yardstick, criterion or expectation (the so-called object-model) to which reference is made. Such comparisons are not confined to inspection operations, there are other inspection-related processes that overlap (such as testing and commissioning), or within the inspection operation (such as locating, recognition, and measurement). Processes that lie within inspection were previously discussed; testing and commissioning have much in common with inspection, although there are differences given below:

- *Testing:* Testing involves active examination of specific operational functions of the product. Before dispatch, energy is applied to an instrument or machine. Its performance under actual or simulated load is then analyzed with the aid of appropriate instruments [HIL85a] (which of course can also be visual inspection systems thus producing the overlap). Testing also describes general procedures right in the beginning of manufacturing operations, samples of raw material are for instance analyzed to ascertain their physical and mechanical properties before larger quantities are released to production.

- *Commissioning:* Commissioning is similar to testing, taking place where the product is to be installed. It is intended to ensure that the instrument, machine or product conforms to the customer's expectations and product specification and seeks to ensure conformity. Commissioning operations are usually one-of-a-kind inspections carried out manually.

Up to now we sought to define the connection between Machine Vision and Visual Inspection and found out that the attempt to define the meaning of inspection is fraught with contradictions and ambiguities, however, a broad (and common) definition is possible [BAT85,BAT92,HAR91,NEW95]:

> *Inspection is the process (or operation) of determining if a product (also referred to as a part, object, or item), in whole or part deviates from a given set of specifications (also referred to as relevant, achievable standards).*

1.2 Motivations of Automated Visual Inspection

In mass-production manufacturing facilities, an attempt is often made to achieve 100% quality assurance of all finished products. One of the most difficult tasks in this process is that of inspecting for visual appearance - an inspection that seeks to identify both functional and cosmetic defects [CHI86]. With an emerging requirement for improved quality control within the manufacturing industry, the use of visual inspection of the manufactured product becomes a necessity, especially to fulfill the ISO 9000 industrial quality standard [ISO94,NOV95, PET95]. With small batch manufacture of highly variant products, visual inspection becomes critical. Using large batch production the inspection process can be prepared and refined manually without significant time constraints, as the set-up costs can easily be absorbed over the total production run. However, with one-off or small batch production the manual set-up of the inspection process becomes unacceptable.

In industry economic considerations play an important role: deciding whether or not a process should be automated. Therefore, it is evident that the question: "When should an AVI system be used?" should be answered not only in a technical sense "is it possible to automate the inspection?" but also in economic, marketing and legal terms [HIL85a].

The containment and preferably reduction of the need for inspection is a long-standing management objective, since inspection requires expenditure on manpower, facilities, and equipment. Simply replacing labor costs associated with human inspection is not a sufficient justification; there are, however, other important reasons [FRE89]. Humans are notoriously inconsistent, both from one to another, and from time to time. They fatigue easily, they reject objects more on the basis of quotas than on actual defect levels. Sometimes defects are rare events but ones which require a rapid response to avoid major economic loss. Machines work patiently for days, weeks, or months at good production, and are good at counting and quantifying, which plays an important role in an economic view.

Technical reasons for installing an AVI system are given by the increasing complexity of designs, as a result of miniaturization, increases of machine speed, controls, and tooling in the inspection area. Manual systems cannot always contain these repercussions as resolution, operational and reaction speeds are too slow [BAT85].

Recent legislation has provided further legal reasons for automated inspection: inspection operations can no longer be performed by humans because machinery is guarded in accordance with health and safety regulations. Moreover, customers are taking advantage of product liability legislation, so inspection is necessary to improve standards. Products should not be shipped to customers unless they can be guaranteed to be truly defect-free. Achieving 100% inspection using humans typically requires a considerable level of redundancy (so called 300% inspection [FRE89]). In fact, the human visual inspection is at best between 70% [BAT96] and 90% [NEW95] effective and furthermore, this effectiveness can only be achieved if a rigidly structured set of inspection checks is implemented [SMI93]. However, in some critical applications, such as aerospace and medicine, even a single faulty product is unacceptable and may cause disasters [NEW95]. Furthermore, in other industrial areas like in the food industry, products with aesthetic appeal sell better.

The majority of visual inspection applications seek to solve quality problems through the use of measuring techniques, model-based visual inspection techniques, and flaw detection techniques. The goal of these applications is to remove defective products from the manufacturing process before additional value is added to the defective product or before the defective

product is released to the consumer. Therefore, inspection systems should monitor all manufacturing steps and report defects as soon as possible [HOL84].

Non-visual automated inspection tasks using contact inspection devices like *coordinate measuring machines* (CMM) are slow in operation because a single touch probe is used to take individual readings from the object. Furthermore, they require that the part is stopped, carefully positioned, and repositioned several times. The programming of the CMM is based on engineering drawings. The location of the reference points is indicated by an operator, who carefully guides the tactile probe to these points. In the inspection, the CMM is unable to proceed until the locations of the reference points have been established. Thus, the CMM approach is not suited to a totally automated method of inspection, since the use of a tactile probe is not always possible due to the nature of the object [MAR92]. Since AVI operations are non-contact, there is also a lower risk of product damage during inspection [HIL85a].

The following statement sums up conditions under which an AVI system should be used:

> *Automated visual inspection is feasible when the application has large part volumes, demands precise measurements, requires consistent or 100% inspection, or is in a hazardous environment. It improves the overall effect on the product guaranteeing a predictable quality, yielding greater customer satisfaction and thus increased market share, while lowering production costs* [HIL85a].

While visual inspection is high in potential, at present the design and implementation of automatic visual inspection systems is labor intensive. In addition, most of the visual inspection systems have been developed in isolation with no systematic approach which has led to the design of inflexible customized solutions involving very high system engineering costs over the last twenty years [CHI82,CHI88,NEW95].

1.3 Constraints of AVI

In the manufacturing industry, tasks that require visual inspection (and this is ideally every single manufacturing step) were always carried out by human operators in the very beginning. The continuing development of machine vision and especially AVI, so that it now provides a competitively priced, robust solution to inspection problems, is initiating a change from human to machine vision for inspection purposes [GAL90].

For AVI, machine vision techniques have both advantages and disadvantages compared with human vision. Humans have integrated vision systems that are interdependent on various sensors, using adaptive coupling with the brain and other sensory organs. In addition, the level of fatigue, previous training, and knowledge all affect the performance of the human visual system in quantifiable ways. Therefore, comparisons can only be made between the two visual systems on a functional basis, as shown in Table 1.1.

When planning to use or develop an AVI system one should bear in mind the overall goal of AVI systems: *An AVI system should solve a given inspection task within given constraints*. Therefore, the constraints of AVI have to be discussed in detail prior to any development or installation of AVI systems. There are several technical constraints as well as economic ones. Hill [HIL85a] claims that the constraints which impinge on automated visual inspection may be described as economical, operational, environmental, and intrinsic constraints:

Topic	Human	Machine
Flexibility	Very adaptable and flexible as to task and type of input	Very rigid as to task, requires quantized data
Ability	Makes accurate estimates on subjective matters	Can make dimensional measurements based on pre-determined data inputs
Color	Subjective in color	Measures magnitude of chromatic parameters
Sensitivity	Adaptive to lighting conditions, physical characteristics of surface of the object and distance to the object. Limited ability to distinguish between shades of gray, varies as function of individual from day to day	Sensitive to level and frequency of illumination as well as physical nature of surface and distance to object. Ability to quantize is high and fixed by sensor, environment and system characteristic
Response	Speed is slow - in the order of 1/10 second	Speed is very high - in the order of 1/1000 second and higher, depending on hardware
Perception	Perceives brightness on logarithmic scale. Affected by surroundings (background)	Perceives brightness in either linear or logarithmic scale
Spectrum	Limited to visual spectrum (380-700nm)	Uses entire spectrum, from X-rays to Infrared

Table 1.1 Comparison of human and machine vision [GAL90]

- *Economical*: Since development costs can be as high as $100,000 or more (there are AVI systems with development costs of several million dollars) [VGO91], all available commercial AVI systems are high priced. Therefore, prior to any installation or development of an AVI system, it should be determined whether the economic expense is justified and whether it can be amortized either over the number of installed systems (in the development case) or due to quality, safety, labor cost, and legislation benefits.

- *Operational*: Management has a logistic duty to provide cover in the event of a breakdown in the availability of the AVI system. The speed with which assistance from specialist personnel can be obtained and the need for personnel to ensure the day-to-day operation of the system must be taken into account.

- *Environmental*: Industrial plants are a very hostile environment for vision hardware. There are a variety of different causes of problems for vision systems, such as distorted products due to essential clamping, machine and tools vibration that interferes with the sensor, mist and sprays that partially obliterate products, grime and dirt that obscure information for inspecting the product, thermal changes which disturb measurements, electrical noise that influences sensors, starting and stopping of adjacent machines (leads to voltage surges), to name only a few.

- *Intrinsic*: In contrast to other constraints, intrinsic constraints are related to the capabilities and suitability of equipment for the specific application. These constraints are most relevant for people working in the field of developing AVI systems, since they have to deal with them. Therefore, they are discussed here in detail.

There are five major intrinsic constraints for vision systems in industry that can be found in the literature [BAT92,CHI92,HIL85a,NEW95,MAR92], summarized below:

- *Speed*: The duration of image acquisition, detection and analysis must be within the speed of the production line flow. The required inspection rate may therefore lie below a fraction of a second which is a severe problem if elaborate algorithms are used. For most assembly line inspections, the upper time limit for inspection is about 1 second [NEW95]. Often the term *real-time inspection* is used in this context, but it is difficult to formally define what is meant by real-time. Van Gool has suggested: "*the AVI system should not be the major bottleneck for reducing cycle time or robot operation speed*" [VGO91]. Usually the term real-time or *computational-delay* refers to the time it takes to receive the results of the analysis of a given image, after the object has been placed in front of the sensor [BAT92]. The *throughput rate* in contrast refers to the number of objects per second that can be viewed and the resulting images processed.

- *Performance or reliability*: The percentage of successfully detected defects must be close to 100%. If for instance a company produces 10,000 products a day, of which on the average 500 products are faulty, a defect recognition rate of 80% would result in 100 faulty products per day not recognized, which is unacceptable. Thus, performance is measured in terms of two parameters: *Accuracy of detection of defects* and *false alarm rate* stated in percentage error. The performance of a system can be adjusted by improving one of the parameters at the expense of the other, biasing the system to make "safe" decisions.

- *Accuracy*: Accuracy in contrast to accuracy of detection of defects, refers to the geometric and/or radiometric resolution of the sensor and the detection algorithms, in 2-d, 3-d, gray-levels, colors and datadepth. Thus accuracy defines the limitations of the system in terms of resolution.

- *Flexibility or versatility*: The inspection procedure should be able to handle objects that are in unexpected positions and orientations. Furthermore, it should be flexible enough to accommodate changes in products. If the layout of a product is for instance slightly changed it should not be a problem to adapt the inspection system without consulting a vision specialist or changing the analysis software.

- *Cost*: The costs of an AVI system are divided into two factors, the image processing, computing, and illumination hardware and the design, development, and system integration costs, which are the largest components, resulting also in high overall costs. The computing power of the hardware is still increasing rapidly while the costs are decreasing. Software development and system integration in contrast, while increasing in quality, are also increasing in costs.

Summing up all constraints, one can see that the automation of a visual inspection task is related to different sub-problems; thus the tradeoff between different constraint parameters is difficult to evaluate. Freeman and Chiu claim that "*it is roughly correct to say that among all the industrial applications for which machine vision* (and therefore also AVI) *"seems" initially appropriate, one third turn out to be unsolvable* (in terms of performance), *one third*

can be solved by other, non-vision alternatives (mechanical inspection or part redesign), *and only one third can be handled reliably by vision technology*" [FRE89]. Thus, all of the constraints have to be considered before an automation of a visual inspection task is undertaken.

1.4 Problems of AVI

Industry needs automated visual inspection because in the manufacturing process many uncertainties like tolerances, defects, relative position and orientation error, etc. exist, which can be resolved by vision sensing [FRE89]. Vision has undergone a development of over 30 years of research (in the computer vision field) and application (in the machine vision field), resulting in a wide range of different algorithms and systems. The problem is how to match the technology to a specific application in an optimal, cost-effective way. From the viewpoint of industry there are several difficulties for automated visual inspection to be used widely:

- *Cost*: Development costs are still too high; vision software development, in particular makes up a substantial portion of the cost of a new application.

- *Speed*: Due to increasing production rates and machine speed, AVI has to be faster, too.

- *Lack of personnel*: There is a lack of non-specialist personnel in industry competent enough in computer vision to set up applications [CRE93]. This situation is similar to that of expert systems before the development of expert system user interfaces. Nevertheless, selecting the appropriate vision algorithm for a specific problem is not a trivial matter.

- *Technology*: At present AVI systems require controlled environment and precise positioning, are unable to handle shadows, highlights, occlusions and are unable to inspect surface properties [CHI92].

- *Ad hoc solutions*: Solutions to ensure robustness are very specific in their application [MAR92]. This approach seems to be the best way to solve a particular problem since such solutions are dealt with case-by-case and tend to be simpler, more compact, and cheaper, but customized techniques for specific problems cannot be duplicated or re-used. However, industry always tended to favor this approach [FRE89] because of simplicity, reliability and easy maintenance.

Knowing the problems, one can try to find solutions for the different aspects. The major drawbacks are the high set-up costs resulting from the extensive pre-inspection set-up by experienced operators, hard- and software development costs, labor, and maintenance costs. The key to solving the problem of flexibility is the development of visual inspection systems which are able to inspect a large variety of different objects without or only partly changing the analysis algorithm. In short, there is a requirement for generic visual inspection cells based on the principles of "Soft Automation" and the "Universal Factory", in which product modelling is coupled with flexible manufacturing systems and flexible inspection cells which can produce any product with the minimum of design and quality control constraints [TOD88a]. The potential exists for general purpose inspection cells to be constructed that are

capable of performing a variety of different inspection tasks [MAR92]. This cell should include different visual inspection methods, a flexible lighting arrangement, as well as robot arms for pick and place tasks.

There are two different strategies for designing visual inspection systems [FRE89]:

- *A bottom-up design* provides a quick solution for a specific problem by solving it case-by-case, in an ad-hoc way. This solves the specific problem with a diagnostic method: First there is an interactive experimentation stage, during which given datasets are tested with existing tools. In this stage all parameters are evaluated and adjusted in order to solve the given task in the best way and then the program code is adapted based on trial and error. The advantage of this strategy is that the solution is compact and simple, just sufficient to solve the specific problem.

  The central problem of this quick solution lies in solving a very specific case. For each application all possible techniques and existing tools have to be explored to find the most optimal solution that solves the problem. The analysis process gained in the specific case can rarely be re-used. If, for instance, the layout of the object to be inspected is slightly changed a complete re-design of the inspection system will be necessary.

- *A top-down design* starts with the definition of the problem space in which the specific problem is embedded. 'Legal' changes in the input data are only specific aspects of the problem space. This can be seen as a development of a general algorithm for a restricted domain, which needs to deal with a variety of possible, different problems. The problem space can be described by an abstract language which covers both the possible inputs and the possible outputs. The analysis successively refines the abstraction until operators can be applied on data. As a result such systems can get very complex since they usually have many different algorithms with a tremendous number of parameters that need to be tailored to solve each specific problem. Therefore, expert systems and knowledge based pattern recognition systems are used to capture the knowledge of the visual inspection system designer that allow the system to be used by less skilled technicians.

The two different strategies could also be regarded as *traditional automated visual inspection* for the bottom-up strategy since industry has favored this approach up to now, and *systematic automated visual inspection* for the second approach which is the main trend of the current research [MAR92,NEW95,BAT96] in the attempt to overcome the drawbacks and difficulties mentioned above.

1.5 What is to Come?

In the previous section drawbacks, limitations and difficulties were discussed, and the resulting attempt to overcome them using a systematic approach. This approach ensures that costs are reduced, due to re-usable systems and that less qualified personnel can operate them, with the help of a user interface. As far as speed and technology are concerned, faster hardware is under development (the processing speed doubles every year), so that better algorithms can be used to solve vision problems. At present, however, these systems are not available; there are only prototypes in use and almost no new concepts can be found in the literature. The

main reason for this is company politics; developments made by so-called *Machine Vision Companies* are not reported in the literature, since competitors in the market could use these concepts and sell them better. In the academic field only a few researchers are working on systematic AVI, the majority is trying to adapt computer vision algorithms to machine vision applications (see [NEW95]), struggling with system engineering problems [BAT96]. Therefore, work on this research topic helps to advance progress in AVI.

This dissertation aims to show a visual inspection concept that speeds up the development of automated visual inspection systems by providing a concept for software re-use and systematic set-up, adding a higher degree of flexibility to the inspection system. This flexibility decreases the set-up costs by taking into account different types of data, positioning, occlusions and tolerances. The speed constraint in terms of hardware will not be discussed, since processing capability is still increasing and price is decreasing. This makes a hardware implementation of the proposed concept possible, once the application has been fully worked out, increasing speed for a relatively low cost.

In the introductory chapter a definition of visual inspection was provided. It discussed the differences to related fields, provided the general motivation for inspection, its constraints and problems. The discussion enabled us to outline the motivation for this thesis. The chapter is concluded with a brief overview of the scheme that will be elaborated in the following chapters. In Chapter 2 developments in automated visual inspection made until now, as well as their concepts and algorithms are outlined. This chapter is intended for readers who are not familiar with the general concepts and tasks of visual inspection. The actual inspection concept and its application is explained in the following 3 chapters, organized from a high to a low level of generality, schematically shown in Figure 1.2. The global concept and the separation of detection and analysis are explained in Chapter 3 on the highest level of abstraction. Following a case study of the concept for different types of analogue display instruments in Chapter 4, the specific application of the concept for the calibration of watermeters and the corresponding results are shown in Chapter 5. This chapter is intended for readers who are mostly interested (also from the system engineering point of view) in the specific, successful industrial application of calibrating watermeters. The thesis concludes with a summary and discussion of the concept and the work in progress is outlined.

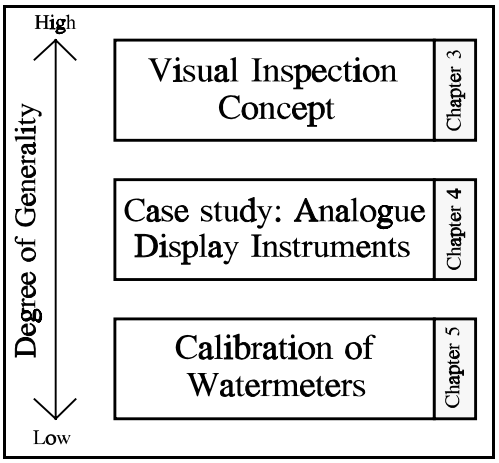


Figure 1.2 Levels of generality of chapters

Chapter 2

State of the Art of Automated Visual Inspection

In this chapter basic definitions, common approaches and techniques to model-based AVI are surveyed. After discussing basic structures and terms used in this field of research, the general model-based inspection scheme is examined (Section 2.2). Next, AVI systems that have been reported in the literature in the last 30 years are surveyed, classified and compared based on their sensory input in Section 2.3. Following an overview on inspection algorithms in Section 2.4, benefits and limitations of current inspection systems are summarized.

2.1 Introduction

Visual inspection of an object may be simply formulated as follows [HED89]: "*Given an object, determine if the object satisfies given conditions of acceptability*". This simply defined problem of inspection is performed at different automation levels, from manual to fully automated. Figure 2.1 shows an abstract version of an AVI system which consists of:

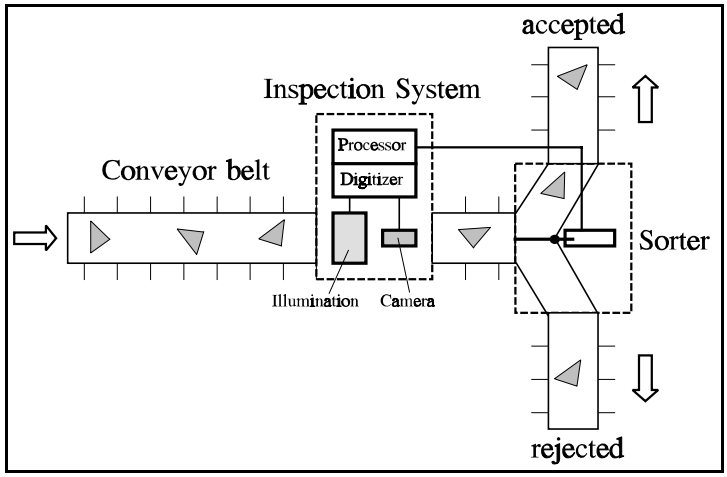


Figure 2.1 Diagram of an automated visual inspection system

- *Transport system*: The transport system, for instance a conveyor belt, moves the object to be inspected to the inspection station.

- *Inspection system*: A visual system, usually consisting of an illumination device - camera - digitizer - processor system analyses the object if it is within the specifications. The processing includes the sensing of the visual data, digitizing, image processing to enhance relevant features, and matching of the detected features with the specified features.

- *Sorter*: Following the matching a decision is made whether the object has passed the inspection or not and the processor directs the sorter to accept or to reject the object.

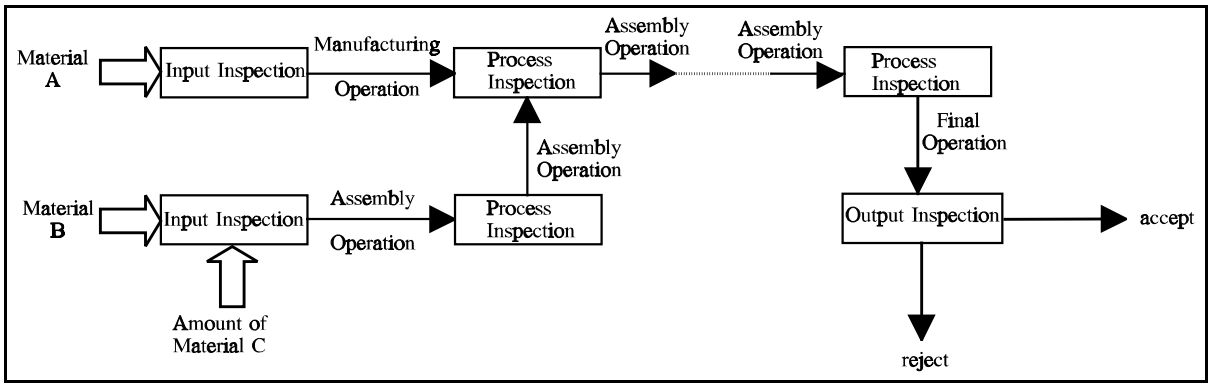


Figure 2.2 Inspection stages in a production line

There are three different types of inspection (see Figure 2.2):

- *Input (receiving) inspection*: refers to testing incoming materials whether their quality is acceptable for further use.
- *Process inspection*: is the intermediate control of the output between two working stages, controlling the product as well as the production machines.
- *Output inspection*: is the final quality control of products finished in an assembly or manufacturing stage to determine if the product has reached a predefined quality standard.

2.2 Model Based Visual Inspection

A priori knowledge is used implicitly or explicitly by all visual inspection systems, since inspection can only be performed by matching the *object* under inspection with a set of predefined conditions of acceptability. These a priori known specifications are described by an explicit *object model* (a system of assumptions, data, and inferences, that describe the object), where all relevant *object features* are described.

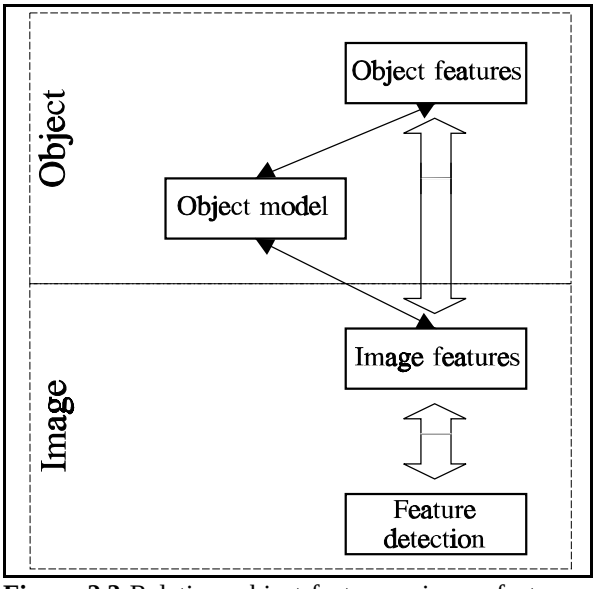


Figure 2.3 Relation: object features - image features

The second representation of the object is the *image* containing the object (the term image describes the image of the object under inspection throughout the thesis). Within the image, object features are represented as image *features*, that have to be detected by *feature detection* algorithms. Figure 2.3 shows the relation between feature detection, image features and object features schematically, the object model is used for representing inspection-relevant data.

A typical method to build the object model (also called description) consists of defining the geometric features of the object. A circular object, as a simple example, would be described by a circle with given radius, center and tolerances. This simple object model would then be used for verifying if an image has the predefined feature within the tolerances.

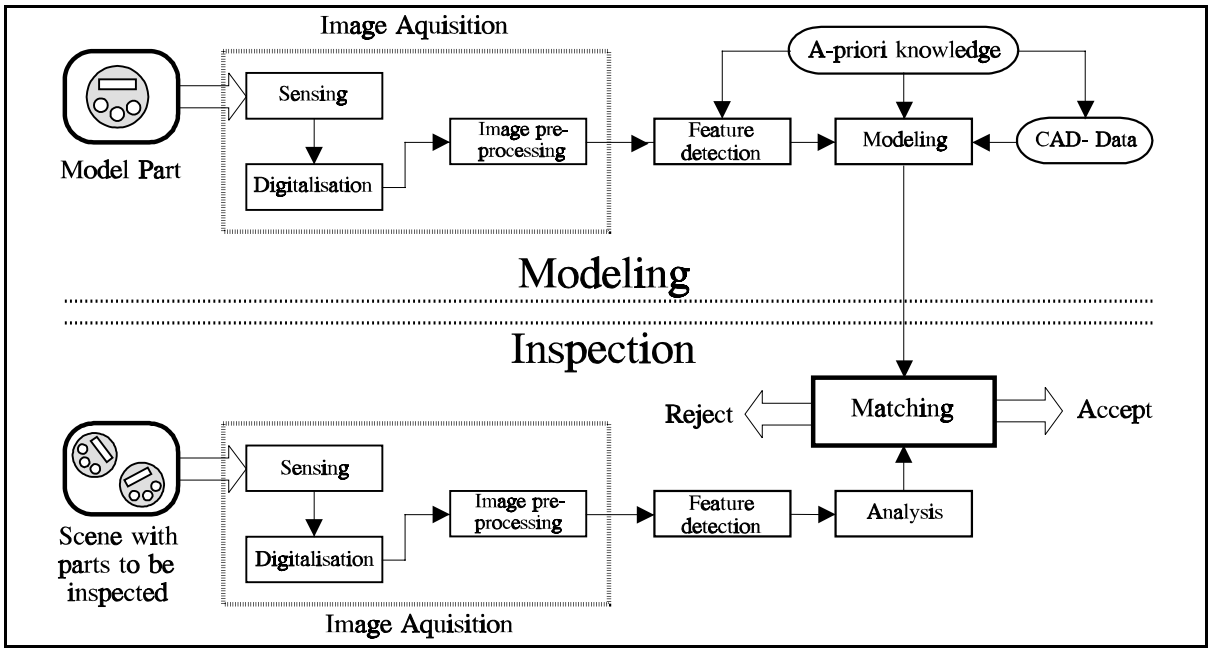


Figure 2.4 Components of a visual inspection system

In general, the design of an AVI system is divided into two stages: the *modelling* or training phase and the *inspection* phase as shown in Figure 2.4. Thus, the major components of an AVI system are:

- *Image acquisition:* consists of sensing, digitalization and image preprocessing, providing the spatial representation of the object.

- *Feature detection:* Is the process by which an initial image is transformed to features, containing a high amount of information to be used for inspection.

- *Modeling:* Selected features together with predefined specifications are used to generate an explicit object model, represented in a model data structure.

- *Analysis:* In the inspection stage (in contrast to modeling) the detected features are selected by a filtering process to reduce the feature set. A feature is said to be detected if a certain decision rule is able to assign it as belonging to a given subset of object features.

- *Matching*: Inspection proceeds by matching the data structure inferred from the observed image to the model data structure. Usually this step is not separated from the analysis.

This section gives an introduction into common issues of model-based AVI, according to the scheme shown in Figure 2.4, image acquisition, feature definition, the resulting models, and how the defined features are detected for the inspection procedure.

2.2.1 Image Acquisition

One of the most important aspects in AVI systems is image acquisition, because the image quality strongly influences the quality of the result. Many people say, a vision system consists only of: computer + framegrabber + camera + software. However, this neglects important issues: lighting, optics, systems integration, and standard industrial inspection practice [BAT85]. However, all parts of a properly designed AVI system bear an equal strain (in a formal sense). A particularly common error is the tendency to concentrate on the image processing to the detriment of the image acquisition (i.e. pose of the object being inspected, lighting, optics and sensor) [BAT94]. "*If it matters that we use the Sobel edge detector rather than the Roberts operator, then there is something fundamentally wrong, probably the lighting*" [BAT95]. This remark is not about the relative merits of the various detection operators but is a statement about the need for a broader "systems" approach.

A well designed sensing system reduces noise, prevents blur, stops object motion, optimizes the contrast between the part and the background for instance, has a resolution that ensures defect detection in the desired size and emphasizes all features relevant for inspection. Therefore, image acquisition involves the design and placement of the illumination device, the choice and placement of sensors and optics and the choice of digitizers and image processing algorithms to provide high quality basis data of the part under examination.

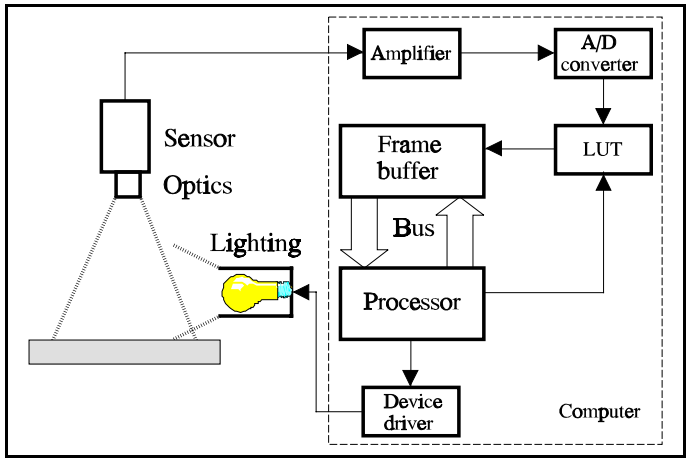


Figure 2.5 Schematic visual inspection hardware

Figure 2.5 shows a schematic diagram of the main hardware components used by a typical AVI system [TOD88a], which consist of:

- *Sensor*: (for instance a CCD camera) converts the radiated (light) energy from the scene passing through the *optics* into electrical signals.

- *Digitizer* (or A/D converter, often called frame grabber): converts electrical signals into a series of digital values representing the voltage or levels of the sensor corresponding to the original scene intensity levels.

- *Frame buffer*: consists of a high speed memory and stores the digital signals coming from the digitizer (usually a component of the digitizer or frame grabber).

- *Computer* (host): is able to access the frame buffer and processes the digital data in the image pre-processing stage (note that there is special image processing hardware available; in this case this image processing hardware is considered to be the host computer of the digitizer) as well as for the visual inspection process.

The choice of illumination devices, sensors, optics and digitizers is highly dependent on the spectrum of radiation of the object, the speed of motion, the minimum object feature size, the shape of the object, the shape and nature of the object features and other application specific characteristics. Due to the large amount of different parameters and imaging problems, the image acquisition design is an experimental effort rather than a quantitative design [CHI88]. Sensors can be in the form of CCD cameras, ultrasound or X-ray cameras built in point, line or array format with or without a high speed shutter in combination with specific illumination like stroboscopic light, tungsten filament lamps, fluorescent tubes, fibre optic light guides, quartz halogen lamps etc., in different resolutions. However, there is literature reporting experiences with problems and feasible solutions for the image acquisition problem (see, for example [MIS83,DOW86,BAT85,COW89,BAT95]). They state that the success of an inspection system is critically dependent upon the correct engineering of the front end component illumination, presentation device and image capture hardware.

2.2.2 Inspection Features

Before an inspection can take place, it must be determined which object features should be taken as *inspection features* represented as features in the input images. There are two major reasons for using features rather than computing the original data set. Features are less sensitive with respect of the encountered variations of the original noisy gray-scale or color images. Secondly, they encode the information contained in the image in a form more economical than that in which the information impinges on the sensors [ATT54], while preserving the information required for inspection [CHI86]. When a manual inspection process is automated, inspection features are known by the human operator a priori, although the decision which should be used for inspection is subjective.

Generally, object features can be subdivided into two classes:

- *Geometric features (or shape characteristics)*: corners, holes, lines, curves, and the like.
- *Properties*: dimensions, orientation, location, reflectance properties, color, and the like.

Once inspection features are determined, they are used to generate the object model - the main task of the modeling phase, called *inspection model generation*. Figure 2.6 schematically shows an object with inspection features and the corresponding inspection model, represented in a model-graph. The nodes of the graph contain descriptions of the geometric features and the arcs describe properties.

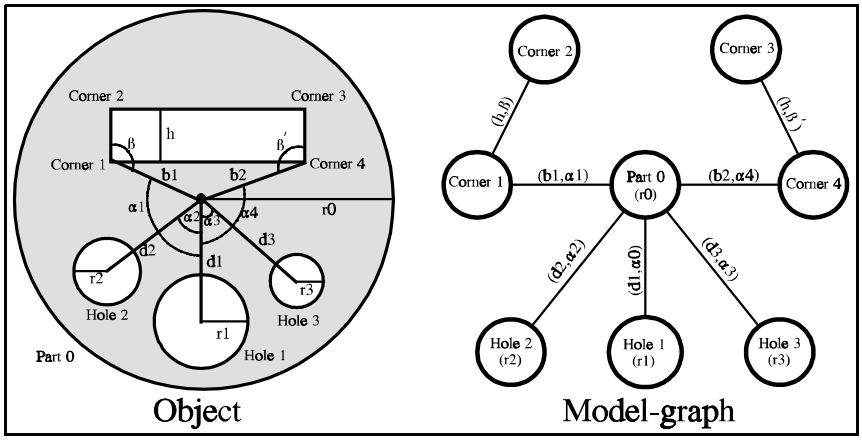


Figure 2.6 Object and its model based on features

There are innumerable techniques for feature detection, ranging from simple binary image segmentation by thresholding to edge detection and line-tracing to more sophisticated algorithms capable of extracting features from complex industrial scenes considering physical surface properties such as reflectance, surface orientation and complex textures as well.

2.2.3 Modeling

The object model is an organized representation of object features from which all information needed to perform the inspection are provided [CHI86]. Usually inspection models (object models used for inspection) are based on geometric features. Therefore, the extraction of the object information, which defines the structure of the object in terms of geometric features and their spatial relationships generates the inspection model.

In the example shown in Figure 2.6 an object shown on the left should be inspected if the 3 drilled holes and the punched rectangle are placed correctly within certain, given tolerances. One inspection model could be generalized as a graph structure shown in Figure 2.6 on the right. In this example, the 3 holes are modeled as geometric features with a given radius *r1,r2,r3*, the rectangle is represented by its corners (*corner1*, ..., *corner4*) inside the circular object with the given radius *r0*. The center of the object is taken as the geometrical reference point (origin), the direction origin - center of hole1, the reference direction $\alpha 0$. These two reference parameters represent properties modelled by the arcs, the relation *(b1,ß)* for instance, describes the position of the lower left corner of the rectangle, the distance to the origin is *b1*, the direction (in relation of the reference direction $\alpha 0$) is *ß*. Further a priori information like the dimensional tolerances can also be part of the model.

The inspection model generation is usually an interactive procedure, where two different ways of creating the model to start with are used:

- *Training by learning* (or also *self-training*): determines the global appearance of acceptable parts and stores this information in a reference model. The parts used for training by learning should be good parts (so-called "*golden parts*"), and, ideally, they should exhibit the entire range of acceptable process variations. Since a human operator is expected to select the training parts, the self-training is supervised. A number of self-training inspection systems have been proposed or prototyped (e.g., based on neural networks [NA95,SAN95] or principal components decomposition [MUR95])

- *Training by showing* (or also *teach mode*): the user works with a reference image and teaches the system interactively the object features to be examined, their parameters, properties, and acceptable tolerances. Following the inspection feature definition, an interactive "*fine-tuning*" phase is carried out, all parameters are checked once again and corrected if necessary.

A simple example of an interactive training-by-showing process is shown in Figure 2.7. The operator works interactively and selects windows containing inspection feature regions. The rest of the image is regarded as background without information for the inspection model generation. In this example, the operator has chosen 4 circles, one rectangle and 4 lines to be the inspection features. After this step the system will ask the operator to specify all the re-

maining parameters and properties, including parameters not contained in the image like name, tolerances and the like. After this interaction the system will define the inspection model and encode it in data structures.

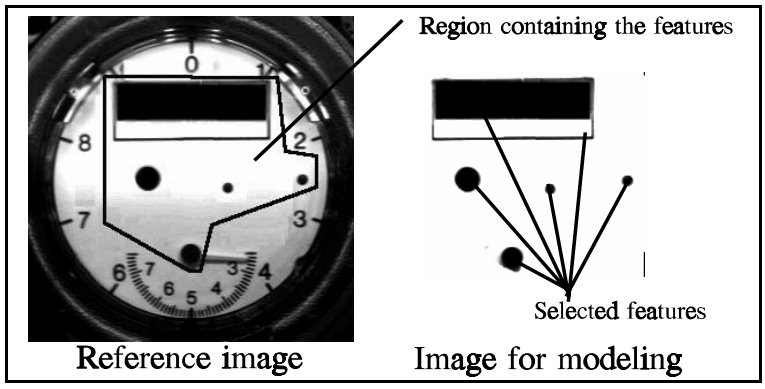


Figure 2.7 Example of interactive training-by-showing

For reasons of speed and robustness, most of the industrial inspection applications involve the inspection of planar patterns of 3-d objects [CHI92]. A few objects (like circuit boards, labels on packages, IC chip patterns, web textiles, and the like) are structurally 2-d (up to a certain degree). However, a large number of industrial parts are of 3-d shape. In this case, the inspection requires the construction of a 3-d model by:

- *Characteristic views*: the 2-d projections of all the possible stable positions of the part with respect to the position of the sensor are computed [FRE80]. Stable in this context means that the vantage point can move around in a neighborhood without changing the structure of the corresponding aspect graph [WAN90]. The term aspect graph [BOW91] has come to be used to refer to a variety of related representations in which the partition of viewpoint space into cells of general viewpoint is used to generate a dual representation that takes the form of a graph [BOW94].

- *Reference depth-maps*: reference data are produced with the help of a 3-d acquisition system (see for instance [AGG89,BES89,POU89,NAL93] for comprehensive summaries of range finding techniques).

- *CAD-data*: if Computer Aided Design (CAD) data of the object are available, a mathematical description of the shape of the object, including an explicit parameterization of surface shape and an explicit encoding of intersurface relationships, is provided. Often, CAD databases are also augmented with manufacturing information, including material type, geometric tolerances, desired quality of surface finish and other information, thus providing a unified description applicable for inspection.

2.2.4 Feature Matching

The decision whether the object can be accepted or not, is made by matching the detected features with the object features described in the inspection model. There are several techniques for doing this matching. Some matching methods use the total image by subtracting the actual image from the reference image. Others use similarity measures (such as correlation or cross-correlation), applied to the image and reference intensity values or coefficients of some mathematical transforms (such as the Fourier transform), computed to determine the similarity between the actual object and the inspection model. In all of these cases the inspection model is the simplest one which can be chosen, namely the image of a "golden" object or a trans-

formed version of it and the detection is nothing but a comparison of images. However, if the image is noisy and the object is not in its supposed position the detection will fail and therefore these algorithms are only effective under certain constraints.

One of the main problems in the matching stage is the precise registration between the object and the underlying model. Therefore, many AVI systems use a mechanical device to register the object or by positioning the camera or the object electronically [NEW95].

If inspection features and properties are used to construct the model, more sophisticated detection techniques have to be used to inspect the object. Basically, feature based matching tries to locate all inspection features. For example, the inspection of the object shown in Figure 2.6 involves locating the features in the image that match with the nodes defined in the object-graph. Size and position of the features are thus compared with the nodes and the relations defined in the object graph. If all measurements correspond to a certain measure with the model, the object passes the inspection. Feature based AVI systems are invariant to translation and rotation and are not very sensitive to noise and image distortion [CHI92]. They depend on the quality of the algorithms used to detect the relevant features, their processing order, and the quality of the inspection model.

2.3 Automatic Visual Inspection Systems

Many different AVI systems have been reported in the literature. Chin and Harlow have surveyed the first commercially available systems from the beginnings to 1980 in their survey [CHI82]; then Chin again presented a survey from 1981 to 1987 [CHI88] and the recent developments in the area of AVI systems were surveyed by Newman and Jain [NEW95]. These three papers demonstrate that the complete inspection literature is vast, appearing not only in computer vision conference proceedings and journals, but also in the robotics, manufacturing and recently also in the CAD literature. Many of the installed AVI systems have not been reported in the literature, due to proprietary considerations. Therefore, it is not easy to survey the current state of the art of inspection systems.

To categorize different inspection domains and applications we choose the type of input data they use as the classification criteria. Note that there is no distinction made between matrix- and line-sensors, since their output differs only in the image size and not in data type. If objects to be inspected move, line scan devices are used, if not matrix devices are used.

2.3.1 Inspection using Binary Images

Binary images are sufficient to inspect silhouettes of industrial objects (like flat objects without surface characteristics). Figure 2.9 shows an example of a floppy disk drive casing to be inspected. These systems usually perform only simple coarse verifications, such as testing for the presence of an object. Binary images have the advantage that they can be acquired by inexpensive sensors in conjunction with simple lighting arrangements,

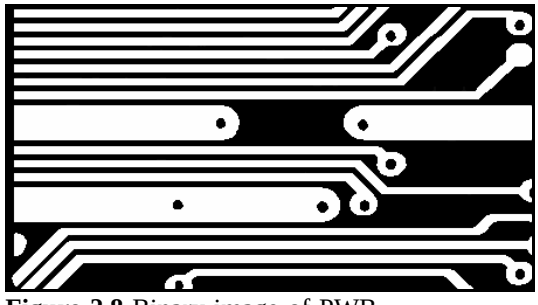


Figure 2.8 Binary image of PWB

which eliminate shadows and highlights for the purpose of binary representation. The lighting constraint is the main reason why this technique is only usable for flat objects, since all other objects will introduce severe lighting problems due to shadows and highlights. Therefore, this method cannot be used if surface reflectance, texture or surface orientation information is needed.

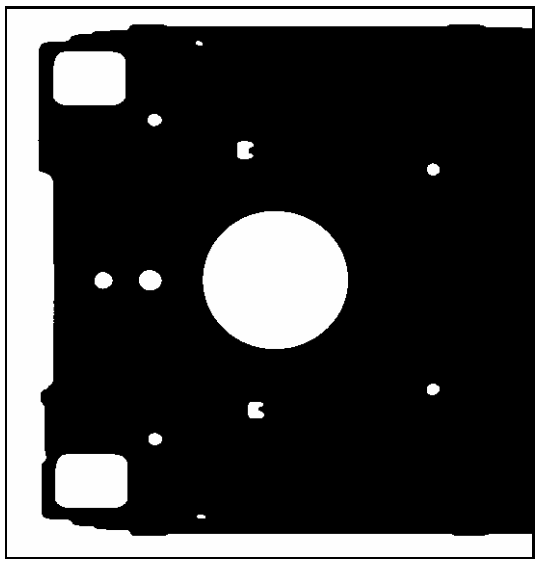


Figure 2.9 Silhouette of floppy disk drive casing

Binary images are used in the inspection of *Printed Wiring Boards* (PWB's), since the presoldered bare boards have only two types of surface, metal and substrate; both surfaces are flat and the inspection does not require surface reflectance or surface orientation information. Figure 2.8 shows a thresholded binary image of a PWB board detail. Binary AVI systems use only simple detection schemes such as pixel counting or boundary examination, which can be executed very rapidly [HUN85]. A simple thresholding scheme reduces the gray level PWB image to a binary one, or this conversion is performed by the sensor itself, providing a variable, controllable threshold.

2.3.2 Inspection Using Gray-Level Images

At present there are only a few systems capable of performing visual inspection on gray-level (intensity) images in complex industrial environments with dirt and unfavorable lighting conditions [CHI92]. To adhere to the speed constraint these systems use special hardware for the image processing and the defect detection, thus increasing the hardware costs. For dimensional verification new systems are in development, since hardware costs are decreasing.

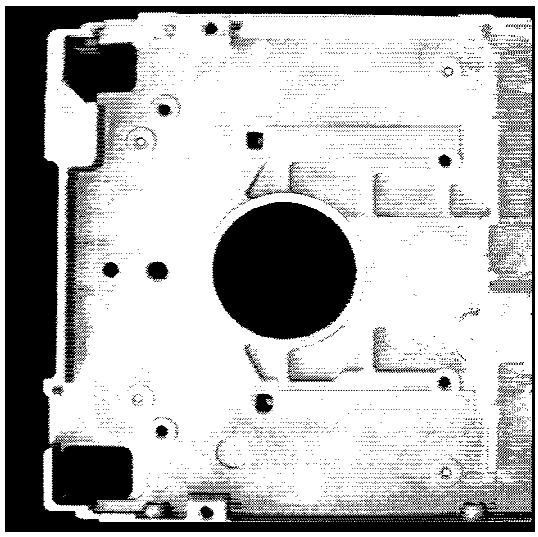


Figure 2.10 Graylevel image of casing

Some inspection tasks can only be performed using gray-level images since they are based on inspecting surface properties of objects like glass panels, coated sheet steel, bottoms of bottles, and thin film heads of magnetic disks. Furthermore, agricultural inspection applications like fish quality control, where black skin and blood spots should be detected, chicken grading systems, and the detection of watercore damage to apples were developed. Figure 2.10 shows the gray-level image of the casing in Figure 2.9 as example.

Most of the applications using gray level images are reported in the field of the inspection of electronical devices, especially the inspection of solder joints, since they are volumetric objects (see Figure 2.11). Therefore, binary processing is not adequate for the description of the

3-d surface of the joint. The surface geometry is used as inspection criterion, associating solder joint features with brightness, size, volume, surface area and surface curvature. Statistical decision rules are then used to classify joints into various types of defects. This method is based on the assumption that the physical solder joint surface can be approximated by the surface reflectance, if fixed illumination is used.

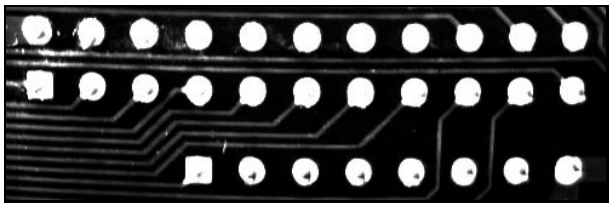


Figure 2.11 Graylevel image of solder joints

Texture inspection can only be done with the help of gray-level or color images. In industry several textures need to be inspected prior to the delivery of the product, especially if inspection is the last step before selling it to the consumer. Therefore, the packaging of products like cigarettes and videocassetes is inspected before leaving the factory. Other products like silk, woven textile, carpet wear and lace produced by the textile industry are also inspected. Many techniques have been presented for textured surface inspection, a survey of techniques can be found in a paper by Tuceryan and Jain [TUC93].

2.3.3 Inspection Using Color Images

Recently a few AVI systems using color images have been developed. They are currently in an experimental stage, since they need high computing power (an acceptable color image is three times the size of a gray level image), intricate optics and controlled lighting. The lighting problem is much more crucial than in the binary or gray level case, since factors such as light source intensity, color temperature, angle of illumination, magnification, and lens aperture are crucial for accurate color monitoring [DAL91]. The increased sensitivity of the inspection system to the energy distribution of the light source and the surface of the object has to be handled carefully, any ambient or scattered light near the inspection system has to be avoided. Nevertheless, color is an important feature in apparel and food inspection, where changes in shade or variations in color indicate a defect. Humans are not reliable color inspectors, since they do not have a consistent color memory, making color inspection tasks like inspection of painted surfaces or wood [KAU95] appropriate for automated systems.

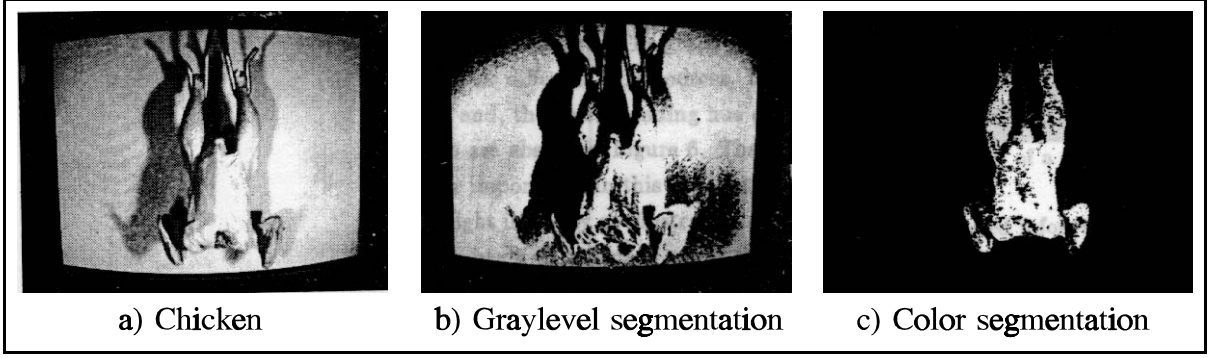


Figure 2.12 Color used for segmentation of chicken for size determination (from [CHA90])

Some of the applications of inspection systems using colors are the inspection of apples, tomatoes, oranges, lemons and the like, which are classified due to their size, color, ripeness and blemishes, defining a certain quality. In Figure 2.12 an example for the use of color-

images in food inspection is shown [CHA90]. The inspection task consists in the size determination of chicken (Figure 2.12a) prior to the packaging process. The problem is solved by using color segmentation (Figure 2.12b) instead of gray-level segmentation (Figure 2.12c), since chickens have a unique color. The packaging industry uses color images to check the correct color of printed packages like cigarette packages or batteries. Due to the drawbacks of speed and lighting these inspection systems are not widely used.

2.3.4 Inspection Using X-ray Images

To inspect industrial parts with internal structure, radiography analysis methods are used. Principal applications of this method are the inspection of weldings, machined parts, and PWB's [ROD95]. Drill holes in metal are also inspected using X-ray images although the images are typically not of high contrast but exhibit subtle shades of gray [HED89]. Another application of X-ray based inspection is the packaging of foodstuffs. Figure 2.13 shows a X-ray image of food packages with alien elements. There, foreign bodies in the food can be detected if the density of the foreign material is higher or lower than the sample density of food. Cost and safety considerations, however, play an important role in the use of this method. For cost reasons, a potential use of nuclear magnetic resonance (NMR) imaging, which is used to "inspect" human organs, is not taken under consideration for the inspection of the texture and composition of food. There are a few other applications reported using gamma rays, ultrasound, near infrared [YAN95], infrared, and ultraviolet techniques, all of them being in an experimental stage rather than in a practical application.

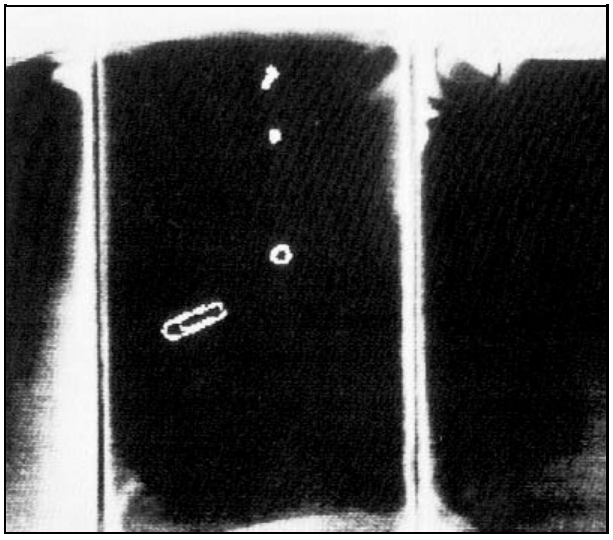


Figure 2.13 X-ray image of food packages with foreign elements (from [AHL91])

2.3.5 Inspection Using Range Images

All the previously described methods project 3-d objects onto 2-d images, which is for some applications impractical. Especially if object surfaces should be measured rather than any defect classification of the surface performed. The silhouette of turbine blades for instance provides negligible information about the surface of the blade and the intensity data can contain variations of reflected light which are not due to a surface variation. Therefore, the inspection of a curved surface profile is a task which requires the sensing of the surface contour and the measurement of the profile parameters. The goal of this task is to derive surface information, a so-called 2½d-surface representation. From this representation, profile measurements such as curvature, surface range, orientation, surface normals etc. are extracted. Defects are detected by examining these parameters; a dent on a surface, for example, will show up in a

change of range. The greatest advantage in using range data is the explicit 3-d surface information representation [JAI90].

There are different range acquisition techniques used in applications [BES89,POU89]. Examples for objects that are inspected using range images are propellers, engine castings, bolts, engine valves, and other automobile parts. There are also solder joint inspection systems using 3-d data because the joints and their key process indicators are complex geometric shapes [NEW95]. The amount of solder per joint can therefore only be determined using 3-d information. Therefore, for particular applications the use of range data in inspection seems to be the only solution although there seem to be no complete 3-d industrial inspection systems for complex objects which have been placed online so far [CHI92].

2.3.6 Comparison Application Domain - Data

The previous sections showed that there are at least 5 different major data types used in AVI, depending on the application and the inspection task the system should solve. Therefore, it is hard to decide which type of data should be used for a specific inspection task, since it strongly depends on the constraints given by the environment, lighting conditions, imaging constraints, inspection speed, development cost and the like. However, due to the development for more than 35 years, commonly used data types can be identified for specific applications listed in Table 2.1. In relation to the data, different inspection algorithms have to be used, where the amount of data per object influences the complexity of the algorithm, ranging from simple algorithms in the binary data computation to complex, time consuming strategies in color and range image computing. A comparison of characteristics of different data types is shown in Table 2.2.

Application type	Example	Data type
Electric/ Electronic	Printed Wiring Boards, IC chip patterns (photomasks), CRT-photomasks	Binary, Intensity, X-Ray, Range
	Solder joints	Intensity, X-Ray, Range
	IC wafer, semiconductors, thin film heads of magnetic disks	Intensity
	Color CRT displays	Color
Metalwork	Casings, drill holes	Binary, Intensity, Range
	Bolts, screws, weldings, machine parts, aluminum and railcar wheels, Automobile parts (bumpers, body, motorcastings, etc.)	Binary, X-Ray, Range
	(Hot) steel and aluminum	Intensity
	Ship propeller, turbines, tubes	Range
Glass & plastic	Bottle caps	Intensity, Range
	CRT faceplates, glass panels, bottles, lamps	Intensity

Food	Industrial food (instant meals), Agricultural products, meat, fruits	Intensity, Color
	Industrial food, food packages	X-ray
Letterings	Labels and packaging	Binary, Intensity
	Printings, painted surfaces	Intensity, Color
Miscellaneous	Web, woven textiles, apparel, Wood, lumber, kitchen cabinets	Intensity, Color
	Break shoes	Binary
	Color CRT displays	Color

Table 2.1: Different data used for different applications

Input data	**Binary**	**Intensity**	**Color**	**X-ray**	**Range**
Amount of data:	Small	Medium	Large	Medium - large	Large
Bit/pixel:	1	typ. 8	typ. 24	typ. 8-16	typ. 24
Algorithms used	Simple	Sophisticated	Complex	Sophisticated	Sophisticated - complex (hardware dependent)
Speed	High, real time without special hardware	Slow, due to time consuming algorithms	Very slow	High, real time possible	High - extremely slow, depends on RF technique
Lighting	Simple, diffuse or polarized backlighting	Difficult, optics very important	Very difficult, color temperature very important, no ambient light	None	None - very difficult, depends on RF technique
Surface reflectance/ texture	No	Good	Very good	No	No - good, depends on RF technique
Hardware costs	Inexpensive	Moderate, special (expensive) hardware for real time applications	High, exact calibration necessary	Very high	High - very high, software cost high
Miscellaneous	Only flat objects, Exact orientation necessary, For simple tasks only	No orientation needed	Inspection better than human inspection, Color classification possible	Complex and closed objects inspectable, High contrast images, Safety risks	Surface orientation, curvature, range, surface normals, High accuracy

Table 2.2: Comparison of different data types: characteristics

2.4 Automatic Visual Inspection Algorithms

Beside the classification of inspection systems on the basis of data type, they can also be classified by the algorithms they use to perform the inspection. The first approach (also in terms of historical use) called *Template Matching*, involves the matching of an area (called "*window*" or "*template*") of a reference model to the sensed data of the object. This reference model can either contain defect-free, "*golden*" reference data or reference data of defects. The better the actual data of the object match the model, the higher is the probability that the object under consideration is defect-free or has a defect, if defect reference data are used.

The second approach extracts features from the sensed object and matches them to the object features in a description or a list of rules that describe the object. If all rules are satisfied or if the actual features match with the description, the object is considered to be defect-free. If the nature of defects can be modeled easily, a defect feature model is used, detected features are matched with the description or the rules of the defects and if there is a correct match, a defect has been detected. The nature of the algorithm remains the same.

2.4.1 Template Matching Methods

There are two approaches to template matching according the data used in templates:

- *Area-based matching*: uses binary and gray level images to perform inspection. They try to compare the value of pixels in the actual image against the pixels in a reference image, called pixel-by-pixel matching.

- *Feature-based matching:* instead of a comparison of binary- or intensity-data, features in the image are compared with object features of the inspection model. This approach compresses the image data and reduces the sensitivity to intensity changes.

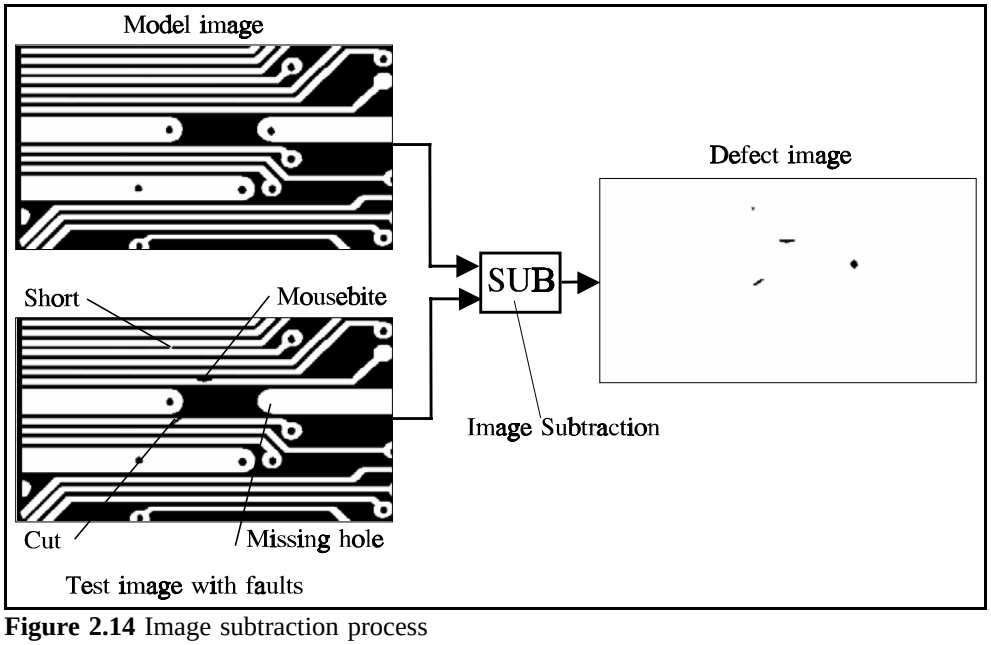


Figure 2.14 Image subtraction process

The reference image can either be a sensed image or a synthetically generated one, for instance from a CAD model. It contains either a golden or a defect model of the object in the form of a binary- or intensity image or a feature image.

The simplest example of template matching is image subtraction, where the matching consists in subtracting the sensed image pixel-by-pixel from the reference image, the template is the whole image. If the image is the same as the model, the resulting image will have a uniform grey scale. Differences either indicate defects or defect-free objects, depending on the reference model. This matching scheme can also be seen as model-based, the "object feature" would be an intensity function of the object at all locations; the "model" is another intensity function of the object. The "detection" is a subtraction operation performed at all locations. The subtracted image can be used to perform an error classification.

Figure 2.14 shows a simple example for an image subtraction process, the binary test image of a PWB with 4 defects (short - two wires are interconnected; mousebite - one wire has missing parts; cut - one wire has a break and is not connected to the rest; missing hole - a drilling procedure did not take place, all 4 defects are very common in PWB production) is subtracted from the binary model image and results in a defect image containing the four defects. This defect image can be taken either to classify the defects or to reject the object under inspection without a defect classification.

Beside image subtraction other pixel by pixel functions like absolute difference, multiplication, logical operations, and others [BAI83] can be used to compute the similarity between the actual- and the reference image. Furthermore, statistical similarity measures like energy, entropy, and correlation (see for instance [HAR86] for a survey on similarity measures) are also used to match the actual template to the reference template.

An improved template matching method uses instead of the complete image only templates (or windows, small parts of the complete image), containing perfect object features or defects. The image of the part under inspection is correlated with an ideal "mask" or "template", a high similarity is given at the locations where the image is like the template. To detect different object features or defects in different orientations and size, different masks have to be used. The matching process consists of finding the defined pattern in the given image. A perfect match would be rarely matched exactly due to image noise and quantization errors. Therefore, the matching is done by computing a similarity measure between the template and the pattern in the image. Whenever the similarity measure is higher than a certain, predefined threshold, the object feature or defect is supposed to be found in the image.

Feature-based template matching uses the same strategy as area based template matching, the reference template consists of a feature image. Figure 2.15 shows a simple example for the feature (pattern) matching technique. Suppose the object under inspection should have 3 holes, two of them of equal size. The features to be searched for are two circles with a given radius. These circles are represented by a binary pattern containing a circle with the two radii, called search pattern. The search pattern is moved in the edge image and every position in the image is checked if it matches the search pattern. In the example in Figure 2.15, only two circles are found because the rightmost hole is too small, and the object would be rejected.

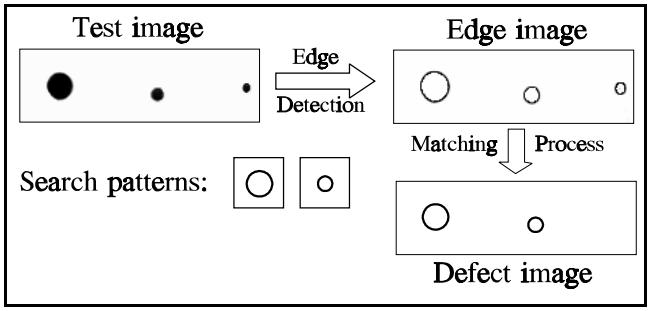


Figure 2.15 Pattern matching technique

Template matching techniques are frequently used in the inspection of printed circuit boards [SIL89]. A simple algorithm and high speed are the benefit of these methods, but various drawbacks can be found. Usually a large data storage is needed for the model as well as very precise alignment, since small shifts or rotations of the sensed image will cause wrong results. Figure 2.16 shows an example of how a shift of two pixel in *x* and *y* direction will influence the result. The actual image of a casing part, shown in Figure 2.16b, is subtracted from the model image, shown in Figure 2.16a. If the two images are correctly aligned, the result in the defect image would be the missing hole (Figure 2.16c). If the input image is shifted, the result (Figure 2.16d) is hard to interpret. The existing defect has to be looked for in the defect image since other "defects" produced by the shift are also present.

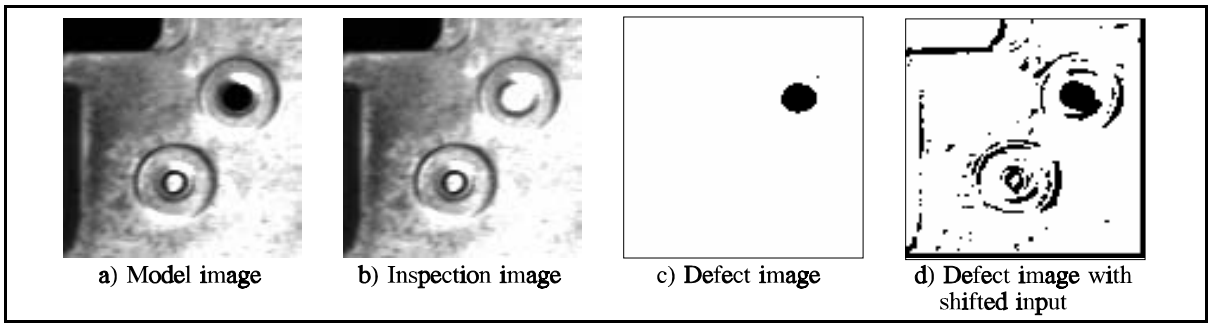


Figure 2.16 Example for image subtraction with shifted input image

Furthermore, other complications in template matching approaches arise because of the nature of the objects to be inspected: there are natural (nondefective) variations between the objects in most manufacturing processes [NOB93]. In PWB inspection it is the case that boards do not match point-to-point to the model because of shrinking or swelling of the board although they are acceptable for further use. In addition, variations in lighting during inspection will generate undesirable effects on the matching, similar to those generated because of product variation. There are several algorithms to overcome difficulties like image subtraction via pseudomodelling [CHI86], learning algorithm based on statistical pattern recognition, and least square fitting approaches, to name a few. However, comparing low-level pixel values utilizes only a small portion of the potential information in a given image that could be used [SIL89]. If the pose (orientation and position) or scale of the object to be matched is unknown, template matching is very slow and not efficient.

2.4.2 Rule-based Methods

Rule-based methods consider a pattern defective if the detected features do not conform with the rules specified for object features in the design. They make decisions about the object quality according to the classification rules, fixed in the design phase [PET87]. The sensed object is checked against a list of design rules or against features that can be extracted from the design rules. Thus, there is no need for direct comparison between the actual image pattern and a reference pattern on a point to point basis. Therefore, there is no need for an extensive database containing the templates and no necessity for the images to be precisely aligned. Instead, these methods try to imitate human inspectors by using features rather than intensity values. Human inspectors do not memorize a perfect version of the product being inspected. Instead, they are able to pick up the deviations from the characteristics of a good product and

mark them as defects [DAR88]. However, rule based methods suffer from the disadvantage that they are less adaptable to design changes, and from the difficulty and time-consuming nature of feature detection. Extraction of design rules from an object design model (for instance a CAD model) is not always a trivial matter, since they are not automatically derivable from databases. Almost all industrial inspection systems using rule based comparison schemes reported in the literature are designed to solve a specific inspection task and are not designed to be adaptable to solve other inspection problems.

2.4.3 Segmentation Algorithms

Following the pre-processing stage (i.e. noise reduction and image enhancement), the amount of image data is reduced by feature detection. Feature detection may consist of a division into significant regions (for instance to separate an object from the background), called segmentation. The objectives pursued in each application condition the kind of regions to be extracted and, as a result, the kind of technique to be used. For this reason no general segmentation theory exists, but a series of different segmentation techniques, each of them with benefits and drawbacks [HAR85]. Two major approaches to segmentation are distinguished [NEV86]:

- *Region-based methods*: Region-based methods try to find areas in the intensity image with homogeneous properties, which in turn give the boundaries.

- *Edge-based methods*: In edge-based methods, the local discontinuities are detected first (called edgel-detection, an abbreviation for edge-elements-detection [NAL93]), then they are connected to form longer, complete boundaries and finally the parametric description of these edges is formed (the parametrisation is application dependent, simple model based inspection systems operate on edges without parametrisation).

These two methods are complementary and the decision which of the methods should be used is application dependent. For a detailed discussion of segmentation techniques, the reader is referred to [ROS82,HAR85,DAL91,JAI90,TOR92].

2.4.4 Edge Detection and Operator Evaluation

Edge detection is based on the assumption that at an edge the intensity in an image changes in a discontinuous way. In two dimensions, the edges have a direction and a magnitude, and their intensity profile is assumed to be uniform along the edge [NEV86]. Edge detectors seek to verify the existence of short linear edge segments that are postulated to lie across image windows, that have the same shape and size as the operator kernel to detect them. The kernel (core) of an operator, is the part of the operator that makes use of the intensity of the image; the kernel of a convolution operator is the non-zero part of the operator [NAL93]. For a review of edge detectors the reader is referred to [PRA91,NAL93].

Since there are different edge detectors and edge detector schemes, several procedures for evaluating the quality and performance of the edge detectors have been developed. It is of interest to evaluate edge detectors both to compare one detector scheme with another, and to

study the behavior under different conditions and parameter settings [KIT81]. The most desirable properties of an edge detector can be formulated as: The detector should respond positively if an edge is present, and negatively if the edge is absent. Due to the presence of noise, however, edge detectors do not always behave in this desired way.

One criterium of the *performance* of an operator is the statistical maximum likelihood ratio test of the true positives or false negatives of the operator, that is, of its positive or negative response to the presence of the edge. This statistical test (see [ABD79] for details) is a function of threshold and of the signal-to-noise ratio if additive Gaussian noise is supposed. As a result, threshold selection and therefore the quality of the result is a tradeoff between the missing of valid edges and the creation of noise induced false edges. Thus, the evaluation of an operator must take into account two types of errors:

- *False negative* (type I error): the rejection of a correct hypothesis,
- *False positive* (type II error): the acceptance of an incorrect hypothesis.

An evaluation of different operators by this criterium is shown in [ABD79], resulting in a superiority of some detectors over others. For instance it is shown that extended mask size edge detectors are superior to 3x3 edge detectors only at low signal-to-noise ratios. This fact has to be taken into account if a decision has to be made which kind operator and which mask size should be chosen to solve a specific detection problem. Further parameters used for evaluation of edge detectors are:

- accuracy of the detected *orientation*
- accuracy of the *position* of the detected edge.

In the early work of Fram and Deutsch [FRA75] the effect of noise on various edge detectors was studied by evaluating two measures: One estimated which fraction of the estimated edge pixels were actually edge points characterizing the *freeness of noise* of the output of the operator. The average value of the second parameter characterized the distribution of the output over the length of the edge. The experimental results showed that the performance of edge detectors increases (measured by the two parameters) when the contrast is increased relative to noise. The accuracy of orientation of the output of the edge detector was also analyzed by [DEU78], demonstrating that some edge detection schemes perform consistently better than others concerning the accuracy of position and orientation of the output. Abdou and Pratt [ABD79] presented an analytic approach to the evaluation of edge detectors, evaluating a deterministic measurement of the edge gradient amplitude (claiming that the amplitude response should be invariant to edge orientation), comparison of the correct and false edge detection, and a figure of merit evaluation introduced by [PRA78].

All of the evaluation procedures mentioned above require a priori knowledge of the location of the actual edge, since they are based upon the discrepancy between the detected edge pixels and the ideal position of the edge. In non-synthetic images, the real position of the boundaries is usually not known, the determination of edges in such images is very much the subjective decision of a human observer, resulting in the fact that these techniques are completely inapplicable to images for which the actual edge positions are unknown. It is more appropriate to use an evaluation procedure that measures the intrinsic coherence of the detected edges, without using a priori information on the image [TOR92]. A method for evaluating the quality of edge detector output based on the local "good" form of the detected edges was pre-

sented in [KIT81]. Good form of an edge is defined in terms of continuation and thinness of an edge, claiming that these measures behave as one would like under the effect of change of threshold, noise and other operations. Since this method does not require knowledge of the true location of edges, it can be used to adjust parameters of detectors such as threshold, for optimum detection of edges in real images.

All the proposed evaluation techniques give similar results when applied to a comparison between different techniques and operators: The larger the dimensions of an operator, the more insensitive it is to noise. Almost all 3x3 operators have practically indistinguishable measurements of quality [TOR92].

2.5 Drawbacks of Traditional AVI

This chapter has summarized inspection techniques and algorithms classified by their sensory input. The methodologies of various inspection systems were discussed and commonly used techniques were highlighted. The systems and techniques examined here allow the discussion about limitations and weaknesses of present industrial inspection systems. The requirements of speed and accuracy are obtained by almost all of the inspection systems since they are installed in industrial environment. From the economic point of view only those systems are installed that have to ensure high product quality, due to security requirements. This fact enables companies to invest a great amount of money into the inspection system.

However, the set-up costs could be reduced if the constraint of flexibility were taken into account. In order to achieve the required processing speed (throughput rate must be within production flow) and accuracy, flexibility has to be sacrificed. This arises because fixed-function dedicated electronics is employed to perform image processing operations at an acceptable speed [BAT92]. Nearly all of the existing AVI systems, with the exception of very few experimental systems, are custom-engineered systems with only one specific task in mind [NEW95]. They therefore represent ad hoc approaches to a single problem but as more complex industrial parts should be handled, the inspection system has to be more flexible and versatile [CHI92]. To summarize, current industrial inspection systems have among others the following main drawbacks [NEW95]: They

- employ ad-hoc approaches and have limited application;
- are inflexible and non versatile;
- are unable to inspect free-form surfaces;
- require precise positioning;
- are unable to handle occlusions;
- use tolerances not defined explicitly;
- suffer from slow speed due to more sophisticated algorithms;
- have fairly high cost (hardware and software).

There is a strong need for so-called "general-purpose" inspection systems that can handle all of the mentioned criteria, but since they are difficult to design they induce high capital cost. However, these high costs can be justified in the long run by their utilization rate. The following chapter introduces a flexible visual inspection concept based on a systematic approach that is able to increase the degree of flexibility and thus decrease the set-up costs.

Chapter 3

A New Visual Inspection Concept

The major drawbacks of existing inspection systems are high set-up costs, resulting from extensive pre-inspection set-up by experienced operators, hard- and software development costs, labor, and maintenance costs. The solution to cost reduction is to increase the flexibility of AVI systems, which makes it possible to amortize the development costs by a high number of installed units for different applications. The AVI system design cost has already been reduced by the development of libraries of image processing algorithms (e.g. Matrox Image Library [MAT92]) or interactive image processing systems (like Khoros and Cantata [KHO95,YOU95], KBVision [AME95], and Matrox Inspector [MAT95]), which allow rapid prototyping and the re-use of algorithms used within these systems. However, it is unlikely that the common user of an inspection system will have the relevant image processing expertise to be able to set up an inspection system by himself. If image processing systems are to be adopted and used for inspection of a variety of applications, it is essential to reduce the expertise required in the configuration of the inspection system [BOD95].

One solution in designing a flexible visual inspection system lies in the separation of the application-independent feature *detection* from the application-dependent *analysis*, forming the model-based AVI system [SAB95e,SAB96c]. Figure 3.1 shows the *off-line* part of the proposed general inspection system concept discussed in this chapter (rectangles indicate processes, rounded rectangles data). This part is called *off-line* since it is not performed at the speed of the production line, there is no explicit time limit to complete the inspection system set-up (except economical set-up time limits). The generated executable inspection system is used on-line, i.e., it works within the production line flow. The off-line inspection system set-up consists of four general processes:

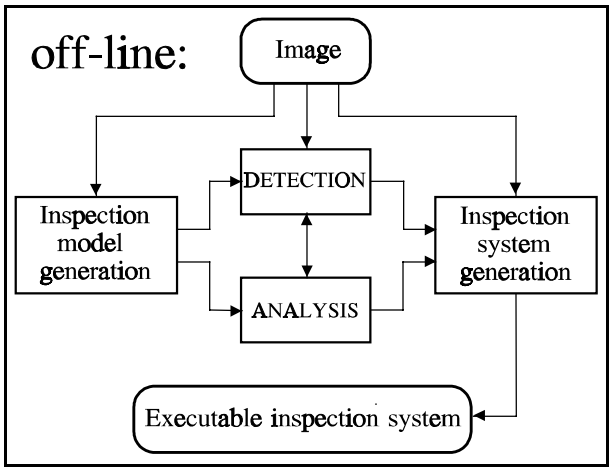


Figure 3.1 General inspection system concept (off-line)

- *Inspection model generation:* Inspection feature detection and possibly the use of an a priori model supplies the inspection model of the object.

- *Detection:* The detection algorithms (like edge detectors, edge linking algorithms, segmentation algorithms) to detect object features are selected in this stage. Existing, standard pattern recognition algorithms can be used, which means that no specific pattern recognition software has to be developed for a specific problem.

- *Analysis:* The parameters of the detection algorithms are adjusted, detection order and search space are selected in the analysis stage. Several algorithms are adapted, tested for the given inspection task and used if they achieve a certain pre-defined detection rate.

- *Inspection system generation*: The preliminary inspection system is tested. If it turns out that for instance a specific feature detection algorithm does not adhere to the desired recognition rate in the test set because of lighting conditions or alignment, it can be replaced by another algorithm which attains the recognition rate. Furthermore, industrial constraints are checked and balanced, before the final inspection system test takes place.

Since the analysis is separated from the detection, the concept allows a flexible adaptation of the inspection system too, because a change of the object layout results only in the requirement of a new description, while the detection remains the same. However, if new primitives are introduced or substantial layout changes occur, analysis and detection have to be re-adjusted. If the test confirms that the inspection system attains the intrinsic constraints of the inspection task, the executable inspection system for on-line inspection is ready.

Figure 3.2 shows the components of the on-line inspection system. The image taken under the same acquisition conditions as in the off-line set-up phase, is the input for the detection. The defined feature detection algorithms are used in the detection order given by the analysis and the generic parameters provided by the inspection model. In the analysis, the features are checked whether they are within the tolerances defined in the inspection model. The description of the inspection result is provided, which states why the object is accepted or rejected.

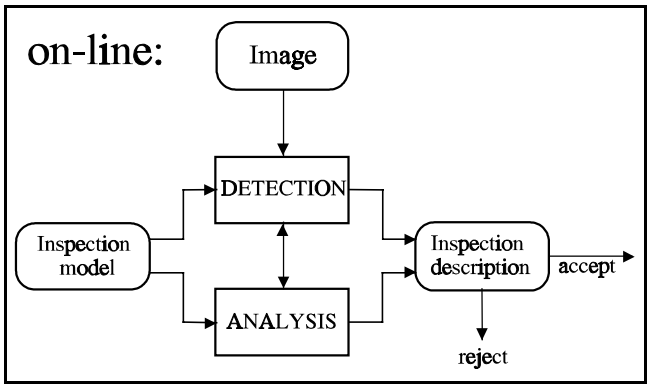


Figure 3.2 On-line inspection system components

In Section 3.1 definitions on expressions used in different abstraction levels of the concept are given. Next, the general concept using detection and analysis is discussed. The central part of the proposed concept is the off-line construction of the inspection system, separated into the inspection model generation, providing the inspection model (Section 3.3); and the detection of the features (Section 3.4). Furthermore, the analysis strategy (Section 3.5) using the inspection model, which consists of primitives and relations among primitives, and the inspection system generation (Section 3.6), where the preliminary inspection system is tested and adapted, are shown. The chapter concludes with a discussion of the concept.

3.1 Abstraction Levels

The proposed AVI concept is model-based. Therefore the definitions for model-based inspection defined in Section 2.2 are extended to express the relation between the object, its model, the image, and the detection within this concept.

In Figure 3.3 the relations are shown schematically, consisting of following components:

- *Object*: the *object structure* is formed by its *shape features* (or geometric features) and its *properties* (see Section 2.2.2).

- *Description* (or inspection model): the description of the object is structured in a description language consisting of *primitives* and *relations*. Primitives are a representation of shape features, relations represent the properties.

- *Image*: the image contains *features* and *patterns* which are a representation of shape features and properties. Since the description is used to perform the inspection, features and patterns interact with the representation of the object structure in the description. Furthermore *image characteristics* are a property of every image influencing the detection.

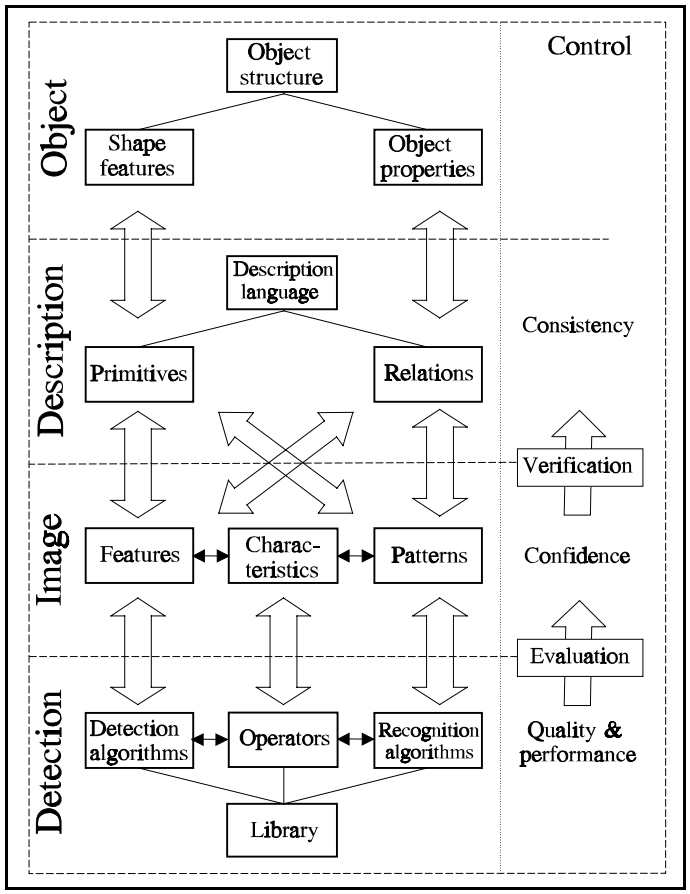


Figure 3.3 Relations between object, description, and image

- *Detection*: to detect features in the image, *feature detection algorithms* are used, *pattern recognition algorithms* perform the detection of patterns. Both of them use *operators*. Quality and performance of operators depend on image characteristics (see Section 2.4.4). Different operators and algorithms are structured in a *detection library*.

The *quality* and *performance* of the detection algorithms is evaluated with respect to the image. The *evaluation* states whether a detector is able to detect a feature or not within a certain *confidence*. The confidence of the detected features and patterns is used to perform the matching, the *verification* of the result of detection defines the consistency of the image with respect to the description and thus the object structure.

3.2 Concept Overview and Training Strategies

The concept of separating detection and analysis for visual inspection systems has to be integrated into a complete system that makes it possible to set up inspection systems interactively with the user. This interaction is valuable since the specific application knowledge can only be provided by the user, who is familiar with the specific inspection problem and the inspection hardware available. It is supposed that the system set-up takes place off-line at the final working site with the illumination and hardware that should be used in the final system.

3.2.1 Elements of the Concept

The elements of the general concept, shown in Figure 3.1, can be subdivided into distinct elements, each of them contributing an important part for the final system. The proposed general concept (Figure 3.1) can be seen in detail in Figure 3.4. The main elements are represented as shaded boxes with section numbers indicating where these elements are discussed in the thesis. Each of the elements is briefly described in this section in order to get a global overview of the proposed concept. Figure 3.4 is intentioned as "map" that guides the reader through the concept. Furthermore, the structure of this and the following two chapters reflects the main structure of the concept.

Image acquisition

The illumination and imaging geometry should be worked out carefully to increase the quality of the source data (see Section 2.2.1), influencing the quality of the result. To provide useful source data for the inspection process, the image acquisition properties should not be changed after the set-up. Illumination should be constant, keeping the influence of ambient light or other disturbing sources, depending on the acquisition system, as low as possible.

Test series

After having fixed the imaging geometry, the user has to provide a representative set of images of the inspection problem, i.e. the object to be inspected. To perform the later inspection system test, there should be $n + m$ different objects of the same type consisting of two groups, one "golden" test series, containing n fault-free objects, and one defect test series, containing m objects with representative defects. These series of $n + m$ images are used to build a statistical model, containing the image characteristics such as noise, texture, and grey level spread. To have reliable statistical data, $n + m$ has to be statistically significant, depending on the reliability required. The "golden" test series is not only used to gain image attributes but also to select the detection algorithms, adjust their parameters, and evaluate the quality and performance of the detection in the analysis graph, and it is used together with the defect test series for the final inspection system test.

Generic detection and primitives proposal

Since image characteristics are available, one representative image is chosen to perform the detection of parametric features in the image. The kind of features to be looked for can be fixed interactively. If there is only a limited number of features within the image, all other types are excluded in order to speed up the detection process. The output of the generic detection is the detected parametric primitives called primitive proposal. These primitives are represented in a preliminary description language without weights and tolerances, since these two parameters cannot be defined by generic detection. Tolerances have to be added manually, depending on the performance of the system, weights for detecting primitives are added by the detection. The primitive proposal contains the specific parameters of the features found, and the spatial relations among them.

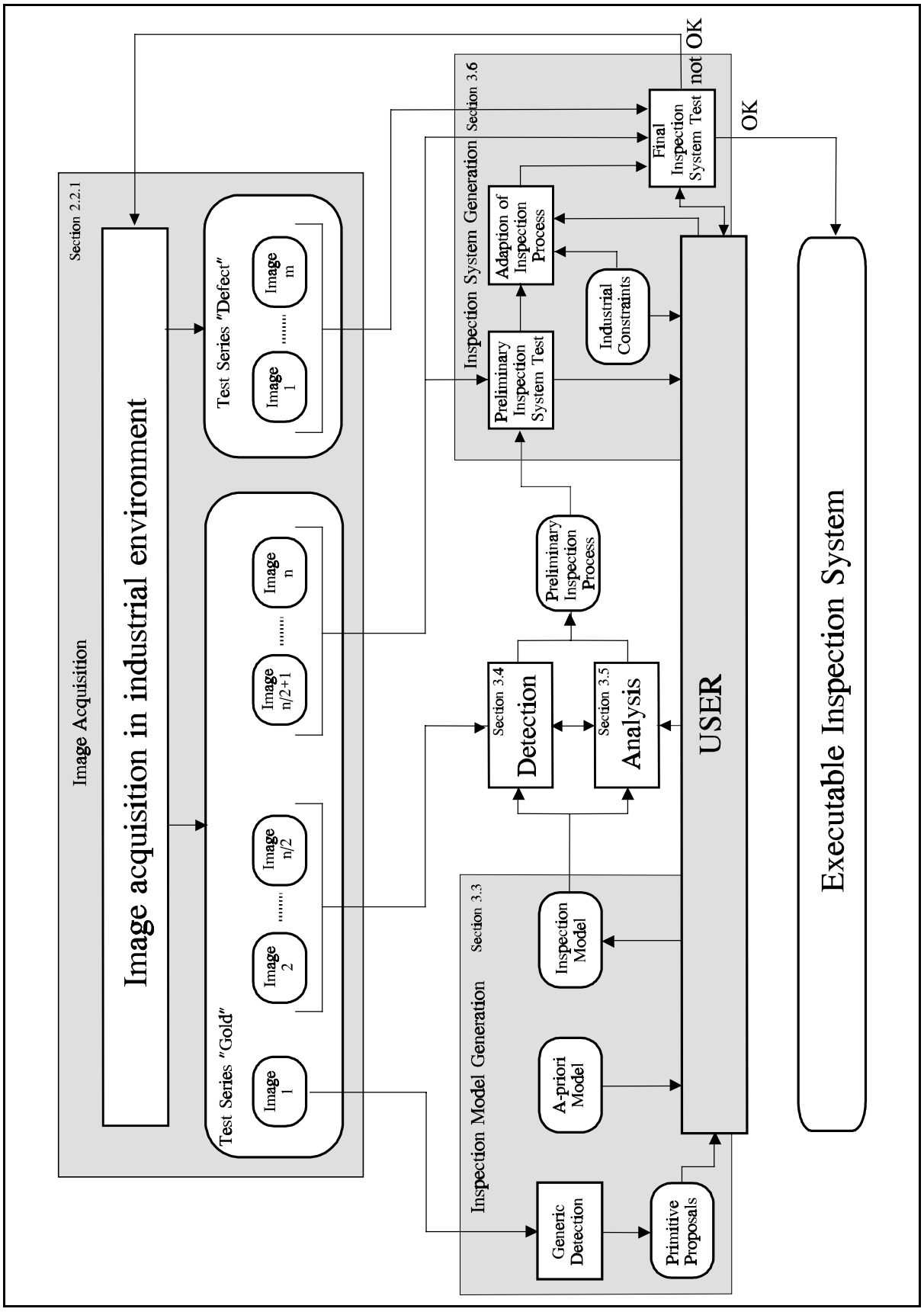


Figure 3.4 Inspection concept (schematic)

A-priori model

A-priori models provide a well defined model for inspection since a mathematical description of the shape and the primitives of the object is available. A-priori models like Computer Aided Design (CAD) models are augmented with manufacturing information, including material type, geometric tolerances and color, thus providing a unified description which can be used. The model data are taken to verify the result of the generic detection and to add tolerances to the description. Since generic detection ensures that the description can be matched, the use of the model data without performing the detection is not recommended. There could be a registration error or an error in the model that would complicate the set-up of the analysis. Furthermore, the resolution of the image can be determined by matching the CAD model (metric values) to the preliminary description with pixel values.

Inspection model

The user interactively defines which of the primitives found should be used for inspection and which tolerances and weights have to be considered. Furthermore, he should provide the calibration (i.e. the unit of measurement), in order to relate image distances to real world distances. A possibility to add non-parametric primitives, is necessary since such primitives (like letterings) play an important role. Furthermore, the user has to add non-geometric relations between primitives, for instance which primitive belongs to which sub-object. This interactive selection of primitives, their parameters and relations generates the inspection model, i.e., the description. It contains the application-specific information for inspection. Therefore, the analysis is guided by the description, holding the a priori information used for inspection. Furthermore, the description provides all parameters necessary for the detection.

Detection and analysis

A preliminary analysis process is produced using a library of different detection algorithms, parameter adjustment algorithms and the analysis graph instantiation, using the series of *n/2* test images and evaluating the results. There is a strong interaction between the analysis and the detection, the defined interface permits the testing and adjustment of the different parameters. This procedure tries to optimize both the detection strategy and the performance of the detection, therefore optimizing the overall performance.

Preliminary inspection process and preliminary system test

The preliminary analysis graph contains all parameters necessary to perform the inspection. This preliminary inspection process has to be evaluated whether it is able to detect the features in defect-free images performing a preliminary system test. The preliminary system test computes the inspection parameters: accuracy, performance and time. The result of this test and the preliminary inspection process have to be checked by the user interactively.

Industrial constraints

Due to industrial constraints, the preliminarily fixed parameters have to be verified. If the preliminary inspection process does not adhere to the given constraints, there are two different

strategies to solve the problem. One is to check if the constraints could be altered in order to have a satisfactory inspection result, the other strategy is to alter image acquisition, illumination and a reduction of tasks and degrees of freedom for inspection.

Adaption of inspection process

The user has to balance parameters like accuracy, performance and speed due to the constraint given by the inspection task. If for instance, the time for inspection is limited due to the production flow, the inspection process has to use faster, but less reliable algorithms or the search space has to be reduced dramatically in order to adhere to this time constraint. Normally the adaption will consist of changing acquisition or hardware parameters first, and if this does not solve the problem completely, other strategies like new software components or decreasing inspection expectations will be taken into consideration.

Inspection system test

After the interactive adaption of the inspection process, a final inspection system test has to be carried out in order to ensure that the inspection has succeeded successfully. The images *n/2+1* to *n* are taken to verify the functionality of the system (detecting false negatives) and the *m* defect images are taken to verify the defect detection rate of the inspection system (false positives). If the intrinsic constraints are attained in this final system test, the developed inspection system can be installed for long term system test during production. If the inspection system does not reach the requirements, it has to be re-adapted again by changing the inspection process or the acquisition or illumination parameters.

Final inspection system

After a successful inspection system test, the final inspection system is ready to be tested under production. The instantiated analysis graph and the description language are taken to perform the inspection. The result of the on-line inspection is stored in an inspection description, where all parameters necessary for inspection are listed, defining if the object under inspection is within the given tolerances or not.

3.2.2 Training Strategies

The proposed inspection concept supports the user in setting up specific inspection tasks [SAB95a]. This procedure can be seen as a training phase for both the user and the system. Suppose the user is unfamiliar with visual inspection algorithms. Then he would expect that the computer guided inspection system would solve any inspection task faster and more reliably than the human operator. After a few interactions with the system he would learn that visual inspection algorithms perform acceptably only if a number of constraints is taken into account. The user has to think about lighting, image acquisition, and hardware, to guarantee a certain minimum data quality needed for detection and inspection. So any human who is confronted with a new system has to learn the behavior and constraints of the new task. This fact is also true for the machine, it has to adapt to any new problem it is supposed to solve.

This "learning" mimics the human ability to adapt to a changing environment. Two major training strategies for visual inspection, can be distinguished by the complexity of the algorithms and strategies used to perform the training:

- *Training by showing* techniques: set up an object model based on a number of samples of the object. The system and the operator interactively try to find features that distinguish between different objects and the same objects. The goal of these techniques is to build up a model that can be used for inspection (see Section 2.2.3).

- *Training by learning* techniques: extract specific information automatically and learn for instance which parameters of an algorithm influence the result and how they have to be set in order to achieve an optimal output. There are several different techniques for training by learning (see [WES96] for a survey). Adaptive image processing, for example, uses attributes of the image, like amount or type of noise, feeds them back or forward in order to adjust parameters of detection algorithms. The learning may be supervised, i.e. there is an interaction with the user, or unsupervised.

The proposed concept uses a training by showing strategy to generate the description; the system part is performed by the generic detection of the primitives and, if available by the use of the a-priori model, the interactive part is the selection and addition of primitives and their tolerances. Furthermore the concept provides a supervised technique to instantiate the analysis graph and to set weights in the description. Detection algorithms are selected and evaluated, their parameters are adjusted in order to guarantee optimal performance. Since this selection and adaption process uses not only one image of the object but a series of them, one can regard this as a combination of training by showing and learning.

3.3 Inspection Model Generation

All visual inspection systems use a priori knowledge to perform the inspection (see Section 2.2). Therefore, the first step in the set-up of an inspection system is the off-line generation of the inspection model. Since this step is performed off-line and only once per application it is not time critical. The goal of the inspection model generation is to provide a description of complex objects using a small set of simple primitives and relations (or structural rules). Due to the structural rules this approach is referred to as structural or syntactic pattern recognition approach [FU82a,BUN92]. The "syntactic-structural" methods use the internal structure of the object as an element of analysis. They are based on the fact that an object can be described recursively from simple primitives by its structure.

3.3.1 Definition of Primitives

Syntactic methods use an analogy between the structure of pictorial patterns and the syntax (or grammar) of languages. Pictorial patterns are specified by composing together subpatterns, called *primitives*, just as sentences are built up by concatenating words, and words are built up by characters. Using the visual inspection terminology, features are represented as

primitives. The inspection model is the language, called description language, with a well specified grammar governing the correctness of a sentence. After each feature has been detected, and the description is established, the inspection is accomplished by performing a syntax analysis of the sentence describing the given pattern to verify whether or not it is syntactically (or grammatically) correct with respect to the specified grammar.

Some approaches especially for PWB inspection were studied, such as [BJO77,JAR78, JAR80], all of them struggling with the problem of extensive preprocessing before the parsing can be performed, making the technique unattractive for inspection. The reason for the computational expense lies in the selection of very simple primitives, like short straight lines, that have to be detected accurately and distinguished from one another. A great number of simple primitives have the drawback that small distortions or noise in the image influences the analysis in such a way that nearly every object is declared faulty.

Figure 3.5 shows a simple example. The primitives defined are *a,b,* and *c*, the boundary of the object is formed by the string *aaabbcc* having *$* as start symbol. In this example, the length and position of the primitives have to be detected exactly, which is not simple if lines are connected without a variation of the direction. Therefore nearly every object would be declared faulty. In the syntactic terminology in this example, the language is parsed on the "character level" rather than on the "word level". Furthermore, a distortion or rotation of the boundary would result in primitives which are not detectable.

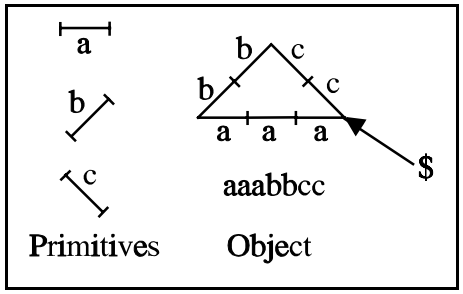


Figure 3.5 The contour of the object is described by using three primitives a,b,c

The drawback of syntactic methods lies in the detection of primitives since there are either too many of them or their pattern is too simple. We decided to use parametric primitives detected by well-known detection algorithms to reduce the influence of ambiguous primitive detection. Again, in terms of syntactic methods, we use the "word level" rather than the "character level".

In order to provide a general concept, primitive types are not limited or constrained. The application type (binary or intensity images representing images from graylevel or color cameras, X-ray, ultrasonic, range, and other sensors - see Section 2.3) and data type (sparse, dense, noisy) is thus application dependent. Generally, the following constraints for choosing primitives have to be considered [DAR88,SAB96b]:

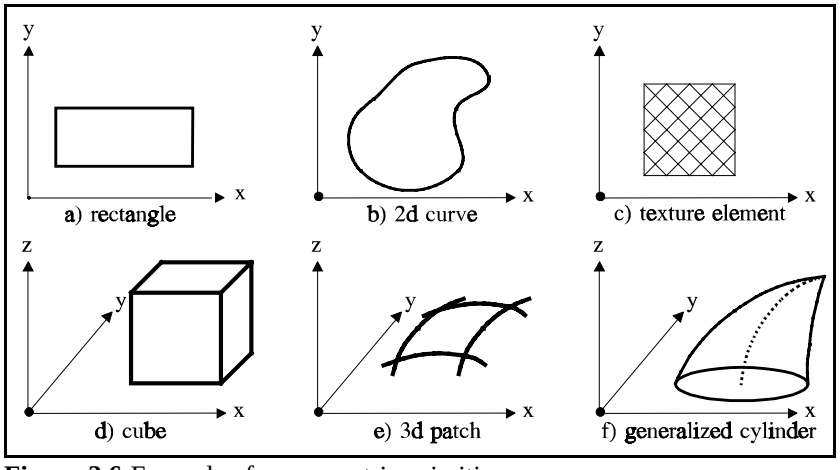


Figure 3.6 Examples for parametric primitives

- Primitives should have a unique name.
- The chosen set should have a finite number of primitives.
- Primitives should be independent, i.e. not defined in terms of each other.

- The set of primitives should be compact. No subset should be replaceable by smaller sets.
- Primitives should be comprehensive. They should be basic pattern elements to provide compact but adequate description of patterns in terms of specified structural relation.

Therefore, primitives can be formulated depending on the type of application. Some examples for such parametric primitives are shown in Figure 3.6. In the 2-d case one can choose simple geometric primitives like rectangles, circles and the like (Figure 3.6a), parametric 2-d curves (Figure 3.6b), or texture primitives (Figure 3.6c) to name a few. In the 3-d case, examples for such primitives are cubes or boxes (Figure 3.6d), surface patches (Figure 3.6e), and generalized cylinders (see [MAR82]), shown in Figure 3.6f. Again only some of the possible primitives were listed here.

In this thesis we concentrate on the detection and inspection of 2-d projections of 3-d objects using a 2-d model representation without limiting the extension to other types of data.

3.3.2 Feature Detection by Generic Algorithms

Generic algorithms can robustly extract the primitives formulated by parametric models [STR95a]. The framework makes accurate and robust extraction of parametric primitives of different types possible. It includes a mechanism that lets each primitive type determine its domain of applicability. The framework is general in the sense that it can be described concisely *without* specifying the following components: A domain of application, a particular type of data, and a set of admissible primitive types (straight lines, ellipses, planes, cylinders, spheres, etc.). A large part of this framework can be implemented as a generic algorithm and thus, it does not only serve as a conceptually clean approach to primitive extraction, but it also provides a highly reusable algorithm which can be linked with a specific set of admissible types of primitives and a specific fitting/detection functions.

Our framework for accurate and robust extraction of parametric primitives consists of four main components: exploration, selection, fit, and a final selection shown in Figure 3.7 [STR95b]. The exploration is a dynamic data-driven masking technique that proposes a set of models from which the selection chooses the ones that explain the data with minimal description length. Selection is performed by *tabu search* [GLO93], a discrete optimization technique that outperforms annealing techniques on many classical optimization problems. The robust fitting technique, which increases the accuracy of the selected models, may change the classification of data elements which the final selection requires.

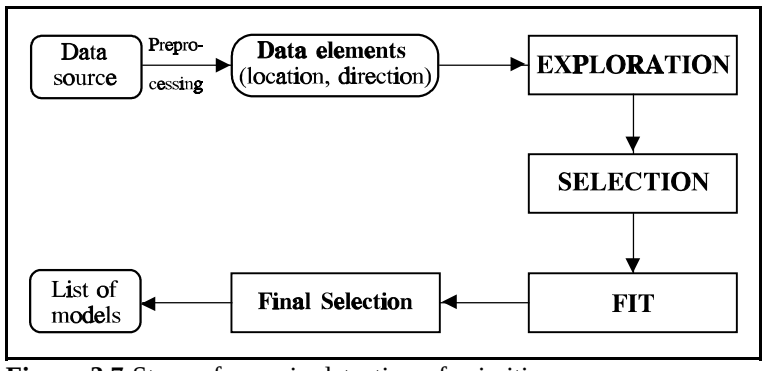


Figure 3.7 Steps of generic detection of primitives

Although the framework for extracting parametric models is completely general, we will place our presentation in the context of extracting 2-d parametric models from boundary elements. We assume that the data consists of boundary elements, *i.e.*, edgels in 2-d, of which we know the location and the normal direction onto the boundary. First we have to choose

the set of admissible model types. For each model type we have to be able to compute its parameters given a *k*-tuple of boundary elements and to compute an approximation of the Euclidean distance from a boundary element to a given model. For technical details on the algorithm, the reader is referred to Appendix A.

3.3.3 Description Language

The object structure (shape primitives and properties) has to be represented in a description language consisting of a graph structure in which nodes represent the primitives and arcs in the relations between primitives. A priori information concerning the quality standard (e.g. manufacturing tolerances and detection tolerances [GRI92]) are also part of the model. From the description language point of view, the modeling can be interpreted as a syntactic pattern recognition approach in which the primitives are transformed into the vocabulary and the relations are transformed into a grammar [FU81].

The proposed approach makes use of the idea of shape decomposition, it divides complex shapes into simple elementary units, i.e. primitives. This concept can be seen as an application of semantic networks [QUI68,DAR88], since semantic networks are labeled, directed graphs where nodes represent objects, sub-objects, or shape primitives and arcs represent relations between them. A set of attributes that describe different object features is attached to each node; a set of attributes that describe different properties is attached to each arc. Once the object is transformed to this representation all operations for recognition, verification, and inspection can be executed on this graph structure. The advantage of a description language lies in the uniqueness of representation, different objects result in different descriptions.

Formally, the description language is a graph $G=<o,R>$, where $o=\{ m \mid 1 \le m \le n \}$ denotes the set of nodes and $R=\{<c,d> \mid c,d \in o\}$ the set of arcs. A node o consists of different sub-objects or primitives. Each node has different attributes *a*, with weights *w*, and tolerance *T(a)* defined as:

$$T(a) = \begin{cases} 1, \; if \mid a^{mod} - a^{img} \mid \le c, \; and \\ \dfrac{1}{\mid a^{mod} - a^{img} \mid} \; otherwise, \end{cases} \tag{3.1}$$

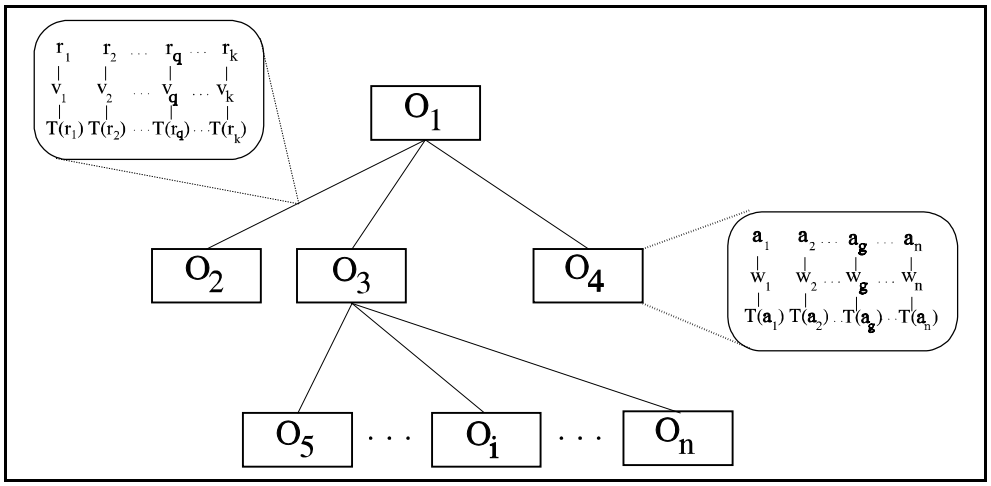


Figure 3.8 Description language graph

where c is the allowed tolerance, a^{mod} denotes the value of attribute a in the model, and a^{img} the value of the attribute a in the image.

Two nodes are in relation according to R. Each relation $<c,d>$ is decomposed into k subrelations between the same nodes, each with a weight v and a tolerance $T(r)$ defined analogue as for weights and tolerances of attributes. Figure 3.8 shows the graph and the inner structure of nodes and arcs. Note that all attributes and relations contain numerical values.

The weights w and v are necessary for the model verification. Each of the geometrical, positional, and relational properties has a certain weight in order to verify the corresponding description to a given image. Since these weights are influenced by the data and therefore application dependent, they have to be fixed during the set-up procedure. The verification of image to description consists of verifying whether the number and type of features and primitives are the same. Next, attributes and relations are checked whether they match within given tolerances. The verification process is carried out by comparing all attributes of a node and its successors with the model. The confidence for a node can be computed based on the result of the comparison:

$$conf(p) = \sum_{g=1}^{n} w_g \cdot T(a_g) + \sum_{<p,q> \in R}^{m} v_{<p,q>} \cdot conf(q), \tag{3.2}$$

where w_g are the weights of the attributes of the nodes and $v_{<p,q>}$ the weights of the sub-relations of the arcs. Observe that n, the number of attribute values, and m, the number of arcs, depend on the node p. Moreover, for leaves we have:

$$conf(p) = \sum_{g=1}^{n} w_g \cdot T(a_g).$$

This enables us to compute the confidence of a node by summing up the weighted tolerances of each attribute of the node and the overall confidence of the subgraph connected to this node. By computing the consistency for different descriptions the one with the highest confidence value can be chosen if the confidence is above a certain threshold. The use of weights also allows a two step identification; primitives or sub-objects with high weights are first detected and checked, next primitives with low weights are postulated on a certain position and then verified.

3.4 Detection

The success of feature-based inspection techniques depends on the quality of feature detection. Therefore, feature detection algorithms have been the subject of investigation over the last ten years. Expert systems have been developed and used to solve and refine feature detection. Problems, such as edge detection, edge aggregation, and region extraction, to name the most important in 2-d feature detection, belong to the mathematical class of inverse ill-posed problems [POG85]. There exists no unique and stable transformation function that can build a specific description starting from an arbitrary observation. To overcome this problem, one has to reduce the number of acceptable solutions by introducing a priori knowledge of the problem space on the solution space and considering the detection process as being decomposed into a sequence of sub-problems that are either well-posed problems or problems for which classical regularization methods exist [CRE93].

3.4.1 Feature Detection for Inspection

The detection of features in the inspection process is different from the generic detection of parametric models since after the definition of the primitives knowledge about the object is already available and should be taken into account. It should be mentioned that generic detection could detect features in the inspection process too, but because of the computational complexity of the algorithm, it is not useful to neglect the already acquired information. Therefore, the detection stage in the inspection concept re-uses standard pattern recognition algorithms to solve specific feature detection problems.

Software re-use has the potential to provide a significant increase in software productivity and reliability. In fact, it has been touted as the "only realistic approach" to meet the needs of software industry [MIL95]. The key to software re-use is a representation for software design: a component must be represented in a way that it supports both retrieval from a library as well as evaluation of usability [PEN95]. The goal is to minimize adaption cost of the software to be re-used, to the new task that should be solved. For any of the defined primitives a library of different pattern recognition algorithms should be accessible. It is possible to constitute a large library of operators and then to envision feature detection as the generation of a program, through model- and operator selection and parameter adjustment.

In [MAT89] a typical planning procedure for feature detection is described. After the user has specified a purpose (like finding a rectangle of a given size in a specified region of the image), a plan consisting of an ordered sequence of abstract processing algorithms is elaborated. For this example a solution of detecting the rectangle could be: Edge Detection - Thresholding - Linking - Grouping. Since multiple algorithms for each processing step are possible, control rules, taking image characteristics into account, assist in the determination of a solution, and in determining parameter values for the operations. If processing costs are available for each operation (depending on the hardware), then the computation cost can be part of the optimization function.

Feature detection algorithms are composed of detection components (for the example of the primitive rectangle there are 4 components). Different components (like different edge detectors) have to be arranged to find an optimal solution in terms of quality and performance. To be able to use different components, a common interface has to be defined for the components and if different algorithms should be used to detect the same type of primitive a defined interface has to exist for algorithms, too. An interface specification describes the function of the component or algorithm. The effects of the execution of a component or algorithm on the program state are described in the definitions provided by the primitives.

The interface specification can be broken up into two components:

- *Detection algorithm specifications*: given a domain and the range of the feature to be detected, in- and output conditions are described. The input condition specifies the set of domain values that have a defined output, called the *legal inputs*. The output condition specifies the acceptable outputs given a legal input. This set of outputs is called *feasible outputs*. Any behavior can occur for an illegal input.

- *Component specifications*: is described in the same way as the algorithm specification, the only difference is that feasible outputs are constrained to be well defined over the legal inputs.

3.4.2 Evaluation of Feature Detection Algorithms

Interface specifications can be used to evaluate a component and verify its usability. The primary goal of evaluation is to show whether a component specification does or does not satisfy a detection algorithm specification. A component satisfies a detection algorithm specification if, for any of the legal inputs of the detection algorithm, the component results in one of the feasible outputs. Two conditions have to be fulfilled:

1. the component has to accept all legal inputs. The component specification assures that a legal input to the component results in a feasible output of the component.
2. all feasible outputs for a legal feature detection algorithm input are valid outputs.

Evaluation has to prove that the component is a legal solution to the given problem. In other words, evaluation becomes verification. Low level image processing, such as edge- or feature detection must reveal as much of the image structure as possible without reference to domain knowledge. Experience reveals that applying algorithms with standard parameter values often results in very poor performance in this respect: for example filtering algorithms need to know the statistical properties of the noise to function properly [CRE93]. One solution to this problem consists in adjusting the parameters interactively, i.e. the user subjectively judges the results on an output device (like the monitor) [MAT89]. The interactive solution defeats the purpose of automated inspection, but it is the only one used so far. Since the human eye is a poor judge for feature detection results, researchers have developed formal error measures to assess the degree of conformance of operator evaluation (see Section 2.4.4) and segmentation (see Section 2.4.3). Furthermore, systems are under development that solve the problem of algorithm selection and parameter adjustment (see for instance [BOD95, CLE89,CLE93,THO95]), while others try to evaluate vision modules and their performance (see [HAR94,DBO96]). However, up to now no general purpose image processing system has been presented because this problem is not trivial to solve [CLO95]. In this thesis selection, parameter adjustment, and evaluation of detection algorithms was performed interactively due to the lack of automated systems in this area.

Generally each detection component of the image processing system has to be tested within the industrial environment, with fixed imaging geometry and illumination. Each component and finally the detection algorithm has to be evaluated with respect to three constraints:

- *Performance*: The performance of detection influences the overall performance of the inspection system. If a detection algorithm attains a certain threshold (typically more than 99%) of performance (recognition rate) in a testing phase, it is qualified for the final inspection system. Algorithms with lower performance may be selected, if strategies for detecting and fixing errors (like multiple detection schemes) are applied.

- *Accuracy*: If the a priori minimum accuracy is not attained by the feature detection algorithm, there are three ways to solve the problem:

    1. The selected detection components or their parameters could be altered to achieve the desired accuracy.

2. The resolution should be checked taking into account the Whittaker-Shannon theorem [ROS82], i.e., the fundamental limit on resolution. To overcome the resolution problem, the area of the feature within the field of view of the camera has to be enlarged.
3. Applying sub-pixel algorithms to enhance resolution in those parts of the image where higher accuracy is necessary. Since sub-pixel algorithms need interpolation techniques they have to be adapted to the kind of data they are interpolating.

- *Speed*: This constraint can be handled by using faster hardware, but this is not always possible since the costs (see Section 1.3) often do not allow sophisticated hardware. To overcome the problem, the global order of analysis may be changed, features that can be found fast should be searched first and then further search strategies can be developed.

Since all of the constraints are interdependent, there is a need to achieve a balance between these constraints, e.g. lower accuracy results at higher speed but may affect the performance, higher performance will require higher resolution and thus longer computation time. Basically all knowledge-based image analysis systems balance these constraints, emphasizing performance and accuracy at the cost of speed.

3.5 Analysis

In contrast to detection, where independent algorithms detect features, the analysis deals with the application specific knowledge: the object model, tolerances and the order of the detection steps. After the generation of the inspection model (description, containing inspection information), the weights for each primitive are not fixed, since at this stage it is not clear which primitive can be found reliably and which not, and which primitive is important for recognition. Therefore, one goal of the analysis is the determination of the weights of primitives. To compute weights, the detection order and the parameters of the detection algorithms are evaluated, features with high detection confidence get higher weights than features with low confidence. To model the detection order in the analysis we use an analysis graph.

The description language provides the inspection model, generally the inspection could take place based on this model. Since performance and speed are crucial and the verifying phase is a graph isomorphism problem whose general case is known to have NP complexity [EVE79], the semantic information stored in nodes and arcs decreases complexity.

There are two different strategies for relating features in the image coordinate system to the object-centered coordinate system:

- *Parallel* (M1): Independent detection and localization of features followed by a verification of the constraints.

- *Sequential* (M2): Detection and localization of reliable features; construction of hypotheses based on imposed constraints of the detected features; verification of hypotheses.

Since each method has its benefits, both principles are used in the proposed analysis process, again formulated in a graph structure introducing a hierarchy (see Figure 3.9). It represents the space of all possible solutions for the problem in the particular domain. A solution

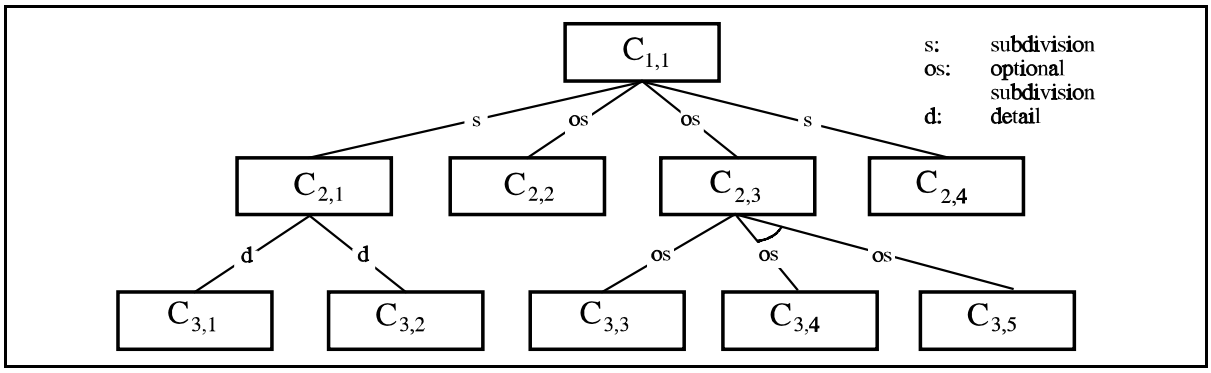


Figure 3.9 Analysis graph with three relation types

is formulated by instantiating the graph to form a unique solution. Each node represents an element of the solution called *cell*, i.e. an image processing task like Sobel edge detection or a working step like image acquisition. The *arcs* between the nodes represent one of the following semantic relations (see Figure 3.9):

- *Subdivide relation*: This is an n-ary relation representing a subdivision of a cell into its constituent set of sub-cells. For example, image acquisition can be subdivided into CCD size, upper and lower threshold for transistor response, digitizer, illumination adaption, and frame transfer to name a few. Each subdivide relation has a weight, which allows an ordering of relations within the graph, subdivide relations with equal weights may be executed in parallel. For the example in Figure 3.9, the cell $C_{1,1}$ is first subdivided into $C_{2,1}$ and afterwards into $C_{2,4}$ because of the higher weight of the left subdivide relation.

- *Optional subdivide relation*: This is also an n-ary relation allowing optional subdivisions of a cell providing alternative subdivisions. For example, a segmentation can be performed by using region growing or optionally by split and merge. For the example in Figure 3.9 the cell $C_{1,1}$ can be subdivided into either $C_{2,2}$ or $C_{2,3}$. The relation is again provided with weights. If there are optional subdivisions that have to be performed together, this is denoted by an arc in the graph. In Figure 3.9 an optional subdivision of cell $C_{2,3}$ may result in $C_{3,1}$ or optionally in $C_{3,2}$ and $C_{3,3}$.

- *Detail relation*: This is a relation between a cell and a detail of the cell. For any cell there may be a number of different possible details. For example, a cell representing edge detection could be detailed in Roberts, Sobel or Canny algorithm.

An example for an analysis graph is given in Figure 3.10 on the left, which shows that for example rectangle detection can be optionally subdivided into the cell "Grouping B" and optionally also into "Grouping A". Next "Grouping B" can be optionally subdivided into the cell "Line detection" or optionally into the cell "Thresholding" which is subdivided into "Edge detection". Up to now only subdivision relations were encountered in this graph, all of them represent a certain class of image processing algorithm. Detail relations like the three possible cells "Roberts edge detection", "Sobel edge detection", and "Canny edge detection" of "edge detection" in the graph represent actual operations on data. One possible solution, a so-called instantiation of the analysis graph, would be the rectangle detection using "Grouping B", "Line detection", and "Burns line detection", shown in Figure 3.10 on the right.

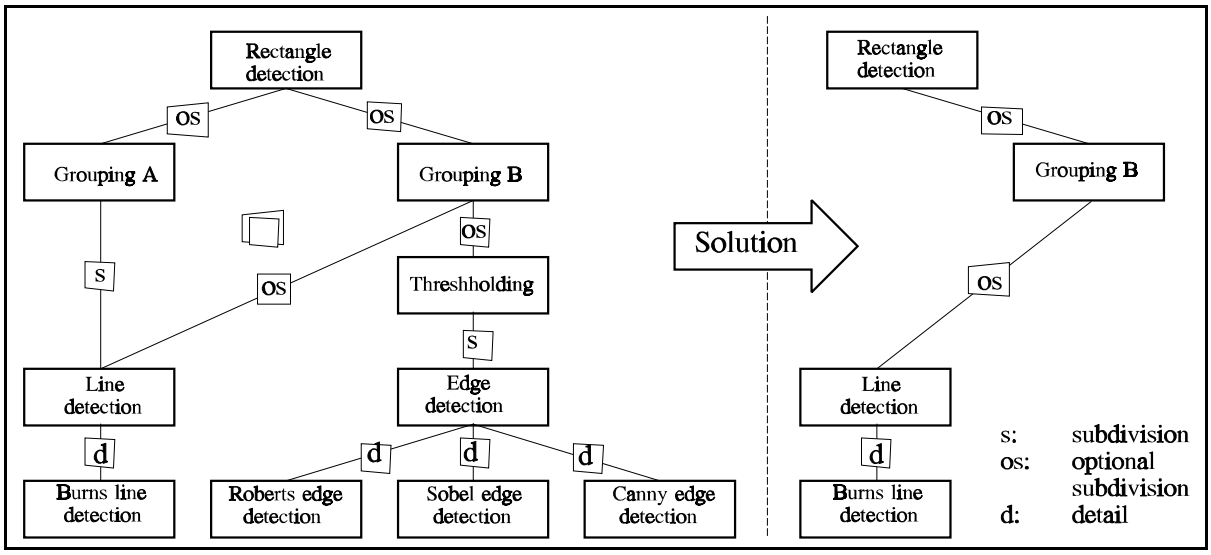


Figure 3.10 Example for a hierarchical analysis graph

Each cell of the analysis graph can have a set of in- and outputs, which can be connected to the in- and outputs of other cells in the analysis graph, representing the data-flow in the analysis graph (see Figure 3.11). The hierarchy in the graph is kept, the output of one cell in one level may be interconnected to an input on the same level, the in- and outputs of cells on different levels represent the same values. In Figure 3.11 the input parameters p_1,p_2,p_3 of cell $C_{1,1}$ are either the input parameters for $C_{2,1}$ (p_2) and $C_{2,2}$ (p_1, p_3) or optional input for $C_{2,3}$ (p_1,p_2,p_3). The parameters p_1 and p_3 are computed either by cell $C_{3,1}$ or $C_{3,2}$, represented by the detail relation of cell $C_{2,2}$. Detail relations of $C_{2,1}$ and $C_{2,3}$ are not shown in Figure 3.11.

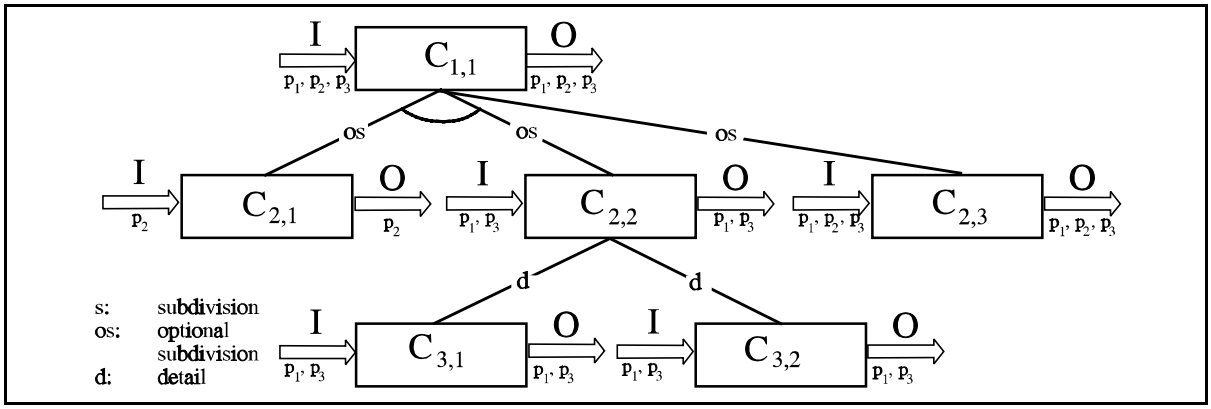


Figure 3.11 Dataflow in the analysis graph

The analysis process contributes two elements to the final inspection process, i.e. the building of the inspection model with the help of the description language with the corresponding weights for each node, and the analysis graph containing the specific detection order and the specific detection algorithm parameters.

The building up of the analysis graph for inspection can be generalized, since inspection differs only in the description and detection, the overall process is common to all of the inspection problems. Figure 3.12 shows the general analysis graph for inspection. The input of the *object inspection* graph is the *description*, the output is the *result* of inspection. The object inspection can be (non-optionally) subdivided into:

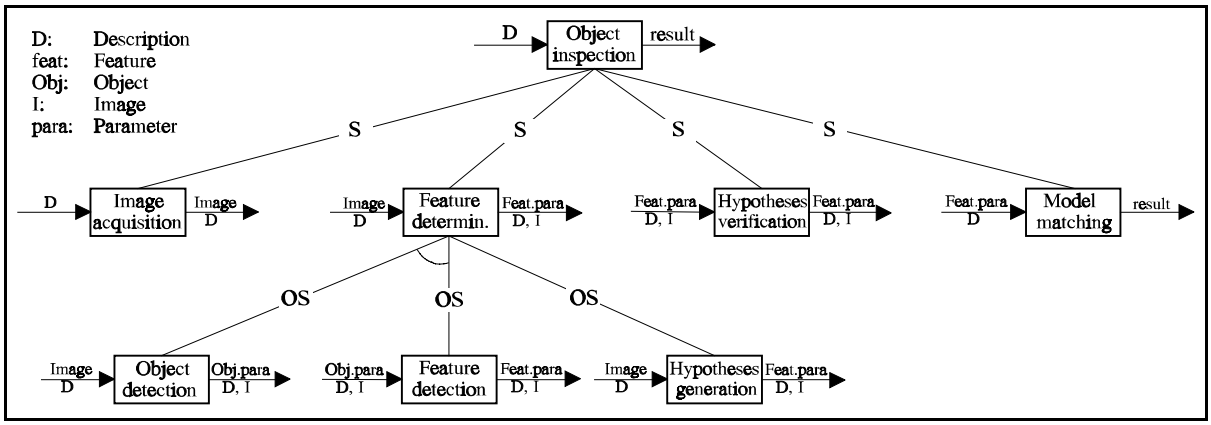


Figure 3.12 General analysis graph

- *Image acquisition*: This cell has the image and its statistical parameters like noise distribution as the output.

- *Feature determination*: The feature determination determines specific parameters of features (like position and size), within the image. Depending on the image acquisition, feature determination can be subdivided into:
  - *Object detection*: The object has to be located within the image and its size has to be determined, and
  - *Feature detection*: Within the object, features used for inspection have to be located.

  Optionally, if the image acquisition parameters are fixed (the object has always approximately the same position, orientation and size) these two cells can be replaced by:
  - *Hypotheses generation*: Hypotheses about the position, orientation and size of all features are generated with the help of the description.

- *Hypotheses verification*: The specific parameters of the features, either determined by detection or by hypotheses generation, are checked whether they match with the description using features, which were not used up to this step.

- *Model matching*: The final step of analysis matches the actual parameters of the features with the description, thus producing the result of inspection.

The primitives have to be detected in all of the images of the test series. The weights of the analysis graph are set in accordance with their probability of detection. The inspection process "learns" which detection algorithms should be used in a specific area of the image and how many parameters must be found in the image rather than in the description. This introduces a detection verification concept guided by weights in the analysis graph. If some primitives can be found accurately with a high probability, they get a high weight. Primitives with high weights are searched first, if they can be found, primitives with lower detection probability are predicted in a specific part of the image (a hypothesis generation) and then checked for their presence (a verification of the hypothesis).

The instantiation of the analysis graph can also be seen as: Given a directed graph, find an optimal solution path with minimum amount of computational effort. For solving this kind of problems standard techniques (like the A^* Algorithm [HAR68,PEA84]) can be used to find the minimum cost path in the graph, i.e., the best analysis strategy.

3.6 Inspection System Generation

The instantiation of the analysis graph supplies the detection order and the detection algorithms to be used. Weights of the primitives are existing, primitives with high weights have to be detected first, because the probability of their correct detection is high. Since the analysis graph was generated using an image set, the preliminary inspection system has to be checked to determine whether it works with a set of other images of the "golden" series. The preliminary system test mimics the on-line inspection system in order to determine the intrinsic parameters of the inspection process.

Following the preliminary system test, the user has to check interactively, whether the inspection system adheres to the industrial constraints. If they are not adhered to, strategies to solve the problem have to be developed. This interactive adaption of the inspection process results in a new instantiation of the analysis graph and possibly in a reduction of features to be looked for. The new inspection process has to be tested again if it attains the constraints within the testset of defect-free images (test series "gold") and a test set of defect-images (test series "defect"). This test determines the rate of false negatives, i.e. the number of objects classified as defective in the "gold" series, and the rate of false positives, i.e. the number of objects classified as defect-free in the "defect" series.

The final inspection system test, performed pseudo-on-line, shows if the inspection system performs correctly with defect and defect-free images, determining computation time, performance and accuracy. If one or more of the results are not satisfactory, an adaption of the inspection process has to take place. If the results are far from being satisfactory, a complete re-design of the image acquisition and illumination is necessary. The inspection model can be re-used, but the relation: world coordinates to image coordinates has to be determined, and weights, analysis graph, and detection algorithms have to be constructed again.

If all parameters lie within the constraints, the final inspection system is ready, description language and analysis graph together with the detection algorithms are used to perform the inspection (see Figure 3.2). Nevertheless, since the inspection system was set up using a limited number of test images, it has to be monitored during the first operational use. The quality of the representative sample influences the likelihood of bad performance of the inspection system; the better the actual image acquisition and object alignment match the final condition, the better the results. If it turns out that one class of primitive cannot be detected robustly, the detection algorithm can be replaced without changing the overall analysis.

3.7 Chapter Summary

The proposed inspection concept of separating detection and analysis is one solution to the problem of building up an inspection system. It is evident that this is not the only possible solution, there are several others reported in the literature. One approach is embedded in the decision theory (see [FUK72]), based on a minimum set of features (from the classification point of view). This approach is mainly used for recognition since it does not allow reasoning on the structure of the object. Therefore, this technique is used for defect classification only.

The already mentioned "classical" syntactic (also referred to as linguistic, structural and grammatical [GON78]) approach was introduced by Fu and extended by Pavlidis (see for instance [FU77,PAV80]). The internal object structure is taken as an element of analysis, based

on the fact that an object can be described recursively from simple primitives. For detailed description of techniques used by both of the above mentioned methods the reader is referred to [BUN92]. However, both of them perform pattern recognition rather than inspection. Applications of syntactic pattern recognition can be found in the field of character recognition, object recognition, classification of areal images and the like. For visual inspection the use of syntactical pattern recognition was reported for PWB inspection only [CHI88,NEW95]. The proposed concept could also apply this kind of technique for solving the analysis, since the primitives have to be very simple and the inspection takes place by parsing the grammar constructed out of the primitives and the relations between them, computational complexity is very high. Furthermore, the necessity for error correcting parsing [FU82b] is introduced since data in industrial applications are not perfect, as noise, dirt, and distortions prevent this technique from attaining high performance.

Differentiation between syntactical and theoretical decision methods is convenient from the theoretical point of view, for applications a combination of them is recommended [BUN92], to model both the object and the characteristics of the environment. First primitives are extracted and the inspection model is generated by syntactical methods, then the image characteristics are defined by statistical techniques. Structure and characteristics are then modeled by a grammar with attributes. Attributes allow a flexible selection of primitives and sub--structures, thus enabling all non-structural characteristics to be obtained as attributes. A further way to solve the analysis part would be the use of *picture languages* that describe objects in the 2-d Cartesian plane by unit lines. A well-known example for this kind of technique is the Freeman's chain code [FRE61] that represents unit lines in 4 (or 8) directions and which is used for contour line descriptions. There are other description models, some of which are similar to the chain code scheme (see [ROS79]). There are no applications of picture grammars to visual inspection reported in the literature, since they are of very theoretical structure and rather for constructing pictures than for analyzing images.

In this chapter the proposed new visual inspection concept was described. Following a definition of expressions used in different abstraction levels, the elements of the concept were described in order to give a general overview. Subsequently this global concept was discussed in the remaining sections of the chapter, always following the main structure of the concept. This section arrangement can also be found in the following two chapters in order to explain how the theoretically formulated stages of concept are applied in practice. The user starts the set-up of the inspection system with an interactive generation of the inspection model, i.e., the description. Geometric features that should be detected were defined and the use of generic algorithms to find them automatically was discussed. Next, the definition of the description language, which has primitives as nodes and relations between the primitives as arcs, was shown. The description had to be set-up interactively, with the results provided by generic feature detection. In order to detect the primitives during inspection fast and robustly, feature detection strategies with defined interfaces were discussed, so as to be able to use different detection algorithms for the same class of features. Selection, parameter adjustment and evaluation of feature detection algorithms was performed interactively, due to the lack of automated systems. Since detection and analysis are separated, detection can be modeled as detail relation in the proposed analysis graph. The analysis itself was defined as subdivision or optional subdivision in the analysis graph. The interactive instantiation of the analysis graph supplies the detection order and the detection algorithms to be used. Finally, the integration of detection and analysis in the inspection system generation were shown.

Chapter 4

Case Study: Calibration of Analogue Display Instruments

This chapter shows the applicability of the inspection concept proposed in the previous chapter in the case study of Analogue Display Instruments (ADI). This type of instrument serves as a demonstration since there are various different types of measuring instruments with innumerably different displays and layouts, but all of them have certain common properties which can be used to build up a specific description.

In the minimal case one instrument consists of an analogue or digital measuring device which measures a physical unit (like temperature) and a display where the measurement value is indicated on a scale. For example, the temperature is indicated on a thermometer by a scale and a pointer mounted on a bi-metal stripe (see Figure 4.1). The angle of difference of the pointer in reference to a scale indicates the temperature, i.e. the display of the instrument. If the different positions are marked using a reference thermometer, the instrument is calibrated, the display is in relation to an absolute measurement value. In Figure 4.1 the calibration would relate *n* to a specific temperature. Any measuring instrument has to be calibrated in order to measure absolute physical units.

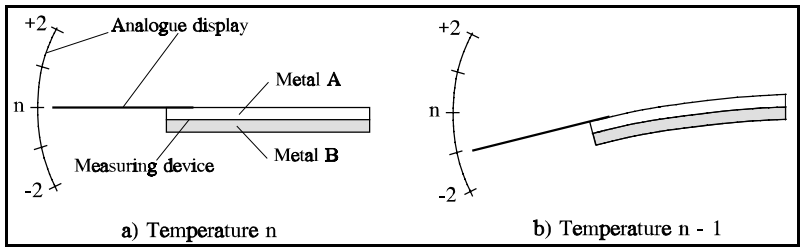


Figure 4.1 Example of a simple analogue display instrument

In this chapter we focus on the generation of the inspection model and the general, uninstantiated analysis graph, which can be used to construct an AVI system that inspects one specific type of instrument. Section 4.1 enumerates examples for ADI's. It discusses the motivation why this display method is still used and which attempts have been made to perform an automatic inspection of these instruments. Following the definition of the inspection model for ADI's in Section 4.2, the description language and the analysis graph performing the inspection are shown in Section 4.3 and Section 4.4, respectively. A summary concludes the chapter.

4.1 Examples of Analogue Display Instruments

There is a large variety of different ADI's, like clocks, watches, speedometers, fuel gauges, voltmeters, thermometers, barometers, pressure gauges, and scales. Every class of instrument itself has innumerable different layouts and forms, but all of these instruments have to be inspected and tested as well as calibrated by human workers after the manufacturing process.

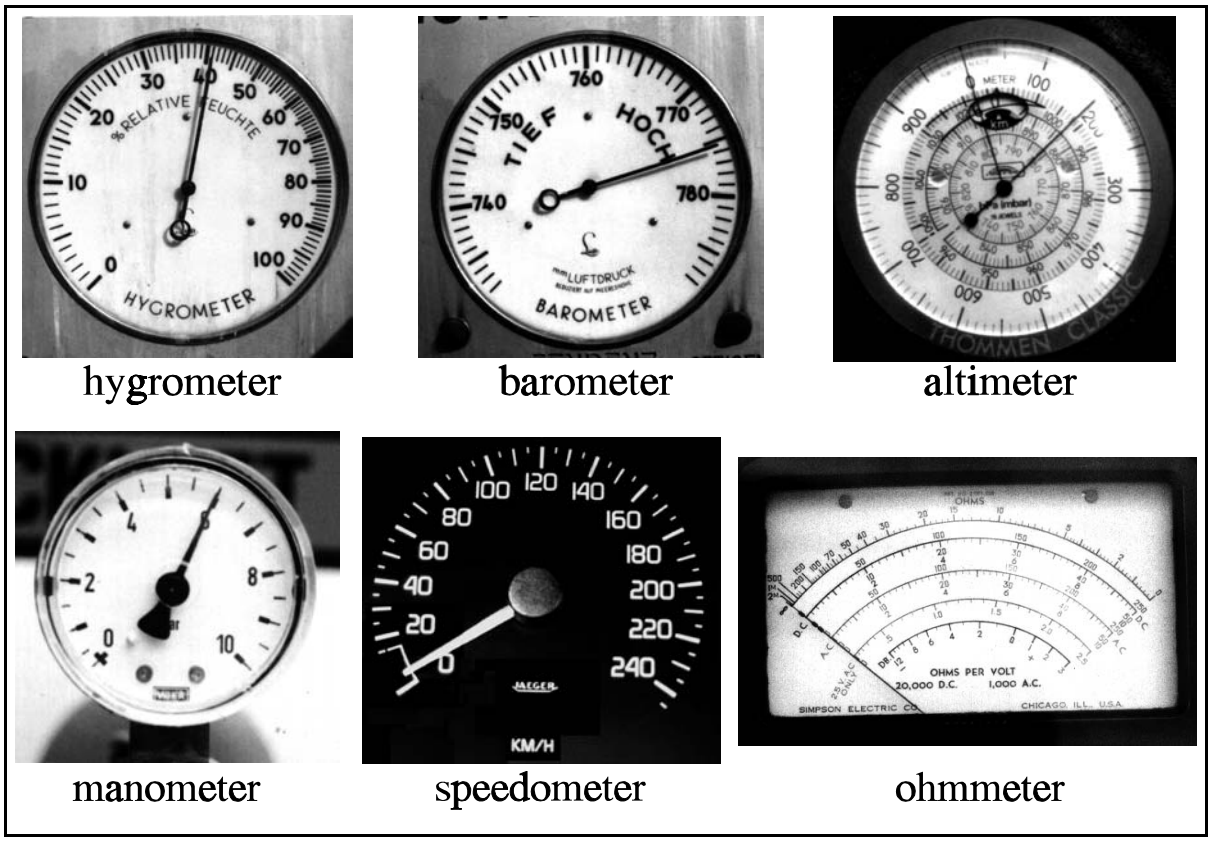


Figure 4.2 Examples for analog instruments displaying absolute measurement values

Measuring instruments can be subdivided into two classes:

- *Single-scale instruments*: have one pointer moving on one scale. Figure 4.2 shows 6 typical analogue single-scale measuring instruments. Manometers, barometers, altimeters and the like usually have a circular shape (see Figure 4.2) where the scale covers most of the circle, the pivot point lies within the visible display. Instruments measuring voltage, amperage or resistance usually have only a sector as display, the pivot point is not visible as on the ohmmeter shown in Figure 4.2 on the bottom right. Of course there are also other shapes that are used in the design of instruments like the speedometer shown in Figure 4.3a, but the pivot point of the pointer lies again within the scale. A completely different display technique is shown by the thermometer in Figure 4.3b, the rotation direction of the pointer is normal to the display, the pointer moves straight on the scale.

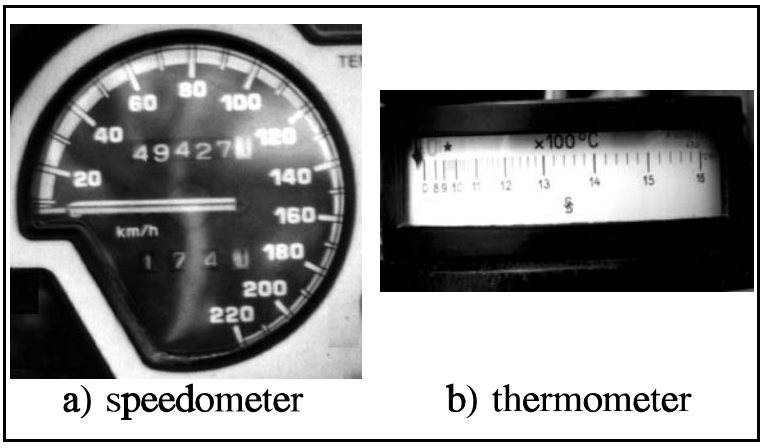
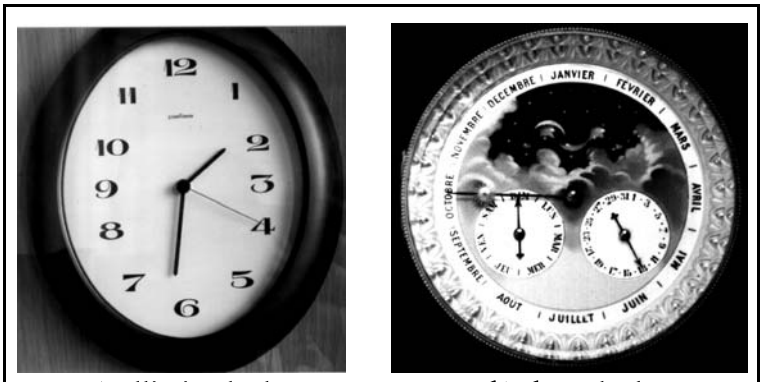


Figure 4.3 a) Speedometer in circular arc shape;
b) thermometer where pointer rotation is normal to display

- *Multiple-scale instruments*: have pointers moving on different scales and measure physical units like time, quantity of consumption (fluids, electricity, natural gas) for accounting, where differences of measuring units are computed. The reason for multiple scales lies in the possibility of displaying different units of the same physical unit on one instrument, just like a clock displays hours, minutes and seconds as different resolutions of time. Due to the spread of watches and clocks, they have innumerably different shapes, most of them are circular but other shapes like the elliptic clock shown in Figure 4.4a exist. Furthermore clocks may not only display the usual units of time; hours, minutes, and seconds but also weekday and date in analogue form as shown in Figure 4.4b.



a) elliptic clock b) date clock

Figure 4.4 Elliptic and date clock

4.1.1 Analogue versus Digital Display Instruments

Digital conversion of analogue signals is a well-studied area since computers are used to interpret measurement values. In applications where the measurement value gathered from a measurement device should not be displayed on the spot where the measurement takes place but at a distant observation desk, an analogue-digital conversion takes place. The measurement value is sent via electric signals in a cable to the distant observation center. So, in many cases, measurement values are already in digital format. One could argue that the measurement values are then displayed digitally, and an automation of the calibration process is not necessary since in a few years analogue displays will die out. But this is not the case.

Humans, in contrast to computers, do not have the same perception of measurement values if they are presented in digital numbers rather than in angles of pointers. This phenomenon was studied by researchers [EBE91], who are concerned with the design of user interfaces. Humans are trained to interpret geometric relations and are not used to interpret numbers which indicate the same measurement value. The reason for this difference lies in the different location of the information processing within the brain. Studies show (see for instance [HEC69]) that the interpretation of numbers takes place in the left hemisphere of the brain. From the neuropsychological point of view the reading of digital displays is similar to computing operations and is therefore part of the "logical" part of the brain capabilities, where the interpretation takes longer than in the right hemisphere of the brain, which is responsible for the emotional tasks. Therefore, digital displays are used if values have to be read accurately, since then two numbers are compared to determine whether they are the same.

For tasks for which it is necessary to know an approximate measurement value very fast, like driving a car, analogue displays are better suited. The "emotional" part in the right hemisphere of the brain is able to decide fast if the value is above a certain threshold, without computing how much the measurement value is above the threshold [EBE91].

Generally all measuring instruments that should be read easily, fast, but not accurately by humans, or should look impressive (like home weather stations or valuable watches), are in analogue form. Therefore, they will be produced further and an automatic inspection of this kind of product will still be necessary.

4.1.2 Inspection of Analogue Display Instruments

The inspection of analogue display instruments consists of two different tasks:

1. Verification whether the *appearance* is within the specifications, i.e. whether all elements of the display of the instrument are present and correct (referring to defined tolerances).
2. Checking the *functionality* (also referred to as testing - see Section 2.1) which consists of the calibration of the instrument, i.e. verifying if the instrument displays the same measurement value as a reference instrument.

Since there are two different classes of measuring instruments, there are two different methods of calibrating them, too:

- *Single-scale instrument calibration*: For calibration, *n* different measurement values are compared with a reference instrument. First a verification of the initial value (for instance the 0 value) takes place. This initial value verification checks the correct alignment of pointer to scale (for instance if a speedometer initially displays the value 0, or is in the rest position). Next, different measuring values are presented to the measuring instrument, resulting in different angles of the pointer. Each position is compared with a reference value and the deviation from the correct value is checked for each step.

  For speedometer inspection for example, these measurement values are produced by rotating the pointer as a result of a certain voltage, or in the case of an older analogue instrument, by rotating the shaft of the speedometer. The inspection of speedometer in cars takes place by putting the car onto rotating rolls imitating the movement and reading the measurement value of the instrument at different velocities.

- *Multiple-scale instrument calibration*: Each of the scales is checked whether the pointer is in the correct relation to the scale in the initial position. Following this static inspection, the functionality is inspected by taking the initial value, putting the measuring instrument into function, and stopping the function of the instrument. Subsequently, the end value is read and subtracted from the initial value. The result is compared with a reference value that has measured the same amount as the instrument under inspection. If the subtracted, relative measurement value is the same (within some tolerance) as the reference value, then the measurement instrument is declared fault-free (or calibrated).

  The calibration of a watch for instance consists in defining a start moment, starting a reference watch, and the verification that, after a predefined time interval, the watch under inspection displays the correct time. The initial time display is subtracted from the end time display and compared with the reference time.

To the author‘s knowledge no inspection system for multiple-scale instruments has been reported in the literature so far. Systems that inspect single-scale instruments exist. Since the vehicle industry (land, sea, air) is a huge branch of industry, systems for AVI of instruments used in these vehicles have been developed and reported in the literature. The first reported system was constructed to inspect only one type of instrument [HAT78]; if the style of the instrument was changed, the system had to be changed, too. Taking this drawback into consideration, a system called GAGESIGHT [BAI83] was developed by General Motors (Spark Plug Division) which was able to adapt the system to small layout changes by defining pass points of instruments interactively. Since the inspection system was developed by the manufacturer, the system was tailored to cover a specific variety of styles. The inspection was performed by checking the 0 position only, i.e. the correct alignment of pointer and scale.

A system that measures and verifies the relative accuracy of an instrument to a specified set of analogue input signals, was presented by Dyer [DYE83]. This system used the Hough transform and a series of images with different positions of the pointer to determine the pivot point of the pointer and different positions of the pointer by subtracting the images from one another. Due to the nature of the Hough transform, this method can only be used for needle-typed pointers and actually was only used to inspect voltmeters without inspecting the layout and registration of pointer and scale.

Recently a system was presented that inspects complete gauge panels using a CNC machine to move the camera [LIU95]. The inspection of the gauges consists of inspecting the rest position of the pointer by computing the overlap between pointer and scale mark of the rest position for one specific gauge panel used in Chinese truck production.

4.1.3 Coupled Pointers in Multiple Scales

The design of multiple scale instruments differs, either the scales are overlaid, like on a watch, or they have separate scales (like the date-watch in Figure 4.4b). If they display the same measurement unit on different scales, then pointers are coupled. Coupled pointers have a specific feature: the previous pointer defines the position of the following pointer. The pointer displaying the lower valence (e.g. 1) defines the position of the pointer displaying the higher valence (e.g. 10). If the lower pointer displays 0 for instance, the pointer with the higher valence has only a limited number of possible positions as shown in Figure 4.5a. If the lower pointer displays 5 for instance, the pointer with the higher valence has a different, limited number of possible positions than in the previous case (Figure 4.5b). The immediate perception of the highest valence is the reason for the design of different scales on one instrument. If it is necessary to read the measurement value more precisely, the lower valence (a different scale of the measuring unit) can be checked. Furthermore, small geometric differences at a higher valence correspond to a geometric difference that are by a factor larger at a lower valence.

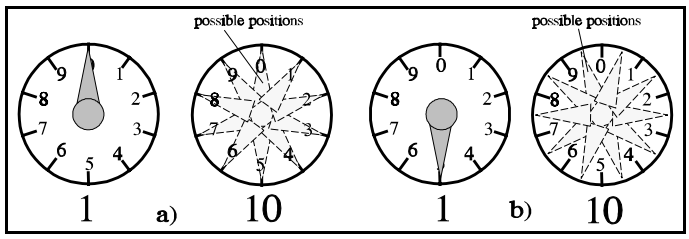


Figure 4.5 Reduced search space

In decimal coupling, every measuring unit contributes one digit to the measurement value. The reading of the values indicated by the pointer position starts with the lowest valence in

order to have the correct value instantly. Suppose a clock displays 11:58 h. If the higher valence is read first, the value of the hour handle would first be supposed to display 12, after reading the minutes handle the value of the hours handle would be corrected to 11. Therefore, all of the instruments with coupled pointers are read from the lowest to the highest valence if it is necessary to read them precisely. In the case of the clock displaying 11:58, a human not interested in the correct time would interpret the instrument as follows: "It is close to twelve".

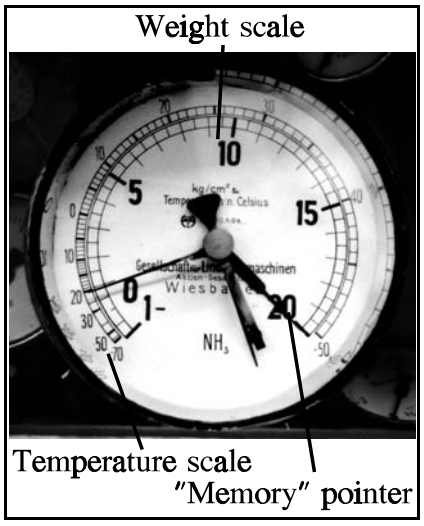


Figure 4.6 Instrument with two overlaid scales, indicating different units

Note that not all multiple-scale instruments have coupled pointers. Especially for monitoring purposes there are ADI's displaying different measuring units on overlaid scales. Figure 4.6 shows an example, where one scale displays the temperature from -50 to +70 degree Celsius (outer scale), the second scale displays the weight from -1 to 20 kg/cm² (inner scale) of NH_3. Furthermore, this instrument has a "memory" pointer which can be set manually, used for instance for difference calculations. Since such instruments can be regarded as overlaid single-scale instruments, they are calibrated using the single scale calibration technique.

4.2 Inspection Model Generation for ADI's

Following the concept proposed in Chapter 3, the first step in developing an AVI system is to generate an inspection model. This model contains the specific parameters of the object necessary to perform the inspection. To construct a general ADI inspection system we decided to use a series of different types of ADI's. In deviation from the proposed concept we do not use a series of images per object and select a representative image but rather, since we are not constructing a specific inspection system, use a series of different ADI images.

4.2.1 Generic Detection of ADI Primitives

ADI shape features can be represented by simple (parametric) primitives such as lines, rectangles, circular arcs, circles, and characters. For any of these primitives it is necessary to define their generic parameters in order to perform the generic detection (see Section 3.3.2) and to define a standard interface for detection. In the following we give some examples how primitives could be defined and further ones added. All of the generic parameters are defined in an object-centered coordinate system referred by the origin in image coordinates (Explicit notation is used for parameters as in Figure 4.7 and in relation formulas):

Line: origin (p_{l0}), length (l), orientation (α_{l0}), thickness (t_l)
Rectangle: origin (p_{r0}), width (w_r), height (h_r), orientation (α_{r0}), thickness (t_r)
Circular Arc: origin (p_{c0}), radius (r), range (α_{cr}), orientation (α_{c0}), thickness (t_c)
Circle: Special case of circular arc with $\alpha_{cr} = 360°$

Round. Rect.: origin (p_{rr0}), width (w_{rr}), height (h_{rr}), orientation (α_{rr0}), offset (o_{rr1},..,o_{rr4}), thickness (t_{rr})

Ellipse: origin (p_{e0}), first axis (a), second axis (b), orientation (α_{c0}), thickness (t_e)

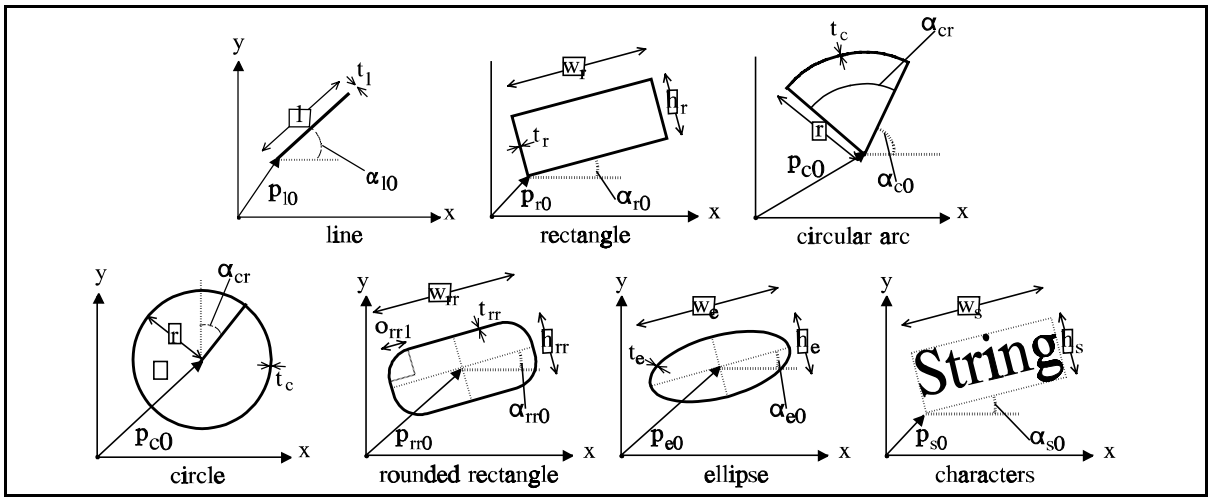


Figure 4.7 Generic parameters of primitives

For non-parametric primitives like strings and logos, parameters describing the bounding rectangle are used:

Characters: origin (p_{s0}), width (w_s), height (h_s), orientation (α_{s0}), content (bitmap)

The shape features are represented as primitives in the description. In the image, primitives are represented as features (see Section 3.1) and feature detection algorithms are used to detect features and their parameters in the image.

Since generic detection is computationally expensive, all geometric primitives not occurring in the sample images are left out individually, i.e. if the image does not contain rectan-

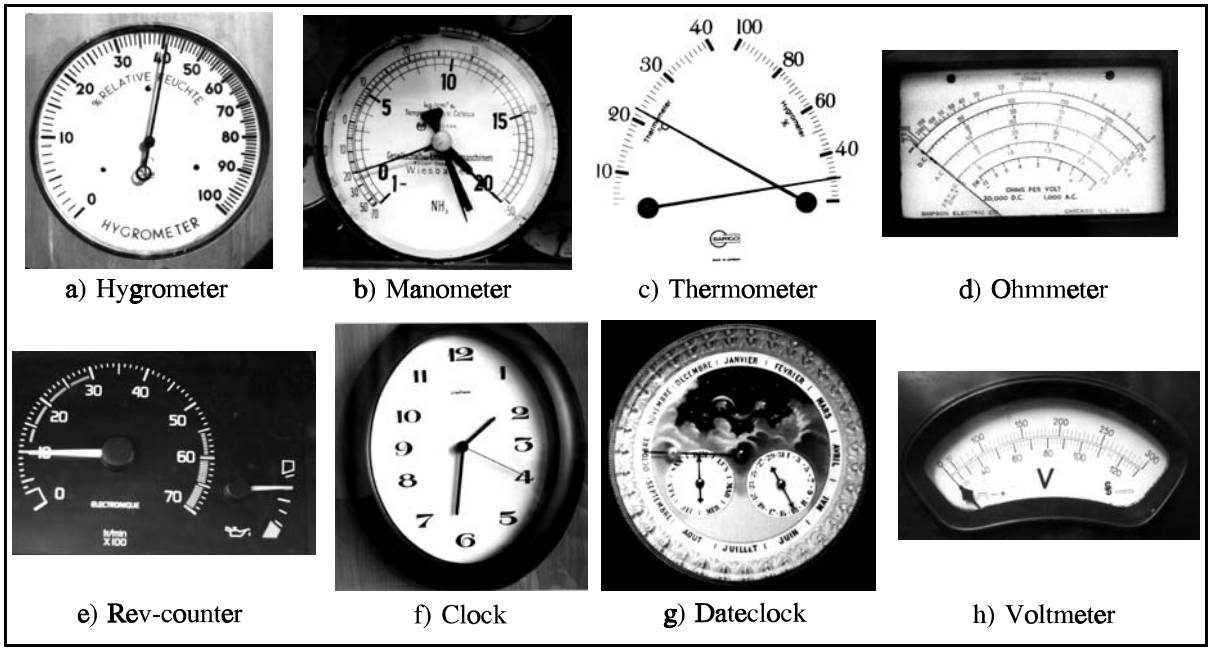


Figure 4.8 Intensity images of different ADI used as samples for generic detection of primitives

gles, the primitive is not searched for in the image. Figure 4.8 shows the intensity images of the instruments used for generic detection of the primitives. The examples were chosen to represent circular instruments with one (Figure 4.8a), two (Figure 4.8e), two overlapping (Figure 4.8b), and three circular scales (Figure 4.8g); a rectangular instrument with one circular scale (Figure 4.8d); an elliptic instrument (Figure 4.8f); and 2 free shape instruments with circular arc scales (Figure 4.8c and Figure 4.8h).

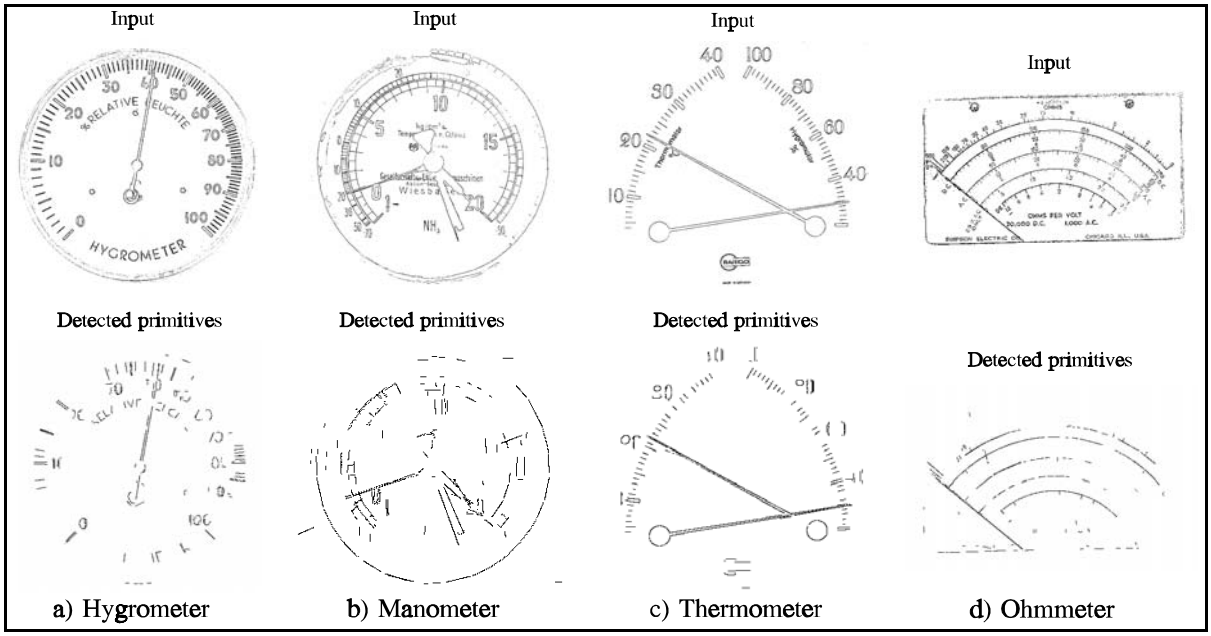


Figure 4.9 Primitives detected by generic detection

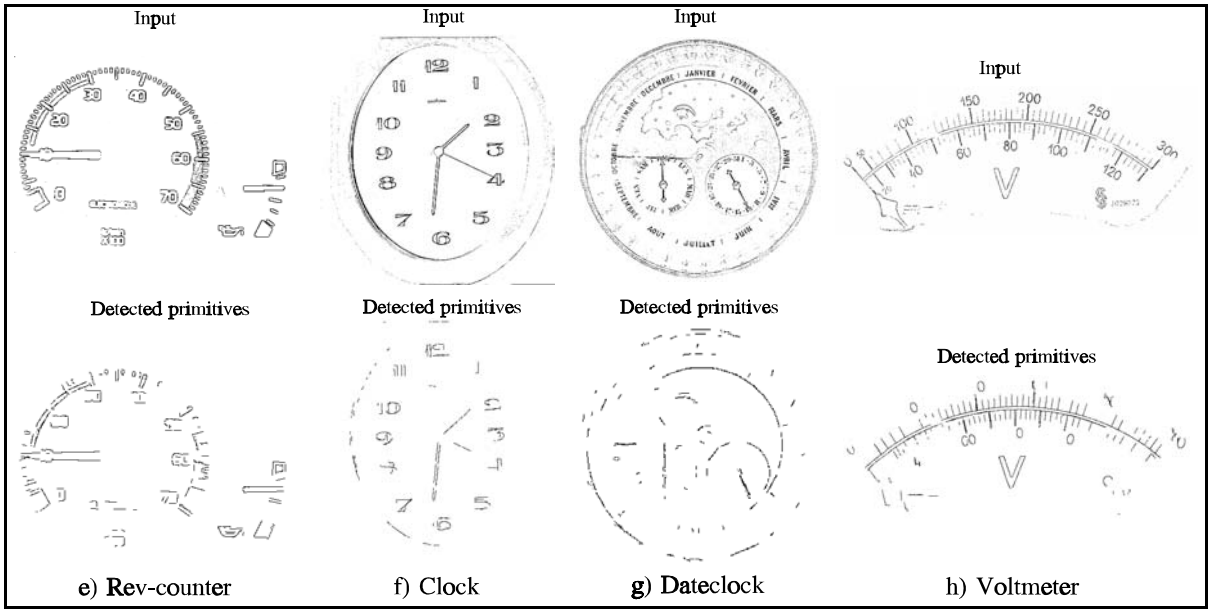


Figure 4.10 Primitives detected by generic detection

These images are the input of a canny edge detector operation producing an edge data set. This set contains all edge points in a local neighborhood reaching a certain threshold of gradient strength. The threshold was set high (0.8, where 1 indicates the maximum strength) to reduce the number of candidate points in the data set. The data set, each element containing the

edge direction and the position, was the input for the generic detection. The input (inverted edge data set) and the result of the generic detection are shown in Table 4.1 and in Figure 4.9 and Figure 4.10, respectively. Note that for the primitive line a minimum length can be selected, for lines shorter than the threshold no hypotheses are constructed.

Type of instrument	Image size (w x h, in pixels)	Quantity of detected primitives	
Hygrometer, Figure 4.9a	400 x 417	Lines: 97	Circles: 13
Manometer, Figure 4.9b	300 x 296	Lines: 96	Circles: 13
Thermometer, Figure 4.9c	400 x 377	Lines: 121	Circles: 8
Ohmmeter, Figure 4.9d	500 x 316	Lines: 80	Circles: 16
Rev-counter, Figure 4.10e	300 x 271	Lines: 101	Circles: 26
Clock, Figure 4.10f	457 x 489	Lines: 87	Circles: 21
Dateclock, Figure 4.10g	300 x 295	Lines: 74	Circles: 25
Voltmeter, Figure 4.10h	711 x 297	Lines: 118	Circles: 25

Table 4.1: In- and output of generic detection

4.2.2 Definition of ADI Primitives

The primitives found by the generic detection are the basis for the interactive construction of the description. The generation starts with a definition of the primitives of ADI's, which are not necessarily the same as the primitives for generic detection, since non-parametric primitives are added in this definition step.

Three primitives describe analogue measuring instruments (see Figure 4.11):

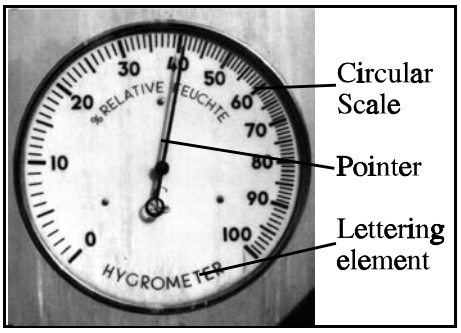


Figure 4.11 Primitives of a hygrometer

- *Pointer*: A pointer can have any symmetric shape such as line, triangle, rectangle or a combination of them. In addition, pointers that rotate have a circle at their center of rotation (see Figure 4.11). The shape is defined by a primitive, a combination of them, or in the case of a shape that is not easily represented by primitives, by a bitmap, containing one half of the shape and the medial axis.

- *Scale*: The shape of a **scale** depends on the motion of the pointer, scales with rotating pointers have the shape of a **circle** or a **circular arc**, (see Figure 4.12a). Pointers moving straight have rectangular scales (see Figure 4.12b). Scale captions are considered to be part of the scale.

- *Lettering element*: Such an element carries information about the measurement and the orientation, this includes all writings such as unit, company name, and emblem of maker.

For the example of generating the description for ADI, primitives found by generic detection have to be related to shape features of the instruments. This is performed by filtering the result of generic detection in that way that only lines of a certain length and circles with certain radii are selected. The user interactively assigns the remaining primitives to shape features. Table 4.2 shows the result for the selected examples.

Type of instrument	**Imagesize (w x h)**	**Shape features**
Hygrometer, Figure 4.9a	400 x 417	Instrument (outer circle): (199,209), 193 pixel Instrument (inner circle): (196,212), 181 pixel Pointer: (191,265)-(199,221) Pointer right border: (203,203)-(231,38) Pointer left border: (202,185)-(213,131), (214,112)-(222,74)
Manometer, Figure 4.9b	300 x 296	Instrument: (151,139), 133 pixel 1st Pointer: (158,167)-(181,230), (165,165)-(188,216) (137,150)-(54,178), (132,149)-(54,174) 2nd Pointer: (162,146)-(181,165)+ (177,156)-(195,184)+ (187,176)-(195,188)+ (187,175)-(198,182) 3rd Pointer: (230,114)-(253,106)
Thermo-meter, Figure 4.9c	400 x 377	Left pointer: (66,141)-(278,258), (64,142)-(268,257) Right pointer: (96,286)-(265,260) + (280,258)-(364,242), (97,290)-(275,260) + (284,260)-(369,245) Left circle: (81,291), 15 pixel, Right circle: (317,282), 15 pixel
Ohmmeter, Figure 4.9d	500 x 316	Pointer: (52,157)-(200,279) Lowest arc: (165,213)-(361,223) Lowest arc, center, radius: (245,296), 145 pixel
Rev-counter, Figure 4.10e	300 x 271	Pointer rev-counter: (192,142)-(57,144) + (38,144)-(8,145) (130,157)-(56,154) + (37,154)-(26,153) Pointer oil gauge: (351,196)-(410,196) (355,203)-(411,203) Inner arc rev counter: (153,151), 123 pixel Outer arc rev counter: (153,151), 130 pixel Scale lines rev counter (selected): (39,233)-(53,221)-(40,199) Scale lines oil gauge: (378,247)-(366,273)-(387,275)-(396,254) (394,178)-(414,158)
Clock, Figure 4.10f	457 x 489	Pointer hour: (238,224)-(290,174) Pointer minute left: (228,239)-(220,318) + (220,328)-(218,353) Pointer minute right: (232,245)-(224,324) + (224,332)-(220,350) Pointer second: (257,240)-(272,248) + (275,248)-(291,256)
Dateclock, Figure 4.10g	300 x 295	Instrument, outer border: (150,145), 138 pixel Instrument, inner border: (150,145), 98 pixel Circle right: (195,178), 39 pixel Pointer right: (197,181)-(210,207) Pointer left: (106,142)-(107,196)

Voltmeter, Figure 4.10h	711 x 297	Upper arc: (45,187)-(179,100) + (202,92)-(650,166), Lower arc: (51,191)-(202,98) + (210,95)-(538,106), Upper arc, center, radius: (324,470), 460 pixel Lower arc, center, radius: (324,470), 454 pixel

Table 4.2: In- and output of generic detection

The nodes of the description contain the shape features of the object. They have the following generic parameters, all of them are defined in an object-centered coordinate system referred to by the origin in image coordinates (Explicit notation is used for parameters that are used in Figure 4.12 and in relations R1-R9 in Section 4.2.3):

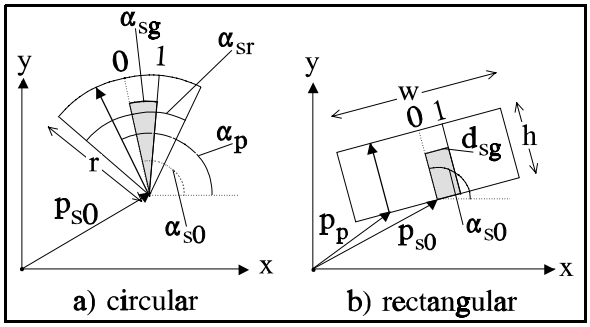


Figure 4.12 Pointer and scale

Measuring instrument:

- type (for every measuring instrument type there must be a description that allows a distinction of the measuring instrument)
- shape (shape of the instrument: circle, rectangle, triangle, free form ...)
- origin (origin of the object centered coordinate system in x,y coordinates)
- size (size of the instrument in pixel and millimeters)
- number of measuring units (n)
- number of lettering elements (m)
- absolute measurement value (mv_n, measurement value in measurement unit, if the instrument has multiple scales it states if they are coupled or not)

Measuring unit:

- normalized measurement (c, states the relative measurements in units)
- measurement digits (e, states which digit of the unit is displayed)
- unit (u, unit of measurement of the scale)
- offset (o, origin of measurement)

Scale: symmetric scales can have three different shapes with following parameters:

- type: circular, or elliptic, or rectangular
- size: radius(r), or w_e,h_e, or height(h)
- origin p_{s0}
- orientation α_{s0}
- graduation: α_{sg}, or d_{sg}
- range: α_{sr}, or width (w)

Pointer:

- type (circular or rectangular)
- shape (bitmap or line with length and offset or rectangle with length and with)
- origin
- position (α_p or p_p, variable)

Lettering:
- type (bitmap, string, geometric primitive)
- origin (p_l, in x,y coordinates)
- content (bitmap, string, geometric primitive)
- size (width and height in pixels)

Measuring space

One constituent is not contained in an intensity image, but implicitly given with any measuring instrument - the measurement space. It defines which unit and which value the **measuring unit**, consisting of a scale and a pointer, displays and at which sample rate the displayed value can be read. The measuring space also defines the absolute measurement value the **measuring instrument** displays, as a combination of all measuring units. Note that we assume that one measuring instrument can measure only one physical unit.

4.2.3 Definition of ADI Relations and Imaging Parameters

Following the definition of primitives, relations between them complete the description. The basis for the generic parameters of the spatial relations are again the results gained from generic detection. All other relations, like measurement value, have to be added. For analogue display instruments following relations (conditions, equations,..) among primitives are defined in the grammar of the description language, further rules can be added [SAB94a]:

At the level of a measuring unit:

R1: type(pointer) = type(scale).

R2: For circular scales only: origin(pointer) = origin (scale).

R3: (unit independency) $c = \frac{\alpha_p - \alpha_{s0}}{\alpha_{sg}}$ or $c = \frac{p_p - p_{s0}}{d_{sg}}$,
where α_p and p_p denote the position of the pointer, α_{s0} denotes the orientation, p_{s0} the origin, and α_{sg} and d_{sg} the graduation of the scale (see Figure 4.12).

R4: (range constraint) $|\alpha_p - \alpha_{s0}| \leq \frac{\alpha_{sr}}{2}$ or $|p_p - p_{s0}| \leq \frac{w}{2}$,
where α_{sr} and w denote the range of the scale (see Figure 4.12).

At the level of a measuring instrument:

R5: Measuring units on the same measuring instrument are related through their position. Note that different measuring units may be overlaid at the same position.

R6: Lettering elements are related through their position.

R7: (measurement digits) $e_i = c_i * u_i + o_i$, where $i = 0..n$ denotes the measuring unit u and o the origin of measurement.

The following rule is only used for measuring instruments that have more than one measuring scale and coupled pointers:

R8: (measurement) $$mv_n = \sum_{i=0}^{n} \frac{\lfloor e_i \rfloor}{\prod_{j=0}^{i} u_j} ,$$

where $u_0 = 1$, e denotes the measurement digit, and u the measuring unit.

Imaging Parameters

The image coordinate system is a projection of the world coordinate system onto the image plane. For industrial inspection applications we assume that object and image plane are parallel. The image acquisition parameters define the image size, the size of pixels in the image and the sampling accuracy. To simplify the analysis process we assume that the imaging instrument has quadratic pixels. Therefore, two imaging parameters are defined:

- image size (x,y)
- sampling accuracy (a)

There is only one relation for determining the sampling accuracy:

R9: $$a^2 = \frac{size(measuring\ instrument)}{imagesize(measuring\ instrument)}$$

Besides geometrical distortions and occlusions the analysis does not depend on other acquisition parameters, because only relative measurements are used. Furthermore good illumination conditions are required. In order to ensure an accurate analysis, the image of the object should have high contrast and there should not be shades or reflections in the image.

4.3 Description Language for ADI's

Following the definition of the primitives, relations and generic parameters, the graph forming the description language can be defined (see Section 3.3.3). The hierarchical structure is built up in a top-down manner; it starts with the abstract class of measuring instrument and ends up in the primitives of the measuring instrument. Figure 4.13 shows the description for ADI's including all generic parameters of the primitives (nodes) and relations (arcs).

The primitive "Measuring instrument", which is subdivided into n "Measuring units" and m "Lettering" elements is the root of the graph. A common measuring instrument like a hygrometer has only one measuring unit, but a clock for instance normally has 3 measuring units. Therefore, this sub-hierarchy is necessary. Every measuring unit is subdivided into one "Pointer" and one "Scale", all the previously defined primitives are the leaves of the graph. This general description language allows a representation of any analogue display instrument in a separate graph. Weights and tolerances are added during the analysis step.

Figure 4.14 shows examples for specific descriptions for the instruments hygrometer (Figure 4.14a), manometer (Figure 4.14b), voltmeter (Figure 4.14c), and dateclock (Figure 4.14g). These instruments were selected from Figure 4.8, since their description shows specific differences. Most of the single-scale ADI's have the simplest description like the one shown in Figure 4.14a. The scale is circular (border of the instrument), and the pointer is a

line of a certain length. Furthermore, to be able to distinguish similar instruments and to determine the orientation, selected lettering elements are described, too. In the case of the hygrometer these elements are: "caption", which is the instrument type (HYGROMETER) indicated on the bottom of the instrument (see Figure 4.11), the left and the right dot within the scale, and the scale caption 100. Note that the selection of lettering elements is user dependent and could therefore also result in other or more lettering elements. The specific selection of lettering elements for this example was made from the point of view of detection.

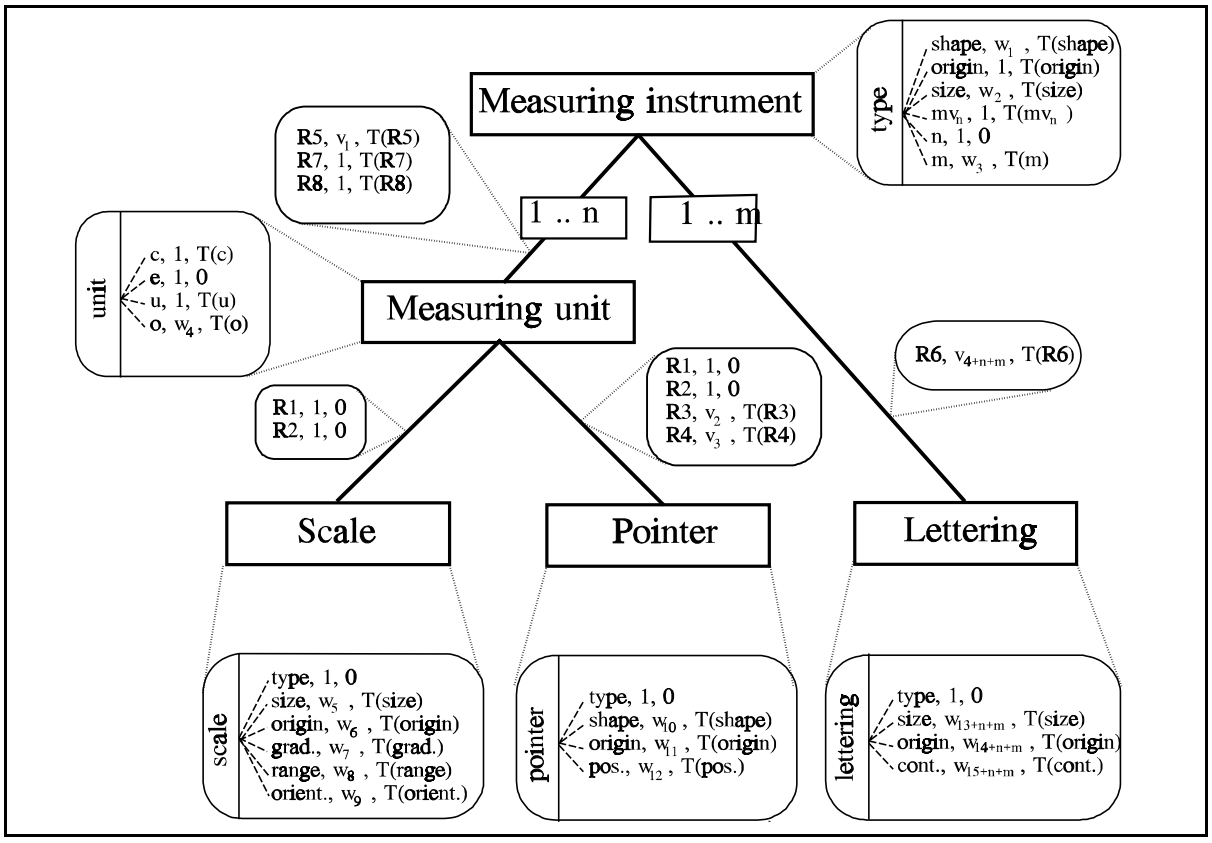


Figure 4.13 Description language for analogue display instruments

Figure 4.14b shows an example for a single-scale instrument with two overlaid scales, a manometer with a temperature scale (see Figure 4.8b). The description results in two different measuring units, temperature and pressure, the only difference to the hygrometer description. The voltmeter example (Figure 4.14c) was chosen because there are different scales on one instrument, having only one pointer (the instrument can only measure one unit at a time). Two scales (DC and AC) with the same pointer were defined. The graph for the dateclock (Figure 4.14d) is an example for multiple-scale instruments, the graph of a normal clock would look the same, only the generic parameters would differ.

The graphs in Figure 4.14 do not contain generic parameters of primitives for reasons of clarity. In the following, two examples for generic parameters are given. The description of the manometer and the dateclock serve as a demonstration. Table 4.3 shows the generic parameter of the root node, *type, n, m, mv* have to be defined by the user.

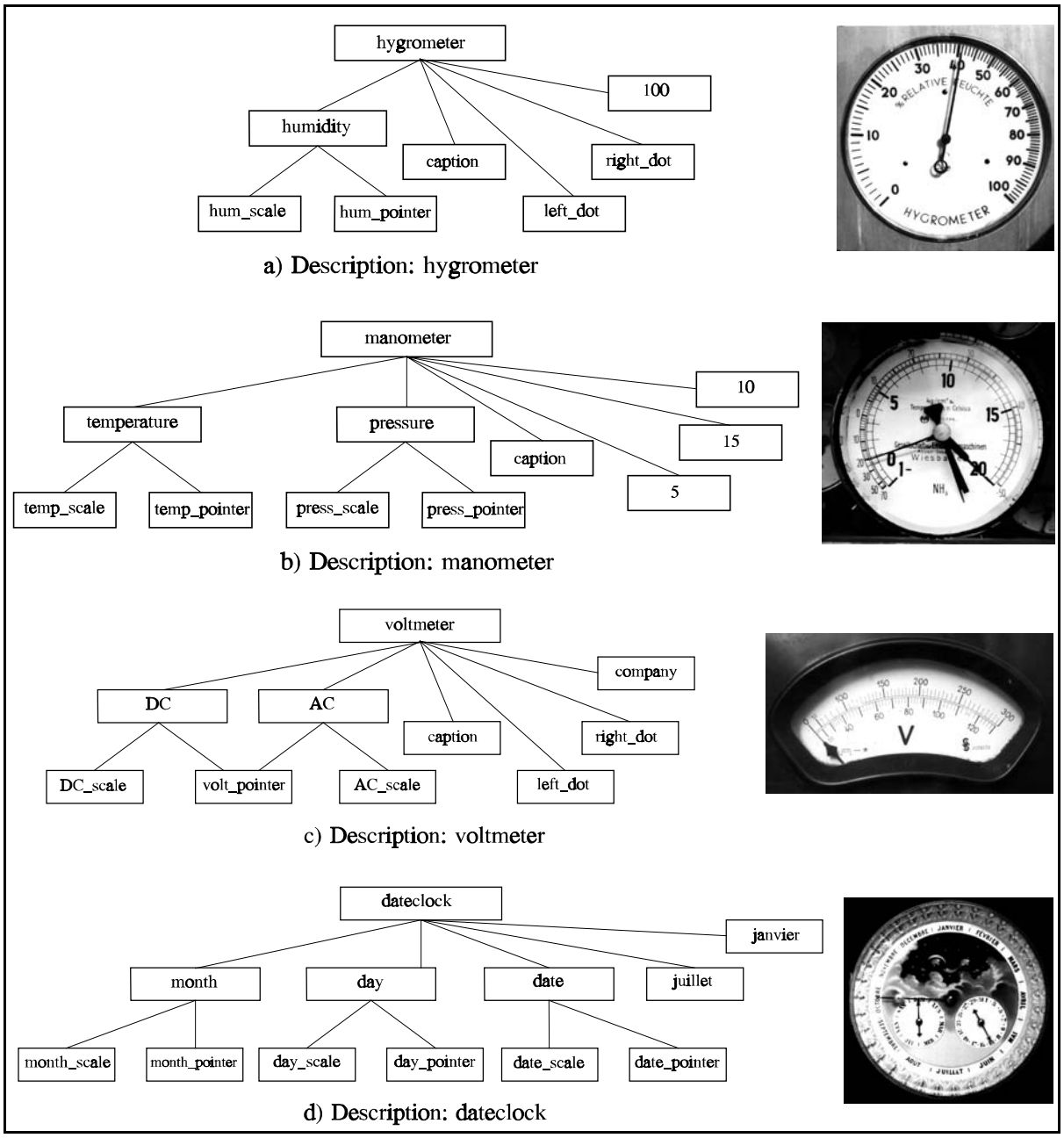


Figure 4.14 Examples for specific descriptions

Measuring instrument:

type	shape	origin (x,y)	size (pixel,mm)	*n*	*m*	*mv*
manometer	circle	0,0	327,900	2	4	separate
dateclock	circle	0,0	350,2200	3	2	separate

Table 4.3: Generic parameters of root nodes

Measuring unit:

Table 4.4 shows the generic parameters of the manometer´s measuring units (2 units), Table 4.5 for the dateclock (3 units). All parameters have to be defined interactively.

unit	*c*	*e*	*u*	*o*
1	1/22	1	kg/cm²	110°
2		1	°Celsius	80°

Table 4.4: Generic parameters of measuring units manometer

unit	*c*	*e*	*u*	*o* (°)
1	1/12	2	month	0
2	1/7	1	day	0
3	1/31	2	date	0

Table 4.5: Generic parameters of measuring units dateclock

Scales:

The result of the generic detection is again used to determine type, size, and origin; α_{s0}, α_{sg}, and α_{sr} have to be added. Table 4.6 shows the parameters for the manometer scales, Table 4.7 shows the parameters of the dateclock scales. Note that the measuring unit 2 in Table 4.6 has a logarithmic scale, therefore, the graduation is non-linear ($y= a^{x+c}-k$, where y is the angle to the measuring origin, a=1.03152 the basis, x the unit in °C, c=129.924 the offset to the scale origin, and k=-6.42108 the offset to the measuring unit, $x= log(y-k)/(log(a)-c)$.

unit	type	size	origin (x,y)	α_{s0}	α_{sg}	α_{sr}
1	circular	224	0,0	110	12,6	280
2	circular	243	0,0	80	a^{x+c}	277

Table 4.6: Generic parameters of scales manometer

unit	type	size	origin (x,y)	α_{s0}	α_{sg}	α_{sr}
1	circular	280	0,0	0	30	360
2	circular	90	-105,-77	0	51,4	360
3	circular	90	105,-77	0	11,6	360

Table 4.7: Generic parameters of scales dateclock

Pointer:

There are three ways to define a pointer: bitmap of the shape of the pointer, geometric primitive line, or rectangle. In Table 4.8 the pointer definitions for the manometer is shown, one pointer is represented as bitmap, the other as line with certain length and offset from the pivot point. The actual position of the pointer α_p is given here, for inspection this parameter has to be determined. The parameter α_p in Table 4.8 no is not defined for unit 2, since the pointer is not within the scale. Table 4.9 shows the pointer parameters for the dateclock.

unit	type	shape	origin (x,y)	α_p
1	circular	line, 214, 24	0,0	107
2	circular	bitmap	0,0	-

Table 4.8: Generic parameters of pointer manometer

unit	type	shape	origin (x,y)	α_p
1	circular	bitmap	0,0	271
2	circular	bitmap	-105,-77	355
3	circular	bitmap	105,-77	150

Table 4.9: Generic parameters of pointer dateclock

Lettering:

Table 4.10 shows the generic parameters of the 4 lettering elements of the manometer. Lettering elements can either be defined by bitmaps, geometrical primitives like line and rectangle or by characters. For the example of the manometer one bitmap lettering element (NH_3, the chemical term for ammoniac) and 3 character representations were chosen. The parameters origin and size have to be determined interactively.

lettering	type	origin (x,y)	size	content
1	bitmap	-28,-190	50,40	matrix
2	char	-158,66	20,50	"5"
3	char	3,131	40,50	"10"
4	char	132,31	40,50	"15"

Table 4.10: Generic parameters of manometer lettering elements

lettering	type	origin (x,y)	size	content
1	bitmap	-47,233	90,30	matrix (Janvier)
2	bitmap	-47,260	90,30	matrix (Juilet)

Table 4.11: Generic parameters of dateclock lettering elements

4.4 Analysis Graph for ADI's

The next step in generating an inspection system for ADI's is the construction of the analysis graph. Note that for this example no actual image series of a specific ADI is used, but a set of different images of ADI. Therefore, no tolerances and weights are defined in the description and detail relations will not be defined in the analysis graph. The general analysis graph (see Section 3.5) is applied to the specific problem. In- and output parameters are described at the top level of the graph only, to simplify the graph. This general concept is used for the definition of the analysis graph for ADI's, resulting in the graph shown in Figure 4.15.

To get an impression of the dataflow, Figure 4.16 shows the dataflow between the leaves of the graph in Figure 4.15. Further subdivisions of individual nodes are shown. Since the graph will not be instantiated for a specific instrument, detail relations are not shown. In the following, the nodes of the analysis graph for ADI's are described.

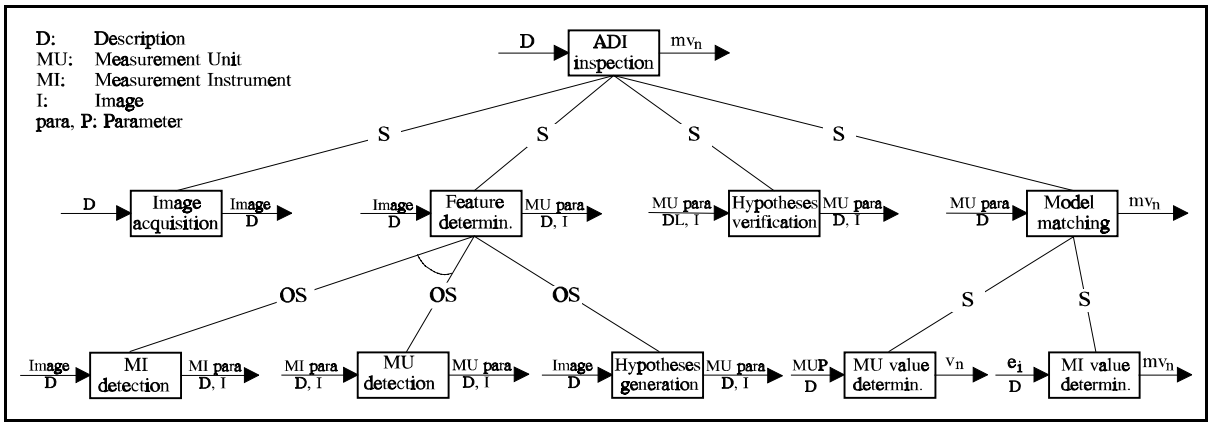


Figure 4.15 General analysis graph for ADI's

Image acquisition:
To simplify the analysis, we assume that the image contains only a single object. Prior to further computation, the actual image has to be checked whether it satisfies the assumption. This includes a control of the contrast and the histogram. If there are strong variations to the statistical mean image of the inspection series, there is an error in the illumination, acquisition, or positioning and the analysis is stopped. The *image* of the ADI in *x,y* coordinates (rows and lines in image coordinates) is the output of this node. This image is the input for the measuring instrument detection, or optionally for the hypothesis generation if imaging parameters are fixed. Note that only newly determined or computed parameters are explicitly mentioned, all others are implicitly available, like the parameters of the description which are accessible to all nodes of the graph since they are the input of the root of the graph.

Measuring instrument detection:
If position and size of the measuring instrument in the image depend on actual image acquisition parameters and positioning of the instrument, this step has to be carried out. The shape of the instrument, defined in the description, is looked for in the image with regard to topological, radiometrical and geometrical features. Depending on the shape, the node is optionally subdivided into the detection of the geometrical shape like rectangle detection, circle detection, and freeform detection. The leaves of the analysis graph (Figure 4.16) are the defined interface to feature detection algorithms. All of the nodes are subdivided into different possi-

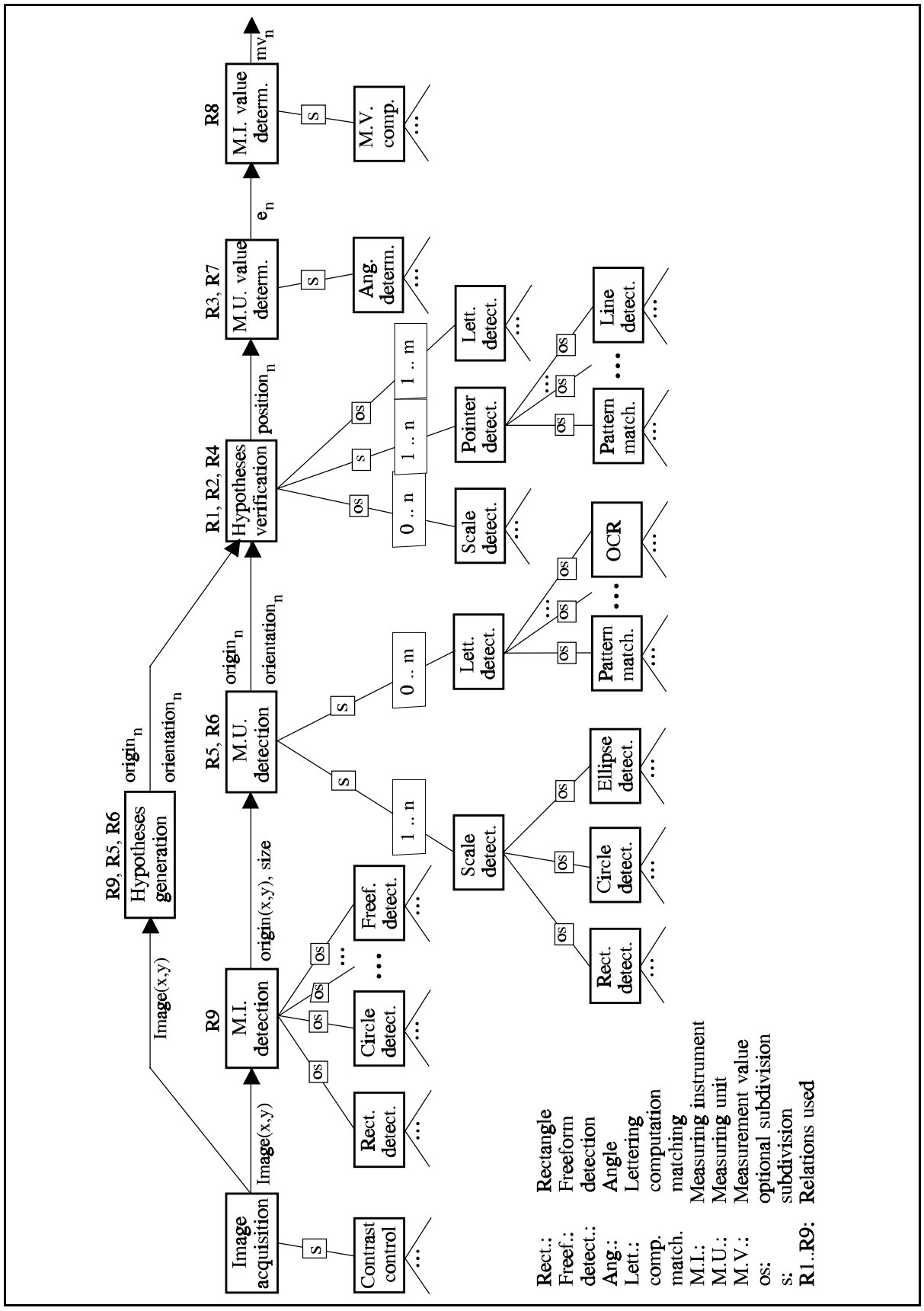


Figure 4.16 Analysis graph for ADI's (no detail relations shown)

ble detection algorithms and detail relations. In the sub-graph *rectangle detection* for instance all algorithms to detect rectangles available to the system are included (as in Figure 3.10 for example), further ones can be added (examples for detection algorithms are described in the next chapter). The *origin(x,y)* in image coordinates of the object-centered coordinate system and the *size* (in image and world coordinates using R9) of the detected object are the outputs of this node.

Measuring unit detection:
Detection and localization of measuring units are carried out in the limited area within the measuring instrument, defined by the hypothesis constructed by the measuring instrument detection. First the *1..n* scales are looked for using relation R5 because they cover a larger area in the image than pointers and their position does not change within the instrument. In different subdivisions of the measurement unit detection. Note that the search space for the corresponding pointer can be restricted by the region defined by the detected scale. Furthermore the measuring unit detection may be subdivided into *0..m* lettering element detections using relation R5 and R6. This is necessary to determine the orientation of the scale if for instance there is only one scale on the instrument. This node supplies the specific origin and orientation for up to *n* scales and up to *m* lettering elements.

Hypotheses generation:
If the image acquisition parameters are fixed, hypotheses about specific parameters of the measuring instrument and measuring units are generated without time consuming search in the image. Scales and lettering elements are induced to be on a specific position with a specific orientation within the image using R5 and R6 of the description language. The output is the same as that of the unit detection.

Hypothesis verification:
For both subdivisions of feature determination, a verification of the generated hypotheses is necessary. In order to answer the question "Are the measuring units on the right places on the measuring instrument ?", the identified type is verified by checking the induced position of the lettering elements (and possibly scales) in the image. The aim of the verification is to find out which of the candidates are elements of a measuring unit and which are not by applying R1, R2, and R4. Furthermore all primitives which were not detected in the previous nodes are detected. In the specific case of ADI's this detection includes pointer detection. The origin of the pointer is the center of the scale, the search space is limited by the corresponding scale. The possibly corrected positions and orientations of the scales and the specific positions of the pointers are the output of this node.

Measuring unit value determination:
The value for each measuring unit is determined by applying R3 and R7. The value e_i, $i=1..n$, for each measuring unit is the result of this node of the analysis graph.

Measurement value determination:
The last element of data flow in the analysis graph is the determination of the value mv_n with the help of equation R8, having the final result as the output.

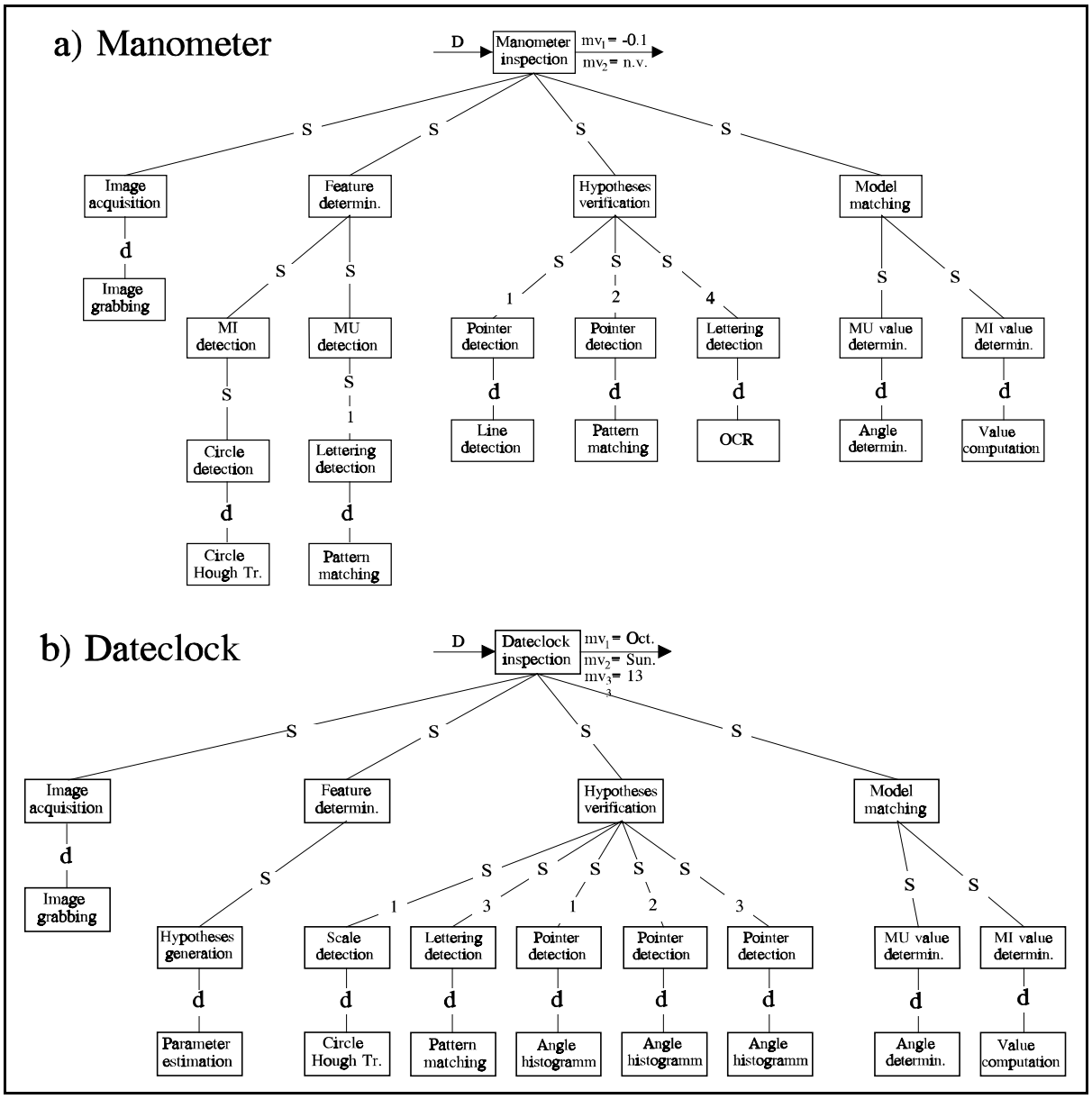


Figure 4.17 Analysis graph: a) manometer b) dateclock

The analysis graph describes the analysis for any ADI. In the example of the two instruments described in the previous section, an instantiation of the analysis graph is shown for the manometer and the dateclock in Figure 4.17. For manometer inspection, the imaging parameters are supposed to be unknown, the inspection of the dateclock is supposed to have fixed image acquisition. Therefore, the analysis graph of the manometer (Figure 4.17a) has two subdivisions of *feature determination*. First the manometer is detected using a circle detection, defining the origin of the object-centered coordinate system. Since the origin of the two scales is the same as the origin of the instrument, only one lettering element has to be looked for to determine the orientation of the scales (in this example, the lettering element "NH_3" is looked for using a pattern matching technique along a circle in a known distance). To verify the result, the lettering element "15" is checked by *hypotheses verification*, i.e. using an OCR algorithm. If the test is successful the positions of the 2 pointers are determined by a line detection algorithm and a pattern matching technique, respectively. In this example, pointer

2 cannot be detected since it is outside of the scale range. Since the orientation of the scales and the position of one pointer are known, the measurement unit determination, computing the angle between the origin of scale 1 and the pointer 1 is performed. The measuring instrument value determination has the result mv_1= -0.01 and mv_2= not detected.

In Figure 4.17b the analysis graph for the dateclock is shown. In contrast to the analysis graph of the manometer, in this example, the positions and orientations of the scales are induced by generating hypotheses. These hypotheses are verified by detecting two scales (scale 1 and 3). Possible misregistrations are corrected using the detected centers of the scales. Next, the positions of the individual pointers are computed, i.e. using pattern matching techniques. The measuring unit determination relates the angles of the pointer to the origins of the scales, computing the measuring value of the unit. Finally the measuring instrument value determination has the output mv_1= October, mv_2= Sunday, and mv_3= 13 for this example.

4.5 Chapter Summary

In this chapter the application of the proposed inspection concept was demonstrated on the case study of analogue display instruments. The function and properties of these instruments and the motivations why analogue instruments are still used and calibrated were discussed. There are two different classes of measuring instruments, single scale instruments that usually measure units like weight or pressure, and multiple scale instruments that usually measure units like time. Multiple scale instruments have coupled pointers in order to have the same measurement value in different scales on one instrument. The distinction between these two classes is necessary since the methods for performing calibration differ.

This class of objects served as a demonstration of the flexible inspection system set-up, starting with the inspection model generation. Following the definition of the primitives to be detected, the generic detection of the primitives of different ADI's demonstrated the first step in the concept. With the help of the proposed primitives it was possible to define the specific parameters of ADI primitives: scale, pointer, and lettering element. Together with the relations between the primitives the description language for ADI's was defined, forming the analysis model necessary for inspection. The description of different ADI's and their specific parameters demonstrated the specific parameters of the description.

With the help of the description language and the proposed general analysis graph the specific analysis graph for ADI's was defined. Examples were given to demonstrate that this ADI analysis graph can be used to model the analysis of specific instruments. This chapter only dealt with the analysis part of the analysis graph, the detection of the primitives was not shown in detail, since the graph provides a defined interface for the detection algorithms, which can be seen as a detail relation of primitive detection nodes.

Chapter 5

Industrial Application of Inspection Concept for Watermeters

In this chapter, an industrial application demonstrates the proposed concept, resulting in a successful working inspection system for watermeters, a specific analogue instrument.

In industrial settings analogue measuring instruments are still used and have to be calibrated by human workers after the manufacturing process. Watermeters are used to measure the water consumption of households. The inspection system to be constructed should calibrate the instrument; the system is not intended to read the individual watermeters installed in the households since it would be too complicated to transport the whole equipment to the individual houses in order to take a single image of the watermeter. In contrast to powermeters, where digitally monitorable and remotely readable instruments exist, watermeters are often installed in places without any connection to networks and without power supply. Therefore, a remote reading of this kind of instrument is not possible. There do exist digital watermeters but their costs are much higher (about five times the cost of an analogue instrument) and they are not installable within a network for a reasonable price. Furthermore, the existing analogue meters have a lifetime of about 30 years, a change to new instruments would increase installation costs and waste raw materials within the next 15 years and longer.

The task of AVI of watermeters consists of a calibration of the meter. Figure 5.1 shows a typical watermeter The inspection model described in Section 4.2 results in the object primitives scale, pointer, and lettering elements. In the specific case of watermeters there are a type-dependent number of circular scales (2 to 5) with coupled pointers, one rectangular scale (not on all types), and different types of lettering elements. Of these, two are always present and of importance: the serial number, which identifies the watermeter unambiguously, and the rotary counter, which displays the measurement value for further reading in the households [SAB95b,SAB95c].

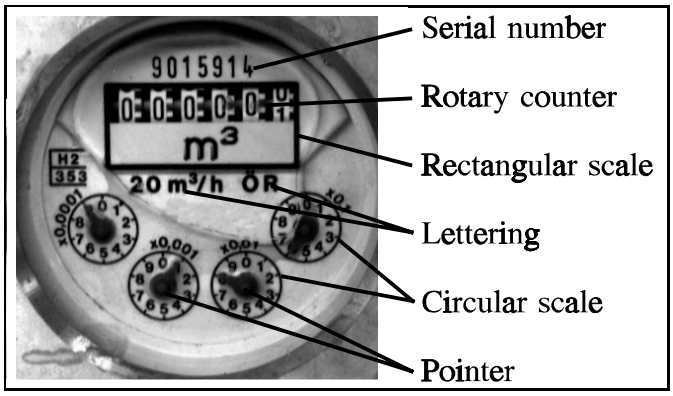


Figure 5.1 Example of watermeter and its primitives

It is important that all automation processes implemented for an industrial application fit into an existing workflow [SAB95d]. Therefore, the first step of the AVI system design is a workflow analysis, determining which machine vision problems have to be solved and what kind of constraints and preconditions are given for the automation of the process. The description of the manual inspection process and its automation can be found in Section 5.1. How-

ever, before the generation of the inspection system can take place, image acquisition providing initial image characteristics must be carried out (Section 5.2). Next, the primitives and their relations which form the description language for watermeters with the help of generic detection have to be defined (Section 5.3). With the help of the description language and the general analysis graph for ADI's the specific watermeter analysis graph is defined. Feature detection algorithms are selected and tested, making an instantiation of the analysis graph possible (Section 5.4). Following the preliminary stage, the analysis graph is refined to fulfill the constraints of the industrial environment (Section 5.5). The chapter concludes with a discussion of the results.

5.1 Calibration of Watermeters

Up to 40 watermeters of the same type are fitted into the testing assembly in order to calibrate them at the same time. Ten instruments are put in one line forming four pipes which can be flooded with water. Figure 5.2 shows a calibration table with watermeters. The distances between each two counters are not fixed. The test equipment is movable on rails and can be translated horizontally.

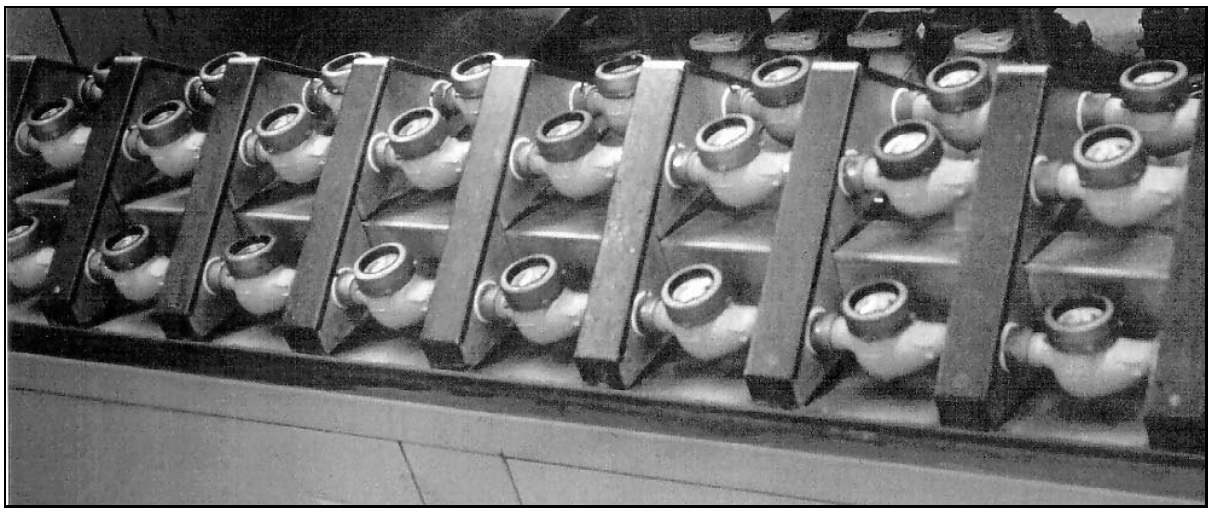


Figure 5.2 Calibration table with watermeters

The watermeter calibration process consists of the following steps [SAB95a]:

1. **Installation**: Watermeters are installed on the calibration table.
2. **Flooding**: Remaining air has to be removed from the system. It cannot, however, be guaranteed that there is no air in the system after flooding. So air bubbles can form in the displays. Flooding has to take place prior to reading since the pointers of the instruments display their correct value only under pressure inside of the instrument.
3. **Reading of instruments**: the values of the instruments are read and protocolled.
4. **Reading of reference water container**: There exists a reference container at the end of the four pipes, where the amount of water that was pumped through the pipes is collected. Before and after each test sequence the level of water in the reference water container has to be read and written down so that the actual amount of water during the test can be determined.

5. **Test sequence**: A defined quantity of water is pumped through the meters. First a smaller amount is used, then larger quantities in ascending order until 200 % of the nominal meter value are pumped through the meters.
6. **Reading of instruments**: see step 3.
7. **Reading of reference water container**: see step 4.
8. **Emptying**: After each test run the reference water container has to be emptied in order to be able to measure a reference amount of water.
9. **Reading of reference water container**: it might happen that the container is not completely empty, the amount of water that is still in the container has to be determined in order to provide a correct result for the next run.
10. **Repeat**: Steps 3 to 9 are repeated four times with different amounts of water.
11. **Removing**: The watermeters are removed from the calibration table.
12. **Protocolling:** All reading results and the container values are put into a protocol and the differences between the corresponding values are determined.

The automatic system has to be included in this work flow in order to facilitate the following steps: reading of instruments and protocolling (steps 4,6,12). Therefore, an AVI system for watermeters must be able to read the pointers.

Both processes, the former calibration and the automatic reading by computer have to be synchronized in an industrial application and an automation of the calibration process. Figure 5.3 shows both processes. The former manual process, which can be seen to the right, has to be automated at the previously defined steps; reading of instruments and protocolling. The solid arrows show the control sequence in the process, dotted arrows show the data flow between the single steps.

All the watermeters calibrated in one calibration session have to be of the same type, which allows a manual input of the watermeter type by the operator. This type-input selects the description of the watermeter within the description language. Following the preliminary steps 1 and 2 the acquisition of images has to be done. There are two alternatives:

1. *Sequential acquisition*: The images of all watermeters are taken with one CCD camera mounted into a handheld acquisition device. A person walks from watermeter to watermeter, puts the attachment over the display and takes an image. The image is automatically checked for contrast and saved. If an error occurs, a warning bell allows the acquisition to be repeated or any problem to be removed. The saving is done while the person moves to the next watermeter. Instead of moving the acquisition device manually there is the possibility to move the camera on rails with a motor fixed above the watermeters.

2. *Parallel acquisition*: For each watermeter a camera that takes the image exists. The images are taken quasi-parallel, there has to be a switching unit that allows the video capturing using only one framegrabber or real-parallel if multiple framegrabbers are used. The images are stored in the video memory and saved after capturing. Like in the sequential acquisition, images are checked for contrast and a warning bell allows an error handling.

While the test sequence runs (water is pumped through the watermeters) the image analysis is performed for every of the 40 images, computing the measurement value each watermeter displays. The automatic reading process has to be finished within 140 seconds, which is the minimum length of the first test sequence. This fact constrains the time limit for the

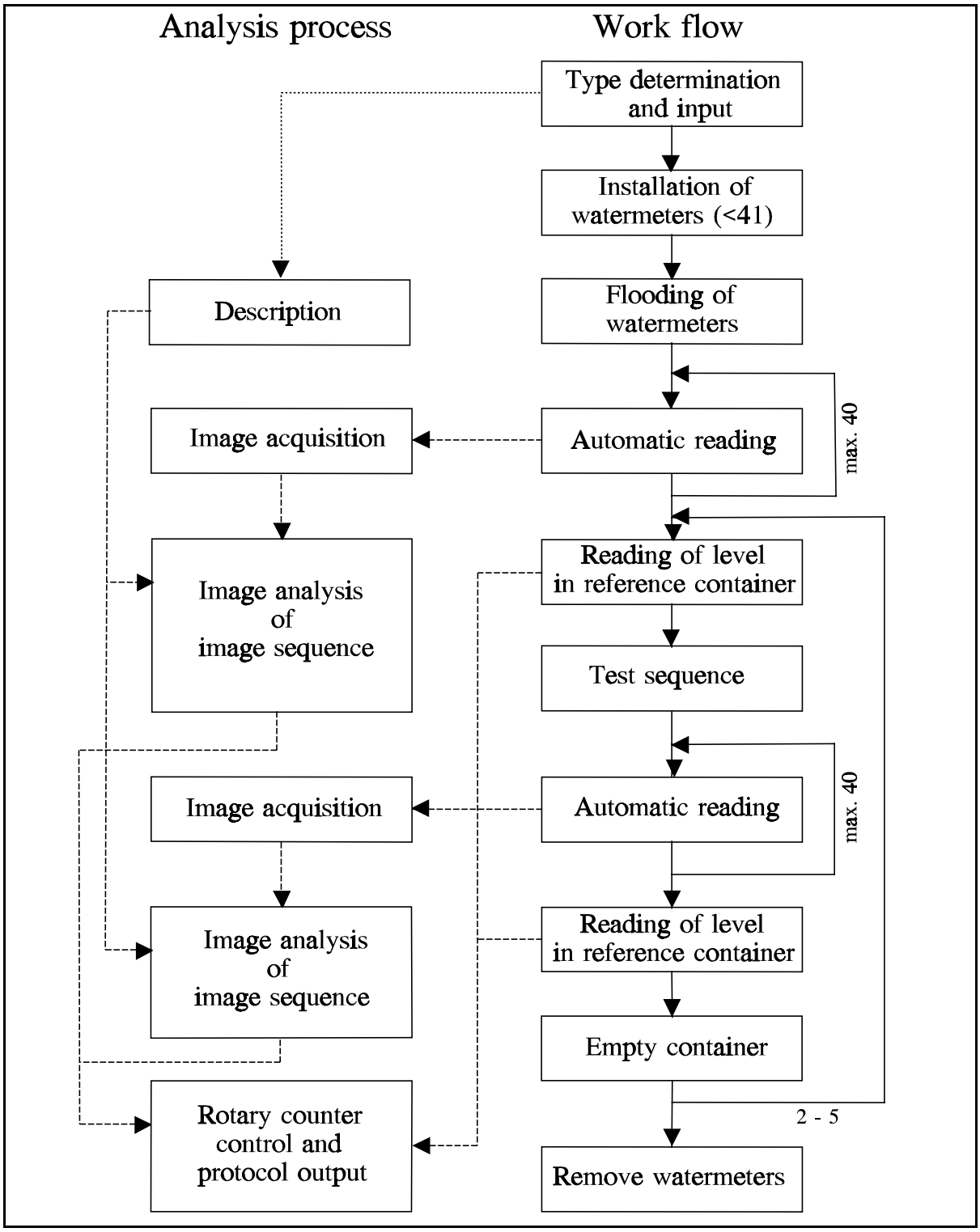


Figure 5.3 Automated calibration process

reading of one watermeter to a maximum of 3 seconds. The level of water in the calibrated reference container has to be read and entered by hand after each test sequence. The steps 3 to 9 are repeated five times to ensure proper calibration. The measurement values of each of the watermeters together with the levels of the reference water container are stored in a protocol and printed at the end of the calibration.

5.2 Image Acquisition for Watermeters

The first step in the set-up of the watermeter inspection system is a definition of the image acquisition geometry and illumination. In order to ensure an accurate analysis, the image should have high contrast and there should not be any shades or reflections in the image.

5.2.1 Image Acquisition

There were 3 constraints for the image acquisition given by the industrial partner [SAB94b]:

- low cost hardware
- pointers can move during image acquisition
- 60 cm maximum distance between camera and measuring instrument

Figure 5.4 Laboratory setup showing camera, PC, and control monitor and preliminary illumination

To optimize resolution and the constraint of low hardware costs we used a commercial quality monochrome CCD camera Sony XC 75CE with a resolution of 752 x 582 pixel in connection with a suitable low cost monochrome framegrabber board mounted into a PC486. Figure 5.4 shows the first laboratory set-up, the camera mounted on a repro tripod, a preliminary illumination using two neon lamps, the PC486, and the control monitor.

To fulfill the second constraint "moving pointers", a camera type with shutter option is used. The shutter speed can be selected from 1/125 to 1/10000 seconds. The maximum speed of the pointer is 15 revolutions per second. A shutter speed of 1/250 was selected to ensure an accuracy of ±18 degrees of the position of the pointer. At lower pointer speed the accuracy of the pointer position is higher.

The distance between watermeter and camera was set to 60 cm, due to the industrial environment. We used a 8 mm C-mount standard quality lens for image acquisition. Lenses with shorter focal length (6 mm and 4 mm) were considered, but not selected because distortions of standard quality lenses are too high. Circular scales on the measuring instrument would be-

come elliptical and circle detection would not operate without lens correction. Figure 5.5 shows the laboratory set-up including watercontainer and pump underneath the table to simulate water circulation. This water circulation set-up is necessary, since the instrument has to be filled with water to operate correctly.

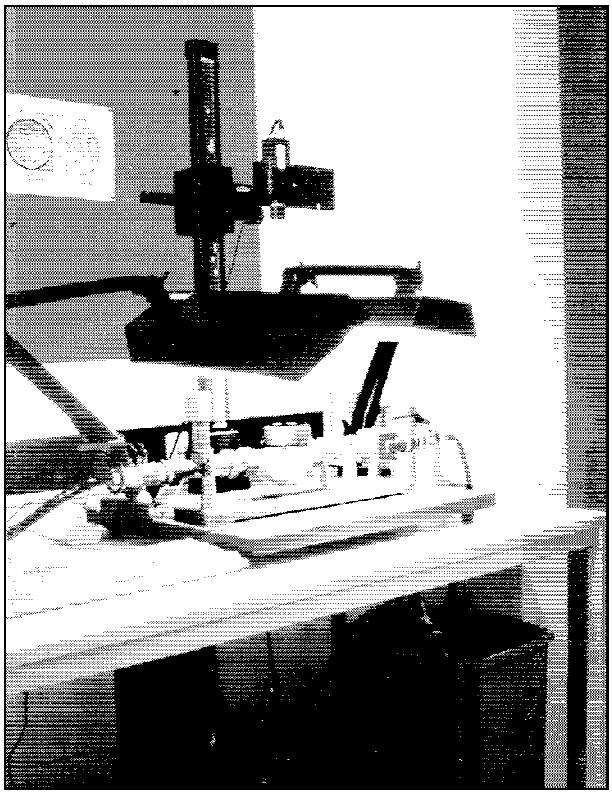


Figure 5.5 Laboratory setup with water pump

5.2.2 Illumination

The industrial environment allows only two solutions for the illumination problem. Either complex preprocessing has to be done to adjust different illumination conditions in the factory or constant illumination is provided by a special illumination device designed for the special illumination problem [AHL91]. The experience with the illumination in the first set-up (two neon lamps) led us to choose the second solution for two reasons: it helps to ensure correct reading and it can be implemented in our specific environment.

Like most of the measuring instruments, our instrument is covered with a glass plate, causing specific problems with specular light and shadows. With specular light a reading of the pointers is impossible because a highlight in the area of a pointer makes the pointer invisible. Another illumination problem that causes incorrect reading of the pointers results from shadows produced by the pointers. Pointers always have to have some distance to the background in order to be moveable (in our case approx. 3 mm). This distance produces shadows cast by the pointer and therefore ambiguities in the determination of the pointer‘s position.

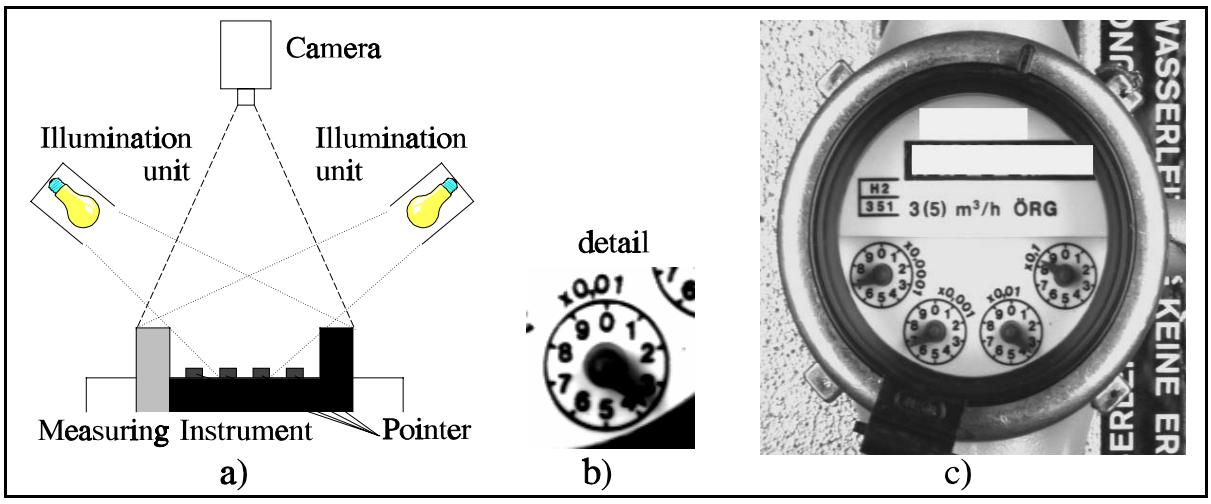


Figure 5.6 Illumination avoiding highlights

There are standard illumination techniques used in industrial applications (see [BAT85] for a survey on illumination techniques), the most common solutions among them are:

- *Directional front* illumination: This method avoids highlights, if the angle between the light reflected on the object and the optical axis of the camera is more than 30° as shown in Figure 5.6a. The result of this illumination configuration is shown in Figure 5.6c, where shadows cast by the pointers can been seen in Figure 5.6b. For this reason and because of the specific shape of our measuring instrument, the illumination configuration shown in Figure 5.6a cannot be used to illuminate the measuring instrument.

- *Profile-projection* or coaxial lighting method: The profile-projection or coaxial lighting method uses a semipermeable mirror positioned at an angle of 45° to the optical axis of the camera to diffract the light on its specular side onto the surface of the object (Figure 5.7a). The illumination direction is at an angle of 90° to the optical axis and is defracted in the direction of the optical axis. Subsequently the light is reflected from the object without producing shadows. The drawback of this configuration is that it works only if there is no glass plate on the measuring instrument. In our case the light was totally reflected by the glass plate (see Figure 5.7c and Figure 5.7b).

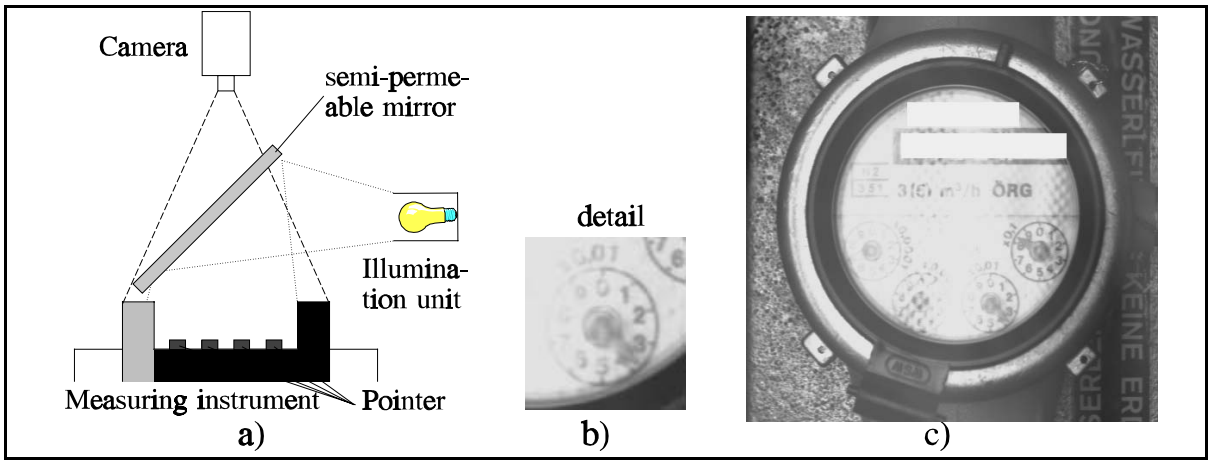


Figure 5.7 Profile projection illumination

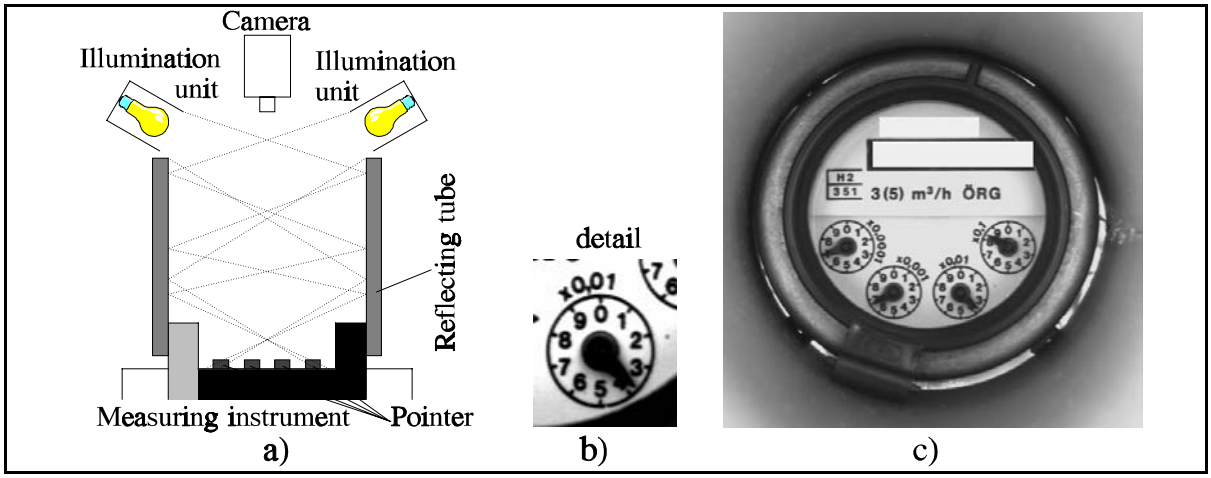


Figure 5.8 Reflecting tube illumination configuration

Therefore another illumination configuration was constructed to overcome the difficulties with specular lights and shadows. A homogeneous, diffuse illumination of the measuring instrument was necessary to obtain a basis for further computation. A tube with a diffuse, specular inner surface was affixed to the measuring instrument as shown in Figure 5.8a. This 20 cm diffuse reflecting tube was illuminated by two lamps on the top of the tube with an illumination angle of approx. 70° with respect to the optical axis of the camera. The surface of the tube defracted the incoming light diffusely onto the opposite side of the tube, there it was defracted once again and so on. The set-up of this illumination configuration was optimal, because neither highlights nor shadows disturbed image acquisition (Figure 5.8c). The pointer had no shadows and a high contrast to the background (Figure 5.8b).

The laboratory set-up with the two lamps (see Figure 5.8a) could not be used in the industrial environment since lamps produce heat causing a higher temperature inside the tube and since the humidity in the watermeter factory is high, the glass plate is steamed up. Due to safety regulations no high voltage is allowed in the vicinity of the waterpipes. Therefore, the illumination had to be solved in another way.

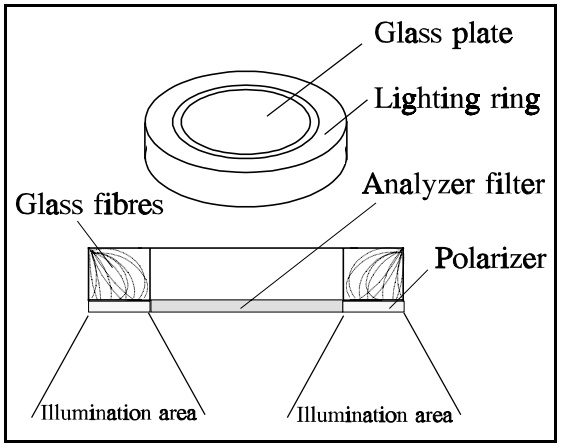


Figure 5.9 Illumination ring

A cold light source was combined with a fibre optics cable in order to have no hazardous voltage and no heat producing lamps in the illumination device. The lighting source was fixed at a distance of 16 cm from the glass plate of the watermeter inside the reflecting tube. To reduce the specular reflection component on the glass plate, a polarizer - analyzer filter system was used. The polarizer was in front of the light source to have only one lightwave direction; the analyzer (a polarizing filter) was in front of the camera lens in order to filter all other light directions than the illumination light direction (see Figure 5.9).

A casing, holding camera, illumination ring, and polarizer - analyzer combination was constructed. Figure 5.10 shows this casing and the illumination ring above a watermeter. Using the illumination ring, high quality images were also taken without the reflecting tube, although the tube improved quality. The cold light source can be seen in the lower left corner in Figure 5.10, a fibre optics cable connects the light source with the illumina-

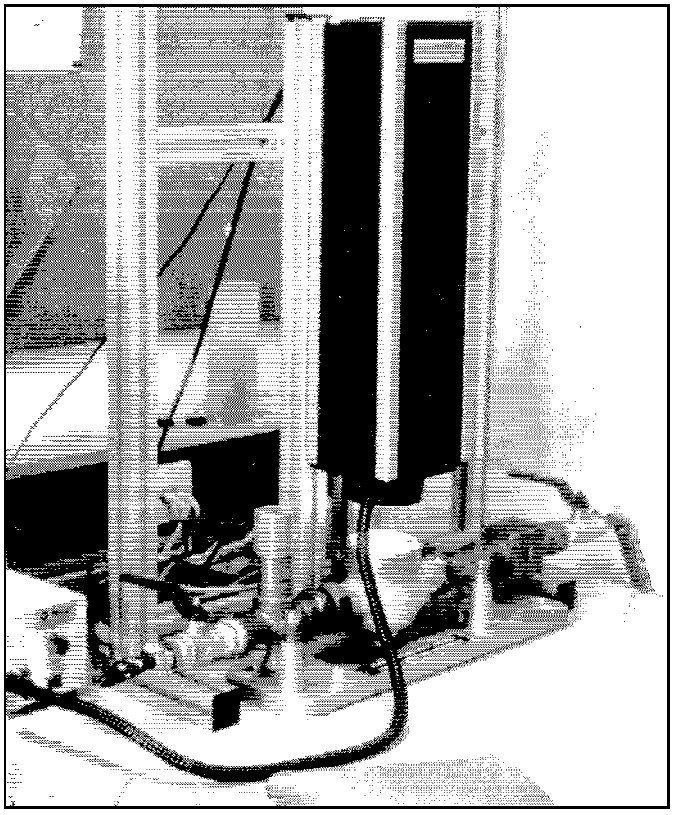


Figure 5.10 Image acquisition device using cold light

tion ring. Due to the construction of an image acquisition device, a fixed distance between camera and measuring instrument was ensured, and therefore, a nearly fixed position of the measuring instrument within the image.

To meet the constraint that the operator has a handheld device that he can move from one watermeter to the other, this device was also constructed using the reflecting tube, as shown in Figure 5.11. The fixed imaging geometry required accurate positioning of the camera or the measuring instrument. This constraint was adhered to by designing a handheld attachment such that it could only be attached in a certain, mechanically fixed way and ensured high contrast, stable illuminated images.

The non-handheld object image acquisition set-up is used to gain the images series, which is the basis for the set-up of the inspection system, since the quality of the images in this set-up is not as high as in the case of the handheld device. Furthermore, the positioning of the object is not as accurate as in the case of the handheld device, since there is no coupling of acquisition device and watermeter. However, the stand-device can be used together with the calibration table and a moving unit that moves the acquisition device to the individual watermeters, making a complete automation of the calibration possible, since no operator is necessary to move the acquisition device.

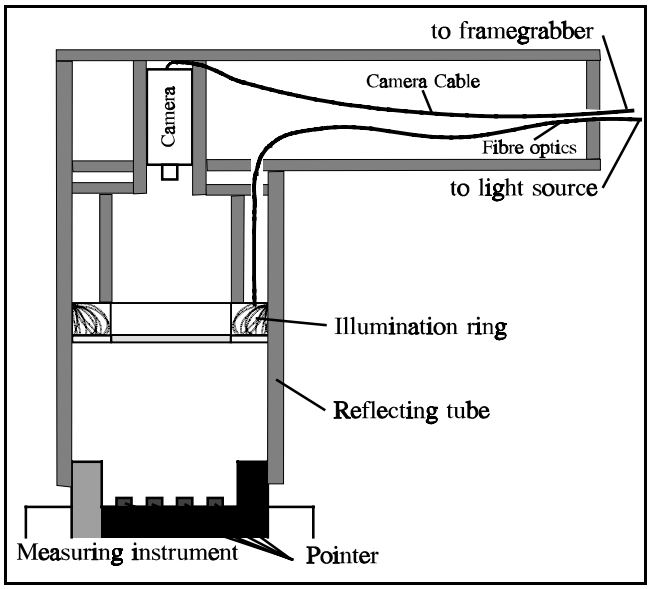


Figure 5.11 Attachment device including illumination and camera

5.3 Inspection Model Generation for Watermeters

The inspection model for ADI's (Chapter 4) defines the inspection model for watermeters. Since primitives have already been defined for multiple scale instruments (where watermeters are a sub-class of), it is only necessary to relate the actual primitives of the different types of watermeters to the primitives defined in the model. Circular and rectangular scales have already been defined in the introduction of this chapter, different types of lettering elements allow a distinction of different types, defined in this section. Since lettering elements are not detected by generic detection, they have to be defined manually. If the type and

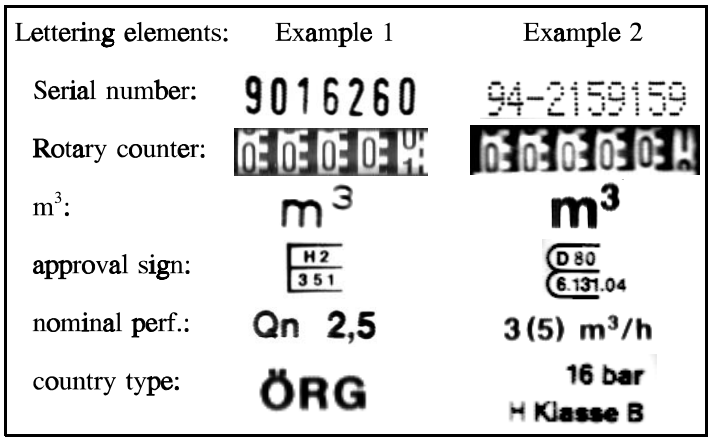


Figure 5.12 Examples for watermeter lettering elements

number of relevant lettering elements is known a priori, like in the watermeter case, they may be defined before the model generation starts.

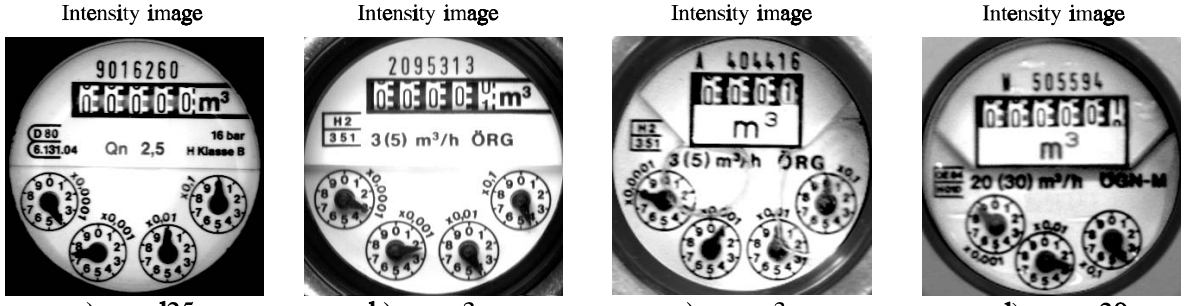
Each watermeter has a certain number of *circular scales* (2 to 5) with corresponding *pointers*, possibly one *rectangular scale*, holding the *rotary counter* and the measurement unit m^3, both defined as lettering elements. Furthermore, there are 4 other lettering elements, allowing a type distinction (see Figure 5.12):

- a *serial number*, which identifies a single watermeter unambiguously;
- an *approval sign*, which shows the government approval for a specific type;
- a *nominal performance*, which indicates the nominal amount of water that may pass the instrument; and
- a *country type sign*, which indicates the country specific type of instrument.

5.3.1 Test Images

In the following, the inspection model generation is shown on the example of four different watermeter types:

- *wz_d25*: German standard watermeter for households (Figure 5.13a), imagesize: 300 x 300 pixels. This type is used in the final inspection system.
- *wz_a3*: Austrian standard watermeter for household (Figure 5.13b), imagesize: 300 x 300 pixels. This type is used in the final inspection system, too.
- *wz_w3*: Former Austrian standard watermeter for household (Figure 5.13c), imagesize: 300 x 300 pixels.
- *wz_w20*: Example for a watermeter for industry and hospitals (Figure 5.13d), image size: 300 x 300 pixels.



a) wz_d25 b) wz_a3 c) wz_w3 d) wz_w20

Figure 5.13 Intensity images of the 4 different watermeter types

The images wz_d25 and wz_a3 were taken out of an image series as representative images for the model generation; the other two images were single test images of types that are no longer used. These four intensity images served as the basis for the model generation by generic detection.

5.3.2 Generic Detection of Watermeter Primitives

In the case of watermeters, the geometric primitives are circles, lines of a certain length, and rectangles. The differentiation between lines and rectangles is necessary, since two of the used types have an "open" rectangle as scale, the right side of the scale is not present. As discussed in Section 4.2.1, a reduced edge dataset is used for the generic detection of the primitives. Figure 5.13 shows the intensity image for each of the different watermeter types.

The result of the generic detection are shown in Table 5.1 and Figure 5.14, respectively. The detected primitives have to be checked interactively whether they represent primitives of the watermeter.

Type	Primitives looked for	Lines or rectangles found	Circles found
wz_d25	Line, circle	Upper line (76,55)-(258,56) Left line (76,55)-(77,80) Lower line (76,94)-(280,96)	Left: (50,193), 33 pixel Left middle: (106,251), 33 pixel Right middle: (187,253), 33 pixel Right: (247,191), 33 pixel Instrument: (147,151), 149 pixel
wz_a3	Line, circle	Left line (79,47)-(79,83) Upper line (23,125)-(64,125), (21,87)-(65,88)	Left: (46,188), 34 pixel Left-middle: (105,246), 34 pixel Right-middle: (190,246), 34 pixel Right: (250,188), 34 pixel Instrument: (147,147), 153 pixel
wz_w3	Rectangle, circle	Rectangle: (92,129)-(215,129)- (215,48)-(92,48)	Left: (62,190), 35 pixel Left-middle: (114,245), 35 pixel Right-middle: (190,246), 35 pixel Right: (244,194), 35 pixel Instrument: (151,149), 144 pixel
wz_w20	Rectangle, circle	Rectangle: (62,77)-(240,77) (240,150)-(62,150)	Left: (94,227), 33 pixel Left-middle: (152,270), 21 pixel Right: (232,245), 21 pixel Instrument: (148, 148), 148 pixel

Table 5.1: In- and output of generic detection

5.3.3 Description Language for Watermeters

The description language for watermeters is a sub-set of the description language for ADI's (see Section 4.3). First, the type specific description is generated using the proposed hierarchical graph for ADI's. All of the watermeters (except wz_w20, a demonstration that there are also other types), have five measuring units, four non-overlapping circular scales with corresponding pointers, and one rectangular scale without pointer. Furthermore, all of them have six lettering elements that identify the type. Note that one lettering element, the rotary counter, could also be seen as a "pointer" since it displays the measurement value within the scale. But as the value is not displayed using the common definition of a pointer, we define this element as a lettering element rather than as a pointer of a scale.

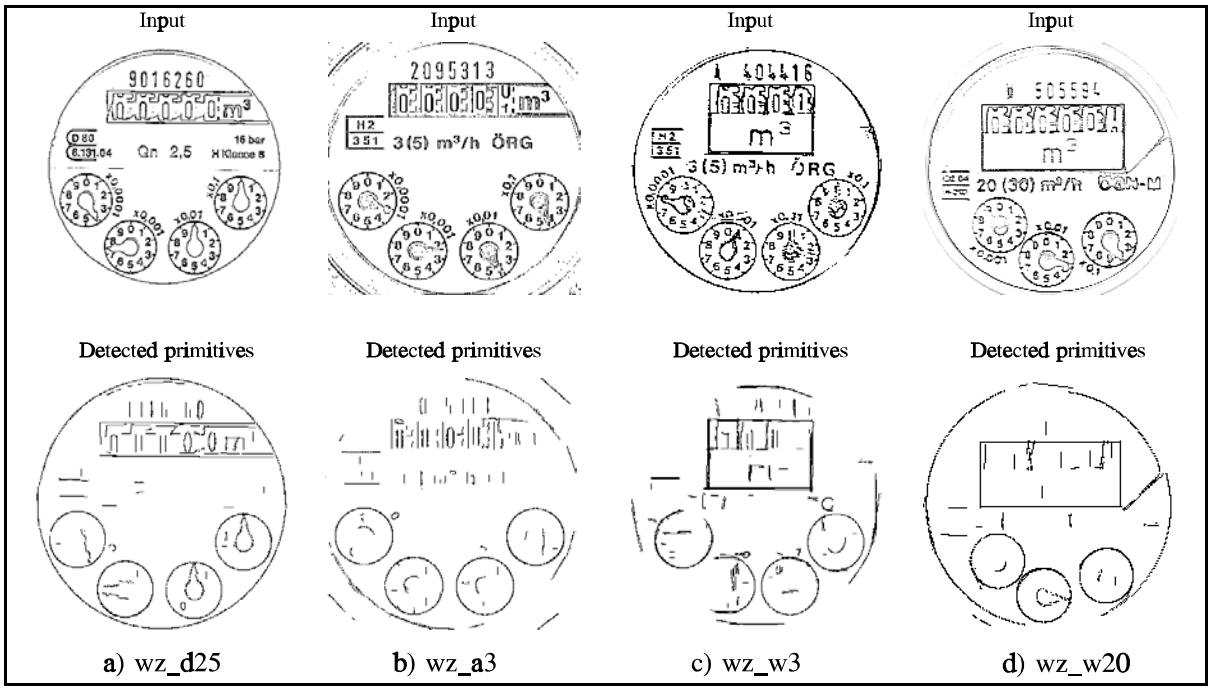


Figure 5.14 Primitives detected by generic detection within the watermeter images

In Figure 5.15a the description of the common type of watermeter wz_a3 having 4 circular scales is shown, Figure 5.15b shows the description of wz_w20 as an example for a different type of description.

In an interactive step, the geometric primitives given by the generic detection are selected, their specific parameters are taken to form the description. Since all watermeters are of circular shape, the center of the object centered coordinate system is the center of the instrument. The relations between the primitives are taken from description language for ADI's. The generic parameters of the lettering elements have to be defined interactively, determining the bounding rectangle of each lettering element. All of the parameters for all examples are inserted into the individual descriptions. Table 5.2 defines the root of the graph and Table 5.3 shows one example for the measuring units for the watermeter wz_a3 (all others are defined in the same way as in this example except wz_w20 which has 4 units).

Measuring instrument:

type	shape	origin (x,y)	size (pixel,mm)	*n*	*m*	*v*
wz_d25	circle	0,0	290,2000	5	6	coupled
wz_a3	circle	0,0	290,2000	5	6	coupled
wz_w3	circle	0,0	290,2000	5	6	coupled
wz_w20	circle	0,0	290,2000	4	6	coupled

Table 5.2: Generic parameters of root nodes

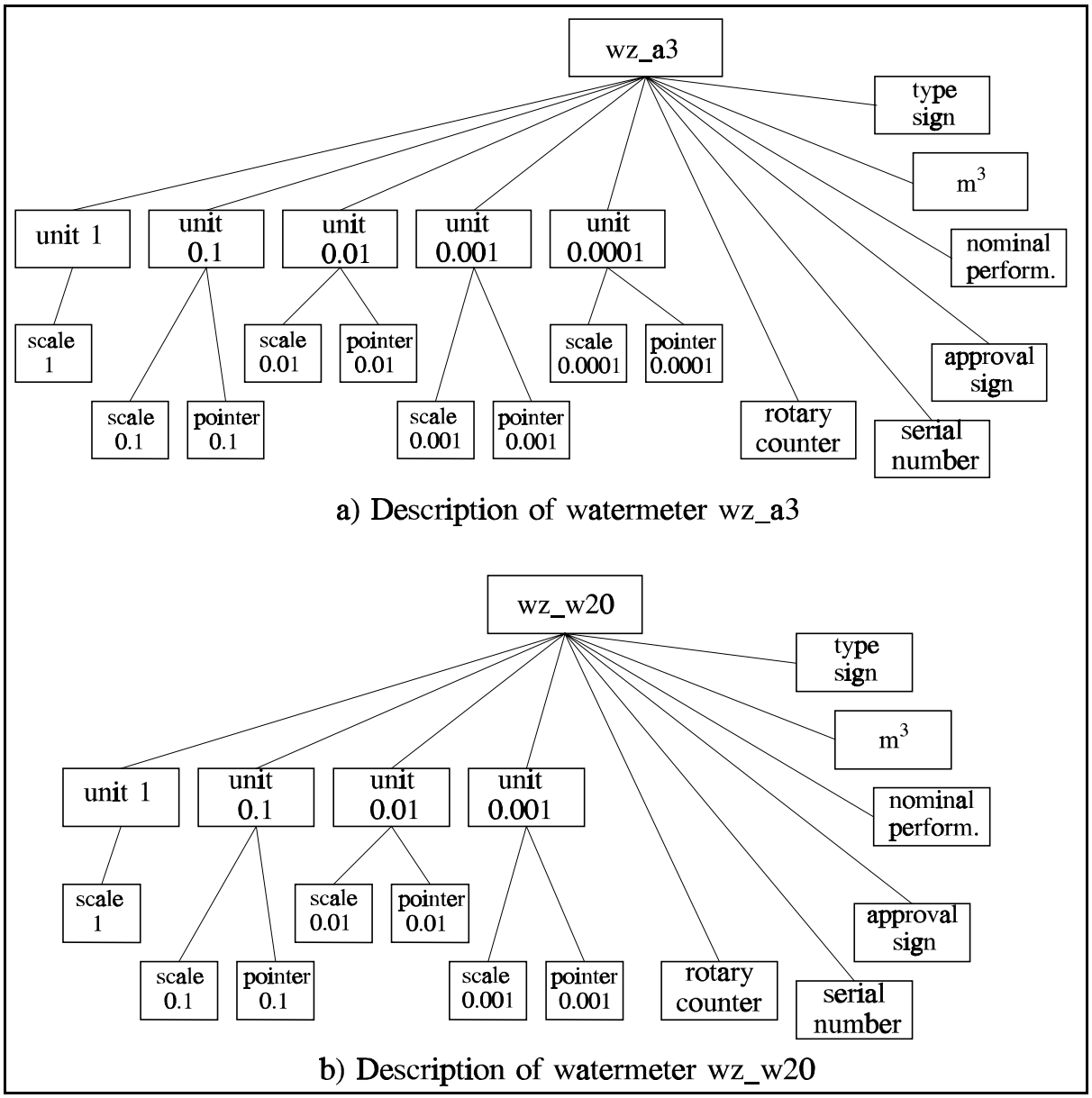


Figure 5.15 Description language for watermeters: a) wz_a3, b) wz_w20 (old instrument)

Measuring unit:

unit	*c*	*e*	*u*	*o* (°)
1	1/10.000	1	m³	0
2	1/10	0.1	m³	0
3	1/10	0.01	m³	0
4	1/10	0.001	m³	0
5	1/10	0.0001	m³	0

Table 5.3: Generic parameters of measuring units for wz_a3

Scales:

Table 5.4 shows the generic parameters of the scales of all watermeter examples.

instrument	unit	type	size (h/r)	origin (x,y)	α_{s0}	d_{sg}/α_{sg}	w/α_{sr}
wz_d25	1	rectangular	41	-71,55	0	0	207
	2	circular	35	100,-39	0	34	360
	3	circular	35	41,-100	0	34	360
	4	circular	35	-42,-100	0	34	360
	5	circular	35	-103,-39	0	34	360
wz_a3	1	rectangular	42	-75,58	0	0	215
	2	circular	35	100,-38	0	34	360
	3	circular	35	40,-99	0	34	360
	4	circular	35	-42,-99	0	34	360
	5	circular	35	-104,-38	0	34	360
wz_w3	1	rectangular	88	-62,21	0	0	130
	2	circular	35	96,-40	0	34	360
	3	circular	35	40,-99	0	34	360
	4	circular	35	-39,-99	0	34	360
	5	circular	35	-96,-40	0	34	360
wz_w20	1	rectangular	83	-85,0	0	0	178
	2	circular	35	72,-81	0	34	360
	3	circular	35	-5,-106	0	34	360
	4	circular	35	-62,-64	0	34	360

Table 5.4: Generic parameters of scales of watermeters

Pointer:

For every circular scale a pointer has to be defined, too. In the case of watermeters, all instruments have the same type of pointer, shape and size are always identical. Therefore, Table 5.5 shows only one of the defined pointers.

unit	type	shape	origin (x,y)	α_p
2	circular	bitmap (30,23)	72,-81	-
3	circular	bitmap (30,23)	-5,-106	-
4	circular	bitmap (30,23)	-62,-64	-

Table 5.5: Generic parameters of pointers wz_w20

Lettering:

Each of the six lettering elements has to be defined for each type. Table 5.6 shows the generic parameters for wz_a3, all lettering elements of the remaining 3 instruments are defined in the same way. Rotary counter and serial number (element 1 and 2) are defined as characters, since numbers differ, all other lettering elements are defined as bitmaps.

lettering	type	origin (x,y)	size	content
1	bitmap, 5	-69,65	140,33	matrix (0-9)
2	bitmap, 7	-53,106	100,24	"0" - "9"
3	bitmap	-130,23	46,42	matrix
4	bitmap	-72,23	100,20	matrix
5	bitmap	76,70	37,25	matrix
6	bitmap	44,23	47,20	matrix

Table 5.6: Generic parameters of lettering elements of wz_a3

5.4 Analysis and Detection

In the specific case of watermeters the general analysis graph for ADI's (see Section 4.4) is adapted in order to make an inspection possible. Furthermore, detection algorithms are tested and selected to perform the application independent detection, represented as detail relation in the analysis graph. This section describes the generation of the analysis graph for watermeters and the algorithms to detect the primitives. The analysis graph together with the detection algorithms, represented as detail relations in the analysis graph form the preliminary inspection process.

5.4.1 Analysis Graph for Watermeters

In the general case, the orientation and position of the instrument is not known exactly, since positioning errors in the image acquisition may occur, even if the distance to the object is fixed. Therefore, the general analysis graph for watermeters allows a handling of the misorientation and does not only generate hypotheses that are verified afterwards. The analysis graph is type independent, however, there is an interaction with the description of the type. Therefore, it can distinguish for instance, whether three or four circular scales have to be detected.

Figure 5.16 shows the general analysis graph for watermeters. Following the image acquisition the position of the watermeter has to be detected within the image. All of the scales (rectangular and circular) have to be detected within the image, since the orientation of the watermeter is not known. The result of the unit detection is checked with the description, performing a hypotheses verification. In case many different types have to be inspected, all three lettering elements (approval sign, nominal performance and country type sign) which allow

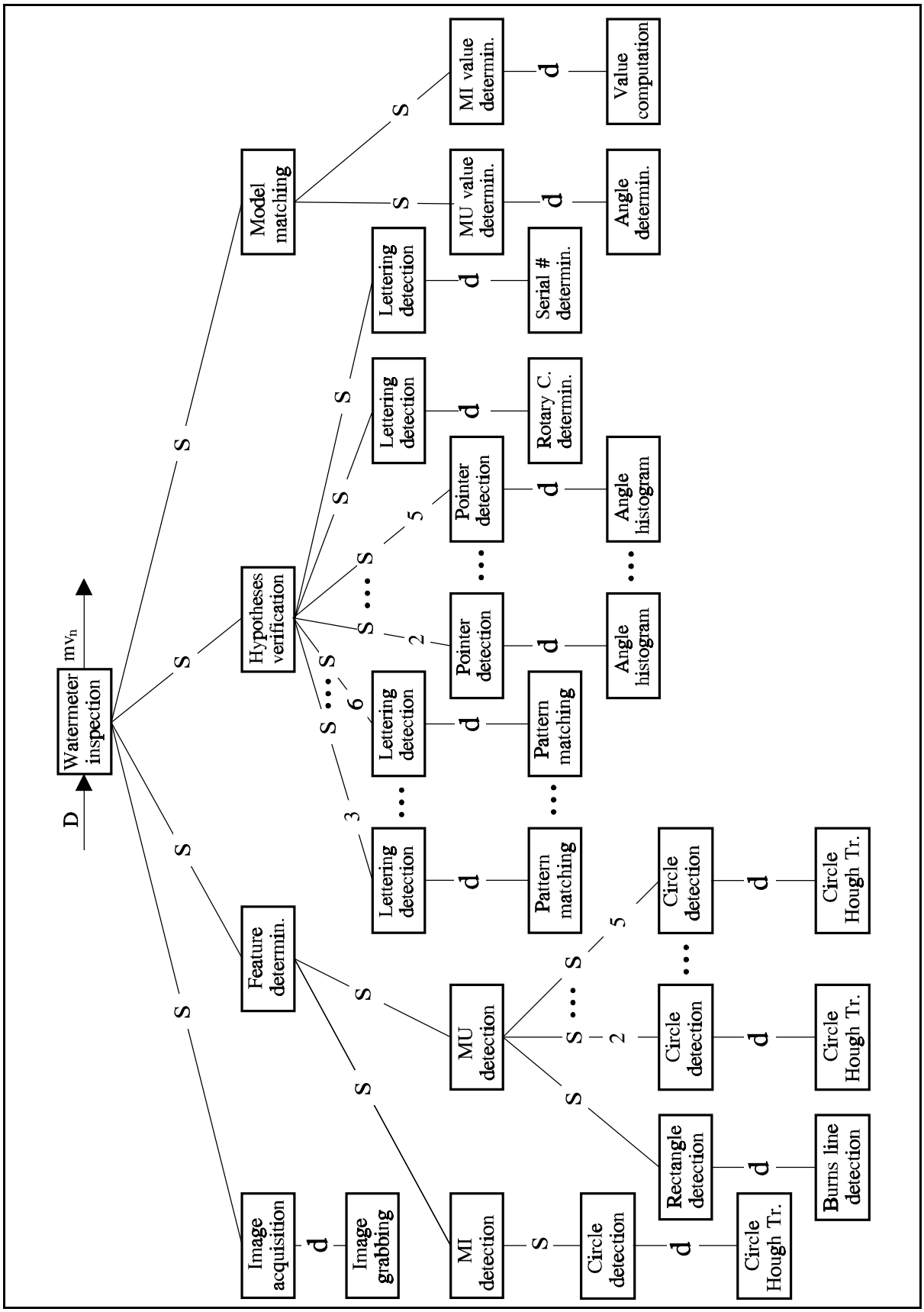


Figure 5.16 General analysis graph for watermeters

a distinction, have to be checked. Furthermore, all positions of the pointers have to be detected within the scales, once the hypotheses have been verified. To achieve the complete measurement value, the rotary counter has to be analyzed, and the serial number has to be read to identify the watermeter under inspection uniquely. At this stage, all the generic parameters of the watermeter primitives are known. The model matching consists of a determination of the value that the single units display and a value computation regarding the coupled pointer computation method.

For watermeters all positional tolerances are defined with ± 2 pixel, which was a result of generic detection. The tolerance in the measurement value is fixed at ± 0.0001, the last digit may differ from the actual value. The tolerance for the pointer detection is defined with 9°, the orientation tolerance is 2°, given by the positional tolerances. The initial value for the weights is thresholded by 0.9, i.e.: all of the primitives have to be detected correctly to be considered for further analysis.

5.4.2 Detection of Watermeter Primitives

Up to now, the analysis graph did not include specific detection algorithms, therefore a verification could not take place. This section shows how the detail relations in the analysis graph are used to detect the features. Having this kind of hierarchy and relation, a fixed interface is guaranteed, each detail relation may be replaced by another detail relation. Different methods can be used to produce one result, selected by instantiation.

In the case of watermeter inspection, the kind of features to be looked for are circles, rectangles, pointers, and lettering elements, represented as circle detection, rectangle detection, pattern matching, angular histogram, rotary counter determination, and serial number determination, in the analysis graph in Figure 5.16. Possible algorithms that solve the specific detection problems are presented in this section without limitation of generality; other algorithms can be used as well.

Circle detection

To detect circles and circular arcs in the intensity image, we use the Hough transformation [KIE92,YUE90]. The Hough method [HOU62] was extended to circle detection by Duda and Hart [DUD75] and extended by Ballard and Brown using the gradients [BAL81]. The detection process has the following steps:

1. *Edge detection*: The derivatives in the x and y direction are computed and form the gradient; different gradient operators may be used;
2. *Hough transform*: Circle centers are clusters in the accumulator space;
3. *Peak enhancement*: The accumulator image is convolved for instance with a 9x9 Laplacian-like peak filter as derived in [DAN90];
4. *Max finding*: A procedure finds the center of the peak in the accumulator with a neighborhood operation.

The result of circle detection is a set of potential candidates for arcs and circles. The circular Hough transformation estimates the position and the radius of the circular arc or circle.

To verify if the candidate is a circular arc or circle, the edge points of the supposed circle are counted along the circle as shown in Figure 5.17. If the density of detected edge points on a sector of a certain angle is more than 70% of the circle points in the sector, this sector is accepted as a circular arc; if 70% of the circle sectors are accepted, the supposed center is a center of a circle. Figure 5.18 shows the result of detecting circles in intensity images. These circles are the scales of a measuring instrument with 4 circles and all of them were detected.

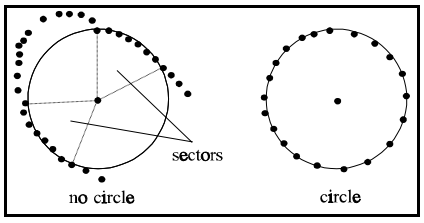


Figure 5.17 Arc verification

Circle detection results (test series I):

- *Number of images*: 50 images of several watermeters of the type wz_a3 under different lighting conditions
- *Image size*: 300x300 pixels
- *Detection rate*: 98%
- *Positional accuracy* of circle: ±2 pixel.

Circle detection results (test series II):

- *Number of images*: 50 images of several watermeters of type wz_a3 under different lighting conditions
- *Image size*: 400x400 pixels
- *Detection rate*: 100%
- *Positional accuracy* of circle: ±1 pixel.

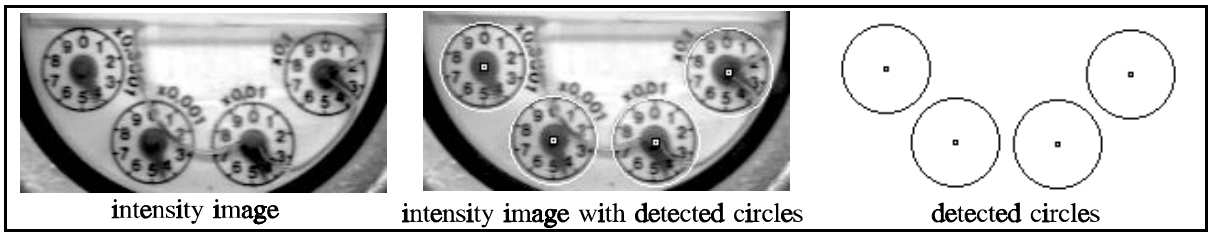


Figure 5.18 Detected circles of a watermeter

Rectangle detection

A rectangle consists of 4 straight lines, which we detect with the approach of Burns et. al. [BUR86]. This algorithm groups pixels into line support regions of similar gradient orientation. The structure of the associated intensity surface determines the location and properties of the edge. The line extraction is more effective than previous techniques (for example Hough transform [HOU62]) because the gradient orientation rather than the gradient magnitude is used as initial organizing criterion and because the global context of the intensity variations associated with a straight line is determined prior to any local decision about participating edge elements. Four steps can be distinguished:

1. *Group* pixels in line support regions;

2. *Approximate* the intensity surface on both sides of the line by a planar surface;
3. *Extract* attributes length, location, orientation;
4. *Filter* lines with certain length, location and orientation.

Steps 3 and 4 have changed from the original algorithm because our aim is to detect lines belonging to rectangles in the intensity image. Only three features given by the Burns algorithm are used, leaving long straight lines in the line set. An orientation histogram is computed and lines with 90° orientation difference are combined to form rectangles (Figure 5.19).

Rectangle detection results:

- *Number of images* in test series: 20 images of several watermeters of type wz_a3 under different lighting conditions
- *Image size*: 400x400 pixels
- *Detection rate*: 100%
- *Positional accuracy* of rectangle: ±1 pixel.

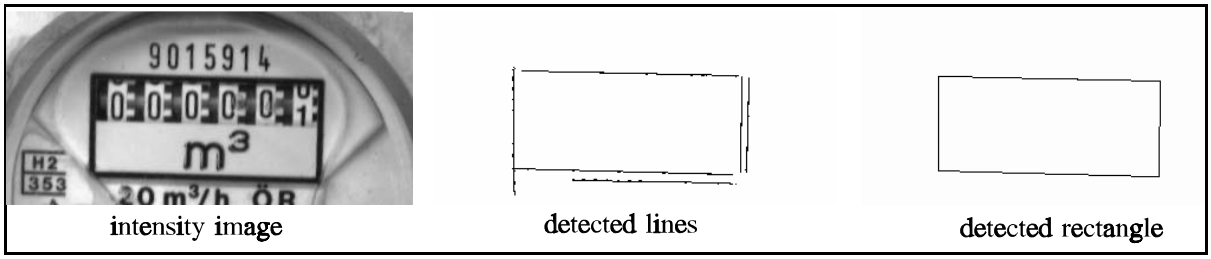


Figure 5.19 Detected rectangle

Pattern Matching

The detection of the lettering elements m³, approval sign, country type sign, and nominal performance can be done by pattern matching techniques. The lettering elements check and verify the type and orientation of the measuring instrument. They are postulated in a certain position and orientation in the image and checked by the correlation coefficient between the window found in the intensity image and the bitmap of the lettering element. To increase speed and robustness in the first correlation step, row and column histograms are correlated as shown in Figure 5.20. Every row and column histogram is correlated with the histograms defined in the description. If the histograms match, no further computation is necessary. If there is any mismatch the complete windows are correlated by a template matching algorithm [ASC92] in order to decide whether there is the correct lettering element on the postulated position. The correlation coefficient defines the similarity between the image window and the reference bitmap and gives the probability for a verification in the analysis graph.

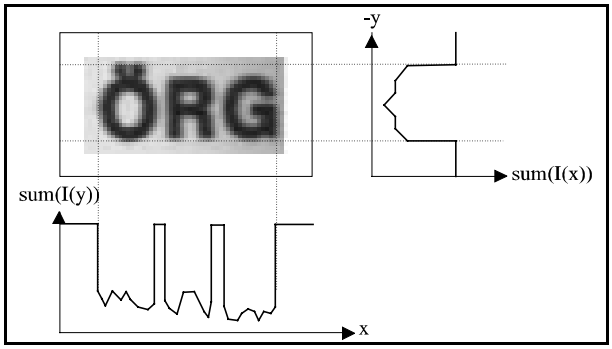


Figure 5.20 Row- and column histogram for lettering element

Angular histogram

Two different methods for detecting the pointer in the scale area were tested:

- *Valence independent computation* of gray level profiles along curves in the image plane: The intensity values are accumulated along an axis from the center of the pointer. The mean intensity value of a 5x5 window at the center of the pointer is computed and subsequently this mean intensity value is used to count pixels with similar intensities in the angle histogram (Figure 5.21).

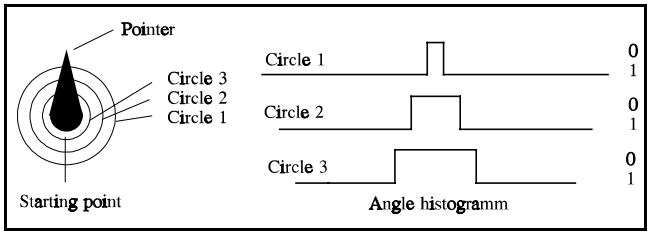


Figure 5.21 Angle histogram of a pointer

- *Multiresolution computation* by coupled pointers (Section 4.1.3): This method is based on dividing the scale area into sectors and computing the weighted deviation from mean. This deviation gives a probability for the position of the pointer in a sector. The reading of the values indicated by the pointer position starts with the lowest valence. This strategy is characterized by low error propagation, since the previous pointer defines the position of the following pointer as shown in Figure 5.22.

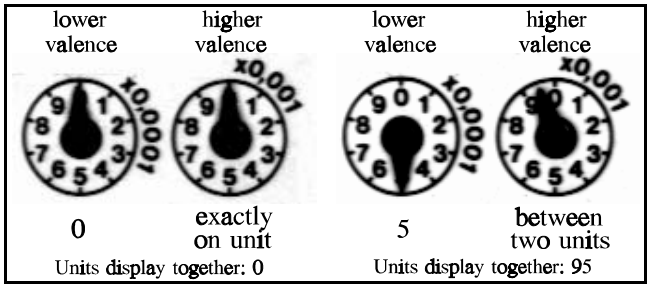


Figure 5.22 Coupled pointers

Let $\overline{e_i}$ denote the measurement value and e_i the correct value. If the reading error for e_{i+1} is $|\overline{e_{i+1}} - e_{i+1}| < \delta$, the reading error for the next valence is:

$$|\overline{e_i} - e_i| = |\frac{\overline{e_{i+1}}}{u_{i+1}} - \frac{e_{i+1}}{u_{i+1}}| < \frac{\delta}{u_{i+1}} \quad . \tag{5.1}$$

For the unit of the next valence $u_{i+1}>1$ holds (e.g. 10 in Figure 5.22), therefore, the error of the lower valence has a small propagation and little influence on the value of the measurement.

Figure 5.23 shows the result of the pointer detection using multiresolution computation. The upper half shows the intensity images of the pointers and the lower half the results of the analysis algorithm. Note that the computed value is always the previous value on the scale if the pointer has not yet passed the scale value. For example, one might

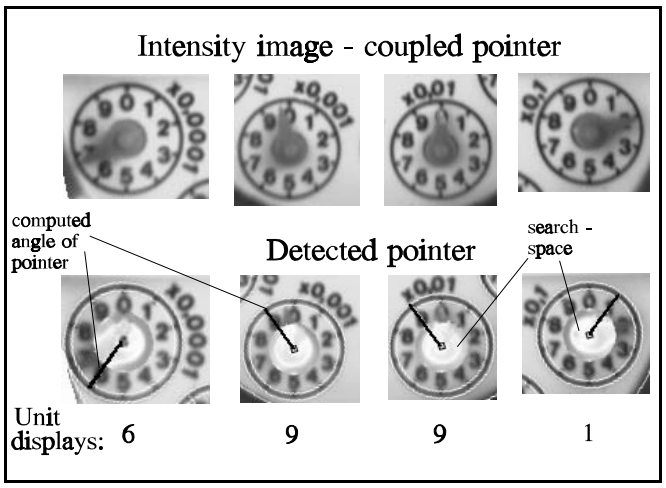


Figure 5.23 Detected coupled pointers (0.1996)

think the computation has led to incorrect results, because the rightmost pointer displays the value 2 and the computed value is 1. In reality the pointer has not passed the value 2, since the lower valence pointer displays 9.

Rotary counter determination

To read the complete measurement value the watermeter displays, it is also necessary to read the rotary counter. The specific case of watermeter calibration after manufacturing it is guaranteed that the rotary counter has the initial value 0, and that only the rightmost digit is rotating. Therefore, only this digit is read by using the pattern matching correlation technique. Since the rotary counter displays values between two numbers (for instance if the instrument displays 0.5 m³, the rotary counter displays approximately one half of the number 0 and one half of the number 1), the correlation is not only performed correlating ten numbers with the actual value but by using a windowing technique.

In the description of the watermeter one complete stripe of one digit roll is represented in a 2-d matrix including all successive numbers from 0 to 9. To determine which number the rotary counter digit displays, a window containing the digit of the actual image is moved in the image of the complete roll. The correlation of the window, containing the actual image and the window on the position in the reference image is computed. For each position the similarity measure is stored, defining a correlation function of the window position. The position indicating the highest correlation is supposed to be the number the instrument displays (see Figure 5.24). The window containing the actual value of the digit is moved from left to right in the description image, the maximum of the correlation function is supposed to indicate the value the counter displays.

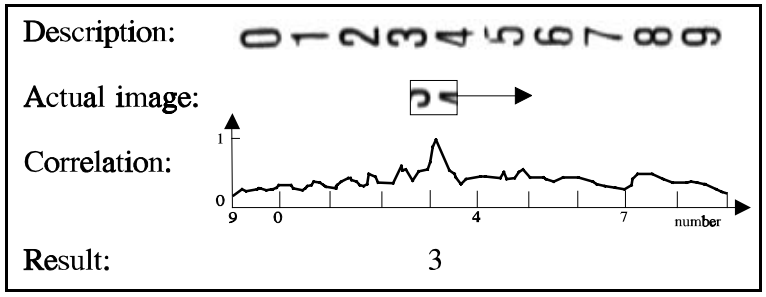


Figure 5.24 Rotary counter determination by window correlation

Rotary counter check

The rotary counter determination is different from another check that is performed at the end of the calibration run if the amount of water led through the calibration table was less than 1 m³. In this case it is necessary to check the counter separately to ensure that the rotary counter is mechanically coupled with the pointers, since the computed value for each calibration run would always result in the same number (0). The rotary counter control takes place at the end of the 5th calibration run. The information whether the counter has moved can be used as a criterion for the rotary counter test. The windows containing the rightmost digit of the rotary counter in the first and the last image of the calibration sequence shown in Figure 5.25 are correlated in the same way as it is done for the lettering element check. A low correlation value indicates a movement of the counter, a high correlation, a mechanical fault of the watermeter.

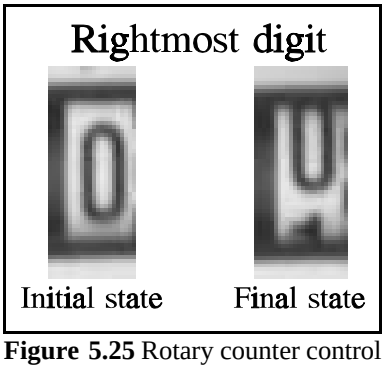


Figure 5.25 Rotary counter control

Serial number determination

The serial number can be determined either by pattern matching techniques used for lettering element detection, or by any other OCR-technique. In our first set-up the above correlation technique is used since the serial numbers are sorted in ascending order. Therefore, the last three numbers are checked by knowing the initial number a priori. A hypothesis of the number that should be on the meter is checked if it is present in the image, making this technique useable only in the specific application of calibrating instruments sorted by serial number.

5.4.3 Preliminary Inspection Process

The preliminary inspection process is constructed by combining the analysis graph and the detection algorithms forming the complete analysis graph. If different detection algorithms are available, the preliminary analysis process contains all optional possibilities to solve the detection. The graph is instantiated performing the preliminary inspection system test. If the detection algorithms were already tested separately, like in the case of watermeters, this instantiation would not be necessary. For the example of lettering element detection, an optional detail relation could consist of an algorithm using an OCR-technique. The test stage would then define which of the algorithms performed better with the given test data.

The analysis graph, containing the image acquisition described in Section 5.2, the watermeter detection and scale detection, the lettering and pointer detection and the measurement value determination are tested together with the description language whether they solve the inspection problem during the inspection system generation. If some primitives cannot be detected or the result is not correct, the detection algorithms have to be changed.

5.5 Inspection System Generation for Watermeters

The preliminary inspection system has to be tested to determine whether it is able to perform the inspection. The inspection system generation ends up with the final inspection system, therefore, the testing of the analysis graph with image test series is the major content of this stage. Furthermore, industrial constraints have to be taken into account. After the first test, parameters like speed, accuracy and reliability are available. These parameters have to be checked with the constraints, if they are not met, strategies to satisfy the constraints have to be developed. This section shows on the example of the inspection system generation for watermeters, what such a test and adaption looks like and how the analysis graph is instantiated in order to produce an inspection system that meets the constraints.

5.5.1 Preliminary Inspection System Test

The preliminary inspection system was first implemented in an experimental developing environment on a Unix platform, using a SUN Sparc 10 workstation. A series of 50 test images of type wz_a3 taken under fixed conditions serves as a test dataset. Note that there exists only a test series "gold", since there are no defect images if the instruments are only read. This

constraint can be guaranteed, since the instruments have to be adjusted manually and within this working step they are inspected on their visual appearance. An instrument that does not work could be seen as defect instrument, however, its images do not differ from instruments that work.

Results of preliminary system test:

- *Number of images* in test series: 50 images of several watermeters of type wz_a3
- *Image size*: 740x575 pixels
- *Image size of instrument*: 450x450 pixels
- *Detection rate*: 100%
- *Positional accuracy* of center of measuring instrument (i.e. the center of the image centered coordinate system): ±2 pixel in x- and y- direction
- *Positional accuracy* of center of circular scales: ±1 pixel
- *Positional accuracy* of rectangle: ±2 pixel
- *Positional accuracy* of pointer: ±3°
- *Average computation time/image*: 26 sec. on SUN Sparc 10 workstation
- *Overall accuracy*: 100%

The detection of scales was performed within the detected instrument, without limitation of orientation. Since the quality of circle detection was not known a priori, the radius of the circles was constrained to ±8 pixel. Therefore both circles, the outer and the inner border of the scales were detected. The rectangular scale detection worked, the open rectangle was defined by three lines, all of them detected correctly. Two lettering elements were checked, the country type sign and the nominal performance sign. They were accepted if the correlation coefficient was above a 0.8 threshold. Serial number and rotary counter were not determined in the preliminary system test by setting the weights in the description to 0.

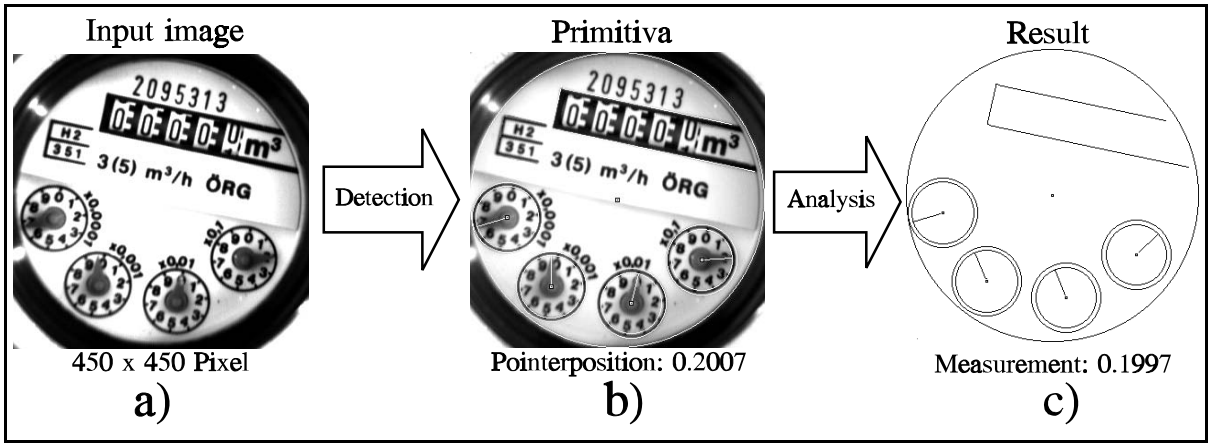


Figure 5.26 Detection indicates the pointer positions 0.6819, relations between pointers give correct result

Figure 5.26 illustrates the result: two different stages of detection and analysis during the inspection are shown. The intensity image shown in Figure 5.26a is the input for the detection. The results of this first step, shown in Figure 5.26b, are the detected scale and pointer positions (which would result in a measurement value of 0.2007). These parameters are checked by the analysis, resulting in a correction of the pointer positions due to the relations defined in the description. The description provides the appropriate relations, resulting in the

fact that pointers which have not passed a value on a scale completely are considered to display the previous value. The numerical measurement value (0.1997) and the positions of the primitives illustrated in Figure 5.26c is the result.

After having demonstrated the practicability of the presented detection strategy, the working process had to be adapted to an industrial environment, since a computation time of 26 sec./frame is not acceptable in this application because the workflow forces the inspection to be finished within 3 seconds.

5.5.2 Adaptation of Inspection Process

Time is not the only constraint that influences the analysis, four other constraints have to be fulfilled for this application, as follows:

- **Speed**: The reading of the pointer has to be performed in a previously defined time of max. 3 seconds/frame.
- **Cost**: Due to industrial application and reselling strategies, the hardware cost has to be as low as possible.
- **Accuracy**: Since the automatic reading is used for calibration and industrial process monitoring, reading accuracy plays an important role.
- **Reliability**: The reliability has to be close to 100% to ensure correct calibration and monitoring.

The computation time was not attained by a SUN Sparc 10 workstation, the hardware costs of which are already too high. Therefore, we made several tradeoffs between speed, costs, accuracy and reliability in order to achieve an optimal balance of the given constraints.

Speed versus number of free parameters

The description has three image acquisition parameters which allow a very general application of the reading process to a given measuring instrument:

- **Size (*s*)**: The measuring instrument can have any size within the intensity image (of course there is a minimum size defined by resolution and reading accuracy). The change of the size is caused by the change of the distance between camera and measuring instrument.
- **Position (*x*,*y*)**: The measuring instrument can have any position in x and y direction within the image.
- **Orientation (*o*)**: The measuring instrument can have any orientation relative to the camera orientation.

When none of these parameters was fixed to a certain range, the reading process took 26 seconds. Therefore, a successive restriction of these parameters was able to reduce the computation time while decreasing the generality. First the size of the measuring instrument was fixed in the analysis process. In this way, the search space of the circle detection, detecting the measuring instrument was confined to a small number of possible radii (in respect of δs) and therefore performed with higher speed. This modification saved 60% of computation time (see Table 5.7), but forced a fixed distance between camera and measuring instrument.

Due to a modification of the final image acquisition configuration it was possible to ensure the position of the measuring instrument on a nearly fixed place within the image, restricting the position parameters to $[x \pm \delta x, y \pm \delta y]$, δx and δy being 10% of the diameter of the measuring instrument. Therefore, the processing step "measuring instrument detection" was eliminated in the analysis graph, again saving another 20% of the former computation time (see Table 5.7). The fixed imaging geometry required an accurate positioning of the camera or the measuring instrument and was obtained by the an attachment device.

Speed versus costs

Hardware costs play an important role in industrial applications if the system is to be widely used and resellable. This led us to the decision for a standard PC 486 configuration instead of a workstation, reducing the hardware cost by 80%. This cost reduction has the drawback of dramatically decreasing computation power. After having converted the software onto the PC we measured a computation time of the analysis of again 25 seconds/frame, including scale, lettering and pointer detection, determination of orientation and measurement value.

As a consequence, the last free image acquisition parameter, orientation, had to be fixed. The fixation of orientation was possible by adapting the attachment device so that it could only be attached in a certain mechanically fixed way. Subsequently the analysis was modified in the following way: only left- and rightmost circular scales were looked for in the image instead of all scales. The size of the search window was defined by the number of pixels representing the positioning error $x \pm \delta x$, $y \pm \delta y$ and the size of the circular scale (δx and δy are 5% of the diameter of the circular scale). This fixation of the position reduced the search space down to 10% of the previous search area (the complete measuring instrument). Figure 5.27 shows the search areas (left and right square) as compared with the search area of the whole image, the two detected circular scales and the two computed scales.

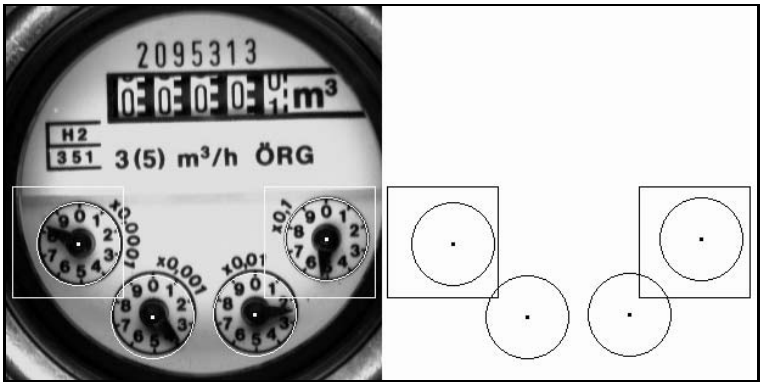


Figure 5.27 90% reduced search area (the two squares)

All other scale positions are computed out of the relative position of the centers of the two detected scales before a simple verification and the pointer detection takes place. With this adaption to the detection strategy, the computation time of 4 seconds is almost within the desired 3 seconds per frame limit with the requested accuracy. Table 5.7 summarizes the computation times in relation to the number of free parameters.

Free parameters:	*s,x,y,o*	*x,y,o*	*o*	none
Computation time workstation	26 sec.	10.4 sec.	4.8 sec.	<1 sec.
Computation time PC	-	-	25 sec.	4 sec.

Table 5.7: Computation time of analysis process

Speed versus accuracy and reliability

Due to the use of coupled pointers with discrete positions and a graduation of 10 units in a circle, the minimum reading accuracy of the pointer angle detection is ±18°. This accuracy decreases the reliability and requires very stable illumination conditions in order to be able to read the position of the leftmost pointer correctly. Therefore, a tradeoff between accuracy, reliability and speed has to be made in order to ensure the requested reliability.

The accuracy of the general detection strategy is ±3° determined by the given image size of approx. 450x450 pixels for the measuring instrument. This resulted in a computation time for the adapted process (all acquisition parameters fixed) of 4 seconds. Our tests demonstrated that an accuracy of ±8° is the best tradeoff between speed and accuracy. This reading accuracy reduced the image size to 300x300 pixels for the measuring instrument and led to a computation time of approx. 2 seconds per frame, which is within the time constraint.

The adaption of the analysis has a consequence for the analysis graph and the description, too. First the weights in the description are changed. The rightmost and leftmost circular scale still have the weight 1 which indicates that they have to be detected. The other circular scales get the weight 0, they are no longer relevant for detection. Furthermore, the rectangular scale need not be detected and all lettering elements except the country type also get the weight 0. Note that this limitation is only acceptable if there are no two types with the same country sign in the description. If a similar type is added the weights for the lettering elements in the description of the similar types have to be changed.

The analysis graph is also changed, due to the fixation of the number of free parameters. Instead of detecting the primitives, hypotheses about the position of the primitives are generated and verified. Only 3 detections in a limited area are performed. Subsequently the pointer detection, the rotary counter determination, and the serial number determination take place. In the final step, the model matching is performed by computing the measurement value the instrument is displaying. The simplified analysis graph is shown in Figure 5.28.

5.5.3 Final Inspection System Test

A final inspection system test checked whether the inspection was working correctly. The watermeter inspection process was tested comprehensively with our PC486/66 configuration.

Results of inspection system test:

- *Number of images* in test series: 200 images
- *Image size*: 300x300 pixels
- *Detection rate*: 100%
- *Accuracy*: 100%
- *Reliability*: 95%
- *Computation time*: app. 2 sec.

The reliability of a little more than 95% is due to slight changes in the illumination during acquisition and air bubbles inside the watermeter, which influence the pointer and scale detection. Although there was a rejection rate of 5%, the reading accuracy was 100%, because

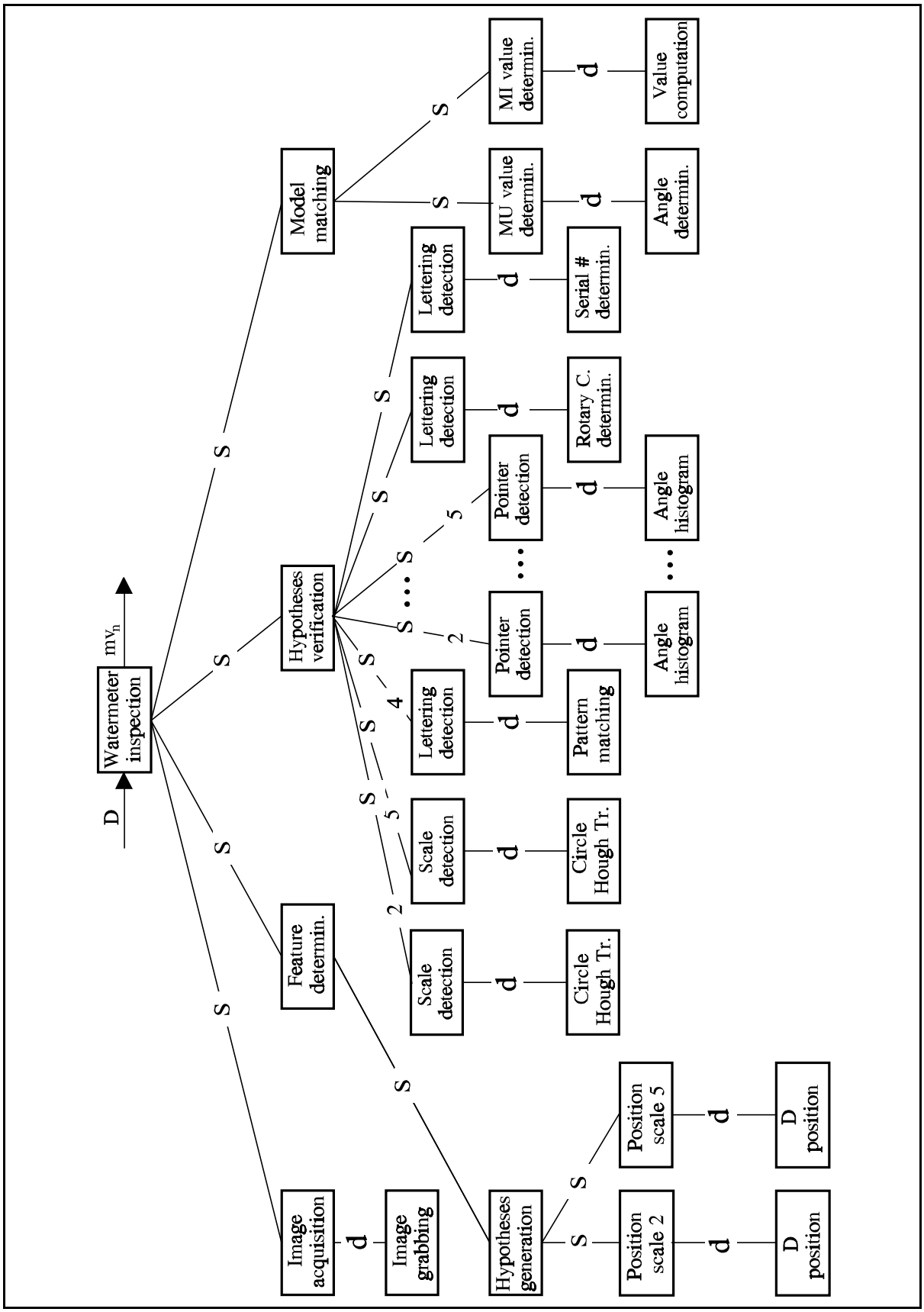


Figure 5.28 Adapted analysis graph for watermeters

error detection worked properly and non-readable images were marked and stored for visual inspection by the operator. All of the computed measurements were correct. Problems with air bubbles disturbing detection occur with rotary counter and serial number determination.

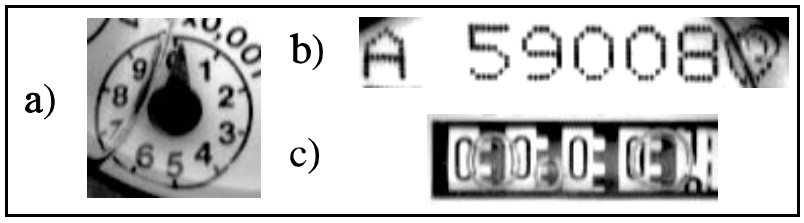


Figure 5.29 Non readable: a) pointer b) serial number c) rotary counter

Air bubbles disturb the pointer detection significantly (see Figure 5.29a). Therefore, strategies for better detection had to be considered. The angle histograms were disturbed by the air bubble borders so much that no exact maximum in the histogram could be determined. To overcome this problem a matched filter approach was used instead of the angle histogram. In this approach the bitmap of the pointer given by the description is weighted high near the symmetry axis, and low at the border. Then this "weighted filter" is rotated, with the center of the scale as pivot point. Every filter response is mapped onto an angular histogram, representing a weighted correlation function. The maximum of this function is supposed to be the position of the pointer.

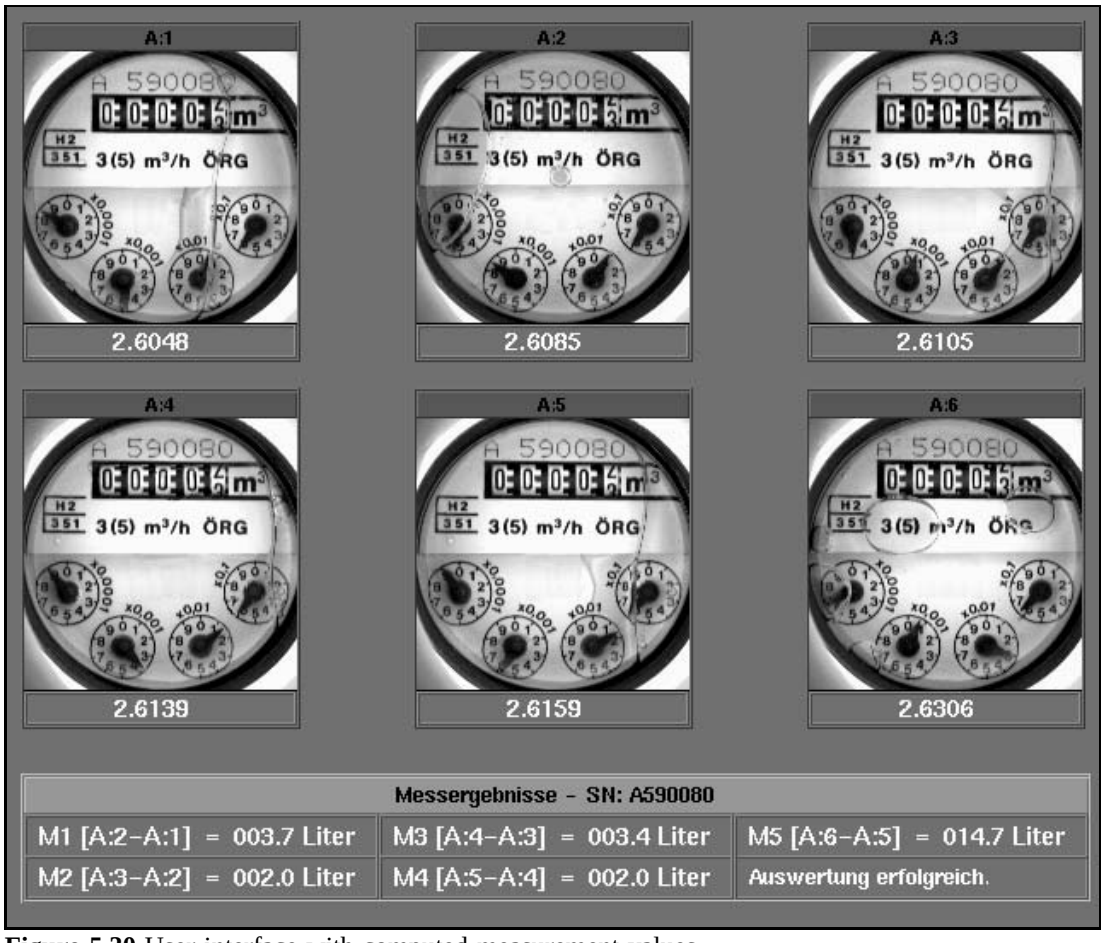


Figure 5.30 User interface with computed measurement values

The exchange of the detection algorithm did not alter the analysis graph, no adaption had to be made. The test series with the 200 images was performed again, now attaining a reliability of more than 99% for the measurement value.

For serial number and rotary counter determination other strategies than detection algorithm improvement are necessary, since the numbers are not readable even by a manual operator, due to air bubbles. In Figure 5.29b an example for an unreadable serial number is shown. The correct number would be "A 590080", the last digit is not clearly readable. The same situation may happen to the rotary counter determination as well. Figure 5.29c shows an example for a disturbed rotary counter. In this case the image is left for manual reading.

A final test series (800 frames) was conducted to determine whether the inspection system could be used for calibration. This test resulted in a rejection rate of approx. 1% (6 frames were not computed because of air bubbles that influenced the detection of circular scales and hence the pointers). Figure 5.30 shows the user interface for a complete calibration run, consisting of an initial value (A:1) and five consecutive measurements for one watermeter. Below the images the computed measurement values are shown (bottom of Figure 5.30). The amount of water between two measurement points is computed in liters (for instance M1 [A:2-A:1] = 003.7 Liter indicates that the first measurement M1 computed from the first and the second image of the calibration series is 3.7 liters of water, which is 0.0037 m³).

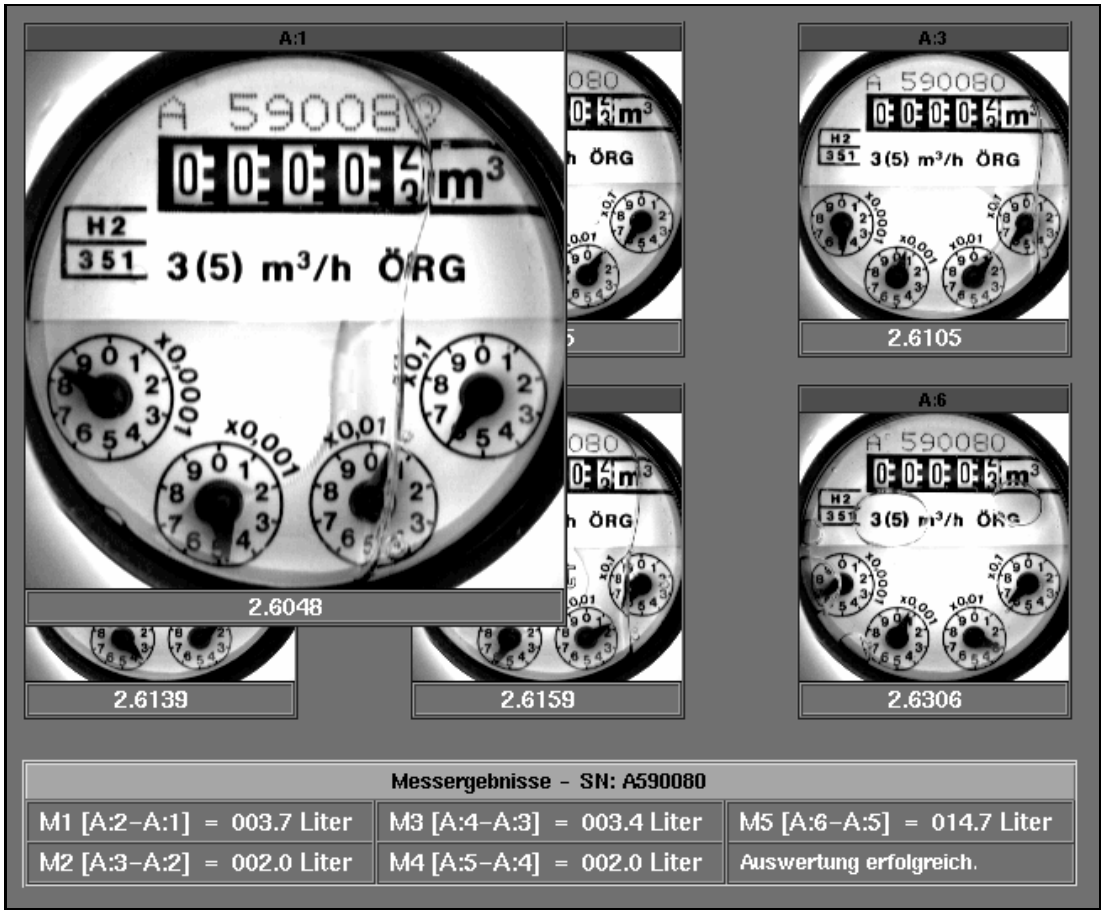


Figure 5.31 User interface for manual check and input

The air bubble problem in the case of serial number and rotary counter is solved using up to 6 intensity images of the instrument. Since air bubbles move during the calibration run, the probability that one of the images is not disturbed is higher if more images are taken. Furthermore, the meters are calibrated in an ascending numerical order of serial numbers, the predecessor and successor of one meter determine the serial number as well. Since the initial

value of the rotary counter is 0 in the calibration run, and the amount of water that runs through the calibration facility is known, and if the instrument works correctly the position of the rotary counter is also known a priori, it only has to be checked whether the counter moved. For unreadable pointers, the images are stored, and read by the human operator after the calibration run. Figure 5.31 shows the user interface for manual reading, the measurement value and the serial number may be checked and corrected interactively.

5.6 Chapter Summary

In this chapter an industrial application of the proposed concept was reported. The specific application in the field of visual inspection of analogue watermeters served as a demonstration of the concept. The industrial application of the concept to watermeter calibration showed that this concept is able to set up an application following the steps of the concept. Furthermore it was demonstrated that it can handle changes in the layout of the object (different types of watermeters) and that detection is separated from analysis since the exchange of a detection algorithm was possible without changing the analysis.

Following an analysis of the watermeter inspection workflow, image acquisition was discussed. It turned out that analyzer-polarizer combination together with a ring light is the best solution to the image acquisition problem for this specific application. The inspection model generation was performed with the help of the primitives already defined for ADI's: each watermeter has a certain number of *circular scales* (2 to 5) with corresponding *pointers*, possibly one *rectangular scale*, holding the rotary counter and the measurement unit m^3, both defined as *lettering elements*. The inspection model generation was shown on the example of four different watermeter types, two of them were used for the actual system, two of them served as examples that changes in the layout can be handled by the concept. The result of the generic detection was used to generate type dependent descriptions. The general analysis graph for ADI's was adapted for watermeter inspection. According to the shape of the features (circles, rectangles and bitmaps), detection algorithms were selected (Hough transform, template matching, angular histogram) and adapted (Burns line detector). The preliminary inspection process was generated by combining the analysis graph and the detection algorithms forming the instantiated analysis graph. The system had to be tested to determine whether it was able to perform the inspection. A test series consisting of 50 images of several watermeters of type wz_a3 resulted in a detection rate of 100% for all primitives; a positional accuracy of ±2 pixel for scales and ±3° for pointers; and an average computation time of 26 sec. The analysis graph had to be adapted, since the workflow forced the inspection to be finished within 3 seconds. The constraints: time, cost, reliability, and accuracy, were balanced, resulting in an acquisition environment forcing fixed size, position, and orientation of the object. The adapted analysis graph was again tested in a system test. In the test-series performed with 200 frames, the positions of **all scales and pointers** were determined in the **requested time** (approx. 2 sec.), with the requested **accuracy** (100%) and **reliability** (95%). The reliability of a little more than 95% was due to air bubbles inside the watermeter, which influenced the pointer and scale detection. To overcome this problem, a matched filter approach was used for pointer detection instead of the angular histogram by changing the detail relation for pointer detection in the analysis graph. The final test series with 800 images was performed, attaining a reliability of more than 99% for the measurement value.

Chapter 6

Conclusion and Outlook

The application of the proposed concept was demonstrated on the calibration of watermeters, a sub-class of multiple-scale ADI's. Without showing that this inspection system is based on a systematic approach using the inspection concept, the specific application could be seen as yet another system, tailored for a specific purpose. This work claims that the concept is flexible enough to accommodate changes in products using a systematic approach. In this concluding chapter, this flexibility is demonstrated by three examples. Furthermore, the contributions of the dissertation are summarized and an outlook on future work is given.

6.1 Flexibility of the Concept

This dissertation aimed to show a visual inspection concept that speeds up the development of AVI systems by providing a software re-use and set-up concept, adding a higher degree of flexibility to the inspection system. To show that this flexibility is attained, three examples that use the concept are demonstrated. Since there is no industrial application for these examples, the tests were performed on a series of images without paying much attention to the image acquisition. The images were taken under laboratory conditions.

The application of watermeter calibration has to handle an unlimited number of different types. The flexibility within the specific analogue instrument is provided by the description language that handles different layouts by different descriptions. The analysis graph is independent of any change of the description. Two examples were chosen to show that the description language and the analysis graph together with the appropriate detection algorithms can also handle other types of analogue display instruments:

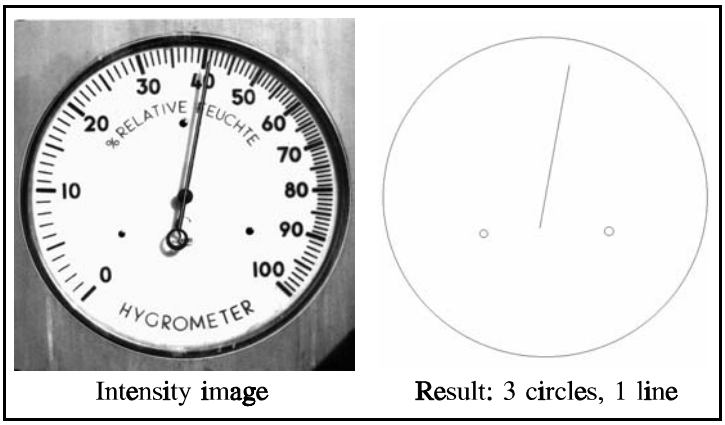


Figure 6.1 Result for a hygrometer (mv_n= 41% humidity)

1. *Hygrometer*: The description defined in Chapter 4.3. was used, and the analysis graph for watermeters was changed in such a way that the circular instrument could be detected. Furthermore, two circular lettering elements to define the orientation and the pointer were detected. Figure 6.1 shows the result for the test image; a humidity of 41% was the correct result.

2. *Clock*: With the description of the clock, generated by generic detection and interactive definition of the primitives (3 circular overlaid scales, 4 lettering elements, and 3 different pointers) the analysis produced the result shown in Figure 6.2, using the coupled pointer computation method. In our test series (20 samples without the specific case of pointers being completely overlaid) all pointer positions were computed exactly, resulting in correct reading of the time in all images [SAB96d].

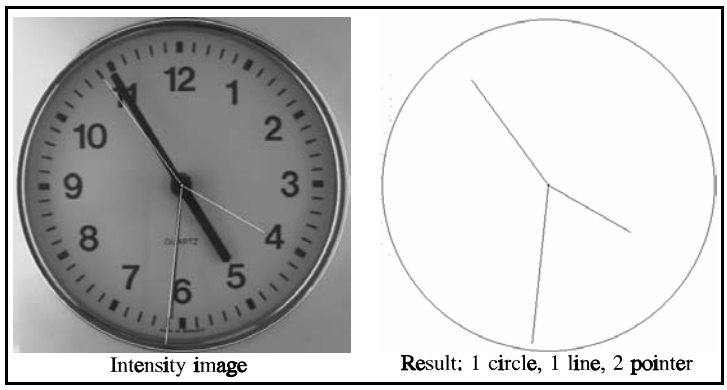


Figure 6.2 Result for a clock (mv_n = 4:54:31 h)

The working time for adapting the description and the analysis graph interactively was approx. 1.5 hours each for the hygrometer and the clock, including the generic detection phase. To solve the problem of overlapping pointers the measurement unit value determination was changed so that the minutes hand was searched first. If the other hands could not be detected, they were assumed to be overlaid by the minutes hand.

These two examples showed that an inspection system can be set up in a short time (approx. 3 hours) for any analogue instrument. To demonstrate that the concept does not only work for analogue instruments, a floppy drive casing inspection was simulated using the inspection concept. For the casing (see Figure 2.9) it was supposed that 6 drill holes and one punch had to be inspected using the concept. Using the binary silhouette shown in Figure 2.7, the generic detection produced 9 circles. Two of the circles were removed from the description, and two rectangles added interactively. The general analysis graph was adapted for the casing inspection, supposing a nearly fixed orientation of the casing within the image. The model matching was supposed to be a checking for existence. The result indicated whether all of the drill holes were present

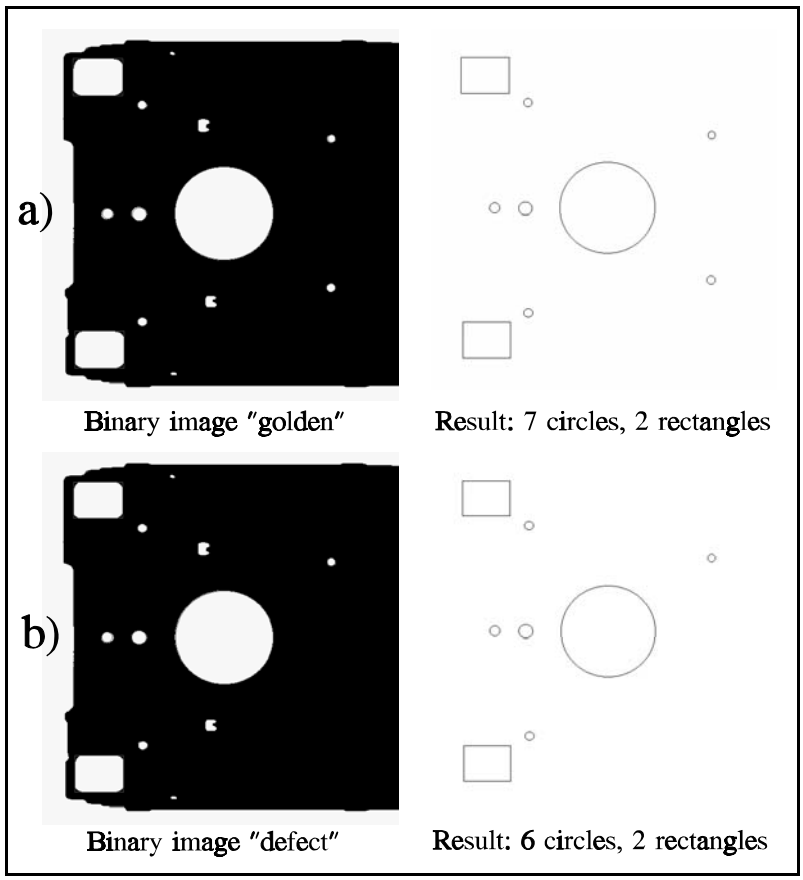


Figure 6.3 Example casing: a) all holes present b) one hole missing

or not. Figure 6.3 shows the result after having performed the inspection for a correct casing (Figure 6.3a) and a (simulated) defect casing (Figure 6.3b). Since one circle in the defect image was not detected, this image was classified as "fault". The same analysis graph (including detection algorithms) can also be used for the inspection of the object using an intensity image, since the parameters of the features do not change. Furthermore, misregistrations do not affect the inspection, since relations between primitives allow their correct detection.

The examples were simulated applications, therefore concrete data about reliability and accuracy cannot be given. Image acquisition, illumination, and image series with golden and defect images of different objects of the same type should be used to determine these parameters. However, the three additional examples showed that the claimed flexibility of the concept is given by adapting description and analysis graph to the specific object.

6.2 Thesis Summary

In this thesis a **concept for visual inspection** has been presented in which the **detection** of primitives was separated from the **analysis**. This separation was achieved by defining a general **analysis graph** for inspection, containing detail relations that represent detection algorithms. In the preliminary analysis graph it was not specified which of the algorithms should be used for the specific inspection, the selection of the algorithm was postponed to a test series. Existing pattern recognition software was re-used for detection. The use of any detection algorithm was possible by changing the analysis graph instantiation in the detail relation, the overall analysis process stayed the same. Together with an object specific description, defined in a so-called **description language**, the analysis graph was instantiated. This systematic approach to inspection allows its application to a wide range of inspection problems. It can be seen as a "recipe" for solving industrial applications, stating at which stage which kind of decisions have to be made. The systematic approach also permits a high degree of flexibility since it contains application specific and application independent parts.

The set-up of the inspection system started with the generation of the inspection model, the description language. First the geometric features that should be found in the data were defined and the use of generic algorithms to find the specific parameters of the features was discussed. The description language consisted of primitives as vocabulary and relations between primitives as grammar. Since detection and analysis were separated, detection could be modeled as detail relation in the analysis graph. The analysis itself was defined as subdivision or optional subdivision in the analysis graph. In order to detect the features during inspection, detection strategies that used defined interfaces so that different detection algorithms could be used to detect the same class of features were demonstrated. A general analysis graph for inspection was shown and other solutions to model the analysis were discussed.

The applicability of the inspection concept was demonstrated on the case study of analogue display instruments. This type of object served as a demonstration since there were various different types of measuring instruments with innumerable different displays and layouts, but all of them had certain common properties which were used to build up a specific description language. Following the definition of the primitives, the generic detection of the primitives of different ADI’s (hygrometer, manometer, thermometer, ohmmeter, rev-counter, clock, and dateclock) demonstrated the first step in the concept. The primitives together with the relations among them defined the description language for ADI’s, forming the inspection

model. With the help of the description language and the proposed general analysis graph the specific analysis graph for ADI's was defined. Examples demonstrated that this ADI analysis graph could be used to model specific instruments like manometer and dateclock.

The concept was applied to the calibration of analogue watermeters, where industrial constraints had to be fulfilled. Reading a measuring instrument meant detecting the position of scales and pointers in the intensity image and the global orientation of the instrument in order to determine the value displayed by the measuring instrument. A validation for the correct type had to be done by checking the actual, detected primitives with the elements defined in the description. The inspection determined the indicated measurement value and the serial number, protocolling each measurement step. In a test-series performed with 200 frames, the positions of **all primitives** were determined in the **requested time** (approx. 2 sec.), with the requested **accuracy** (100%) and **reliability** (95%). The reliability of a little more than 95% was due to air bubbles inside the watermeter. To overcome this problem, the detection algorithm was changed without changing the analysis. The final test series with 800 images was performed, attaining a reliability of more than 99% for the measurement value.

The extendibility of the concept was demonstrated by testing the analysis process with the description of two other ADI's (a hygrometer and a clock), which was performed by adapting the analysis graph but without changing the detection algorithms. Since the detection is represented as detail relation in the analysis graph, having a designed interface for primitives, a change of the detection algorithm is possible without changing the overall analysis. The separation of detection and analysis therefore ensures that any existing pattern recognition software can be re-used in order to solve the detection problem in inspection tasks.

6.3 Future Work

Future work will be directed towards extending the concept to other AVI domains represented by other types of data as well as towards improving the adjustment of the different parameters of the detection algorithm.

The first goal in future work is the industrial application of the concept to other 2-d inspection problems, which includes the design of acquisition and illumination devices, the acquisition of test series, the construction of description language and analysis graph and the preliminary and final inspection system test on a series of different objects of the same type.

One obvious extension would be to apply the concept to other domains, using different types of primitives, for example texture and 2-d curves, currently not represented as primitive types. Furthermore, the extension to 3-d primitives could open a large field of applications of the concept in the machine vision and in the robot vision area. Since the generic detection can handle sparse and dense data, the concept is applicable to grasping and the hand eye problem of robots by defining primitives in range images.

Another interesting challenge would be to solve the problem of automatic detection algorithm selection and parameter adjustment using knowledge based decision strategies. Progress in this research area could decrease the user interaction in selection and adjustment of detection algorithms, since the inspection system tests could be performed automatically. Together with a user interface this could help to spread this concept in the industrial domain. The description and the analysis graph have to be easily adaptable using a graphical interface; detection algorithms should be selected automatically.

Appendix A

Generic Detection

For the generic detection of parametric primitives we use the strategies presented in this appendix. In order to reduce the computational expense in the exploration step, a limited number of primitives are used (those that are within the images) and a local constraint is introduced: the seeds are not distributed randomly but based on a neighborhood criterion, seeds have to be connected. This reduced the computation time for the exploration dramatically, instead of average computing times of 4 hours for exploration on a 300x300 pixel image on a SUN Sparc 10, the actual computation time is approximately 20 minutes on the average. Selection and Fit were used unchanged to the original algorithm proposed by [STR95a].

A.1 Exploration

The purpose of the exploration is to produce a list of hypotheses which later can be used by the selection to explain the data. A rough estimate of a model which satisfies minimal conditions qualifies as a hypothesis. It is important that these absolute conditions are kept minimal. It is left to the selection to evaluate the hypotheses on a relative basis, *i.e.*, to compare the descriptive power of different configurations of hypotheses. The exploration should use all admissible types of models; again, the selection will sort out which combination of them explains the data best. Postponing all decisions and thus enumerating all possible models of all types would be astronomically expensive. Hence, the major challenge is to balance the trade-off between avoiding evaluating numerous hypotheses and a reasonable computational complexity. In [KAM90] it is assumed that the number of instances of models in the data is known. Stewart [STE94] assumes that the bad data are uniformly distributed. Such assumptions certainly lower the computational complexity, but they severely restrict the generality of the method. For a single model type, researchers working on randomized HT have proposed methods to avoid the complete enumeration of all models (see for example [LEA92, XU93]). All of them try to decide soon after a model has been detected whether it is useful. Thus, they incorporate selection features into the exploration, and as a result they lose robustness and the ability to detect models which are only supported by few boundary elements. Running a separate exploration on each connected component of the data [ROT93] restricts the generality because connected component labeling needs dense input. Furthermore, due to occlusions, boundary elements of one model might lie on different connected components and thus, their approach will either fail to detect such a model or produce unnecessarily many instantiations of the same model. Simple windowing as a way to restrict the exploration works only on dense data and for very simple model types. The generalized neighborhoods [CAL89] are more effective; however the authors mention the combinatorial explosiveness of their idea and implement a considerably restricted version. The method of growing models around seeds

[LEO93] needs dense input data.

The minimal conditions that our models must satisfy in order to qualify as hypotheses can be regarded as a more robust version of the RANSAC [BOL81] condition. We say that a *boundary element P supports a model M* if its distance from the model is less than a fixed threshold δ_{dist}, and the absolute value of the angular difference between the normals at *P* and at the closest point to *P* on *M* is less than δ_{ang}. It is usually not a problem to derive reasonable values for δ_{dist} and δ_{ang} from the knowledge of the sensor that captured the data and the operator that was used to produce the boundary elements. The usage of error tolerances can also be justified from a utilitarian point of view: The agent which requests the model extraction defines the acceptable tolerances, *i.e.*, it defines what is straight, round, etc. If the support of a model *M* of type *t* is larger than a threshold s_t, then *M* is accepted as a hypothesis. The threshold s_t can be small because the selection procedure is able to remove false positives. This enables us to accept models at different scales as hypotheses without the danger of including many false positives in the final result. This is a major advantage over RANSAC in which the consensus threshold has to be relatively large to prevent the acceptance of an overwhelming amount of false positives.

Since we neither make any assumptions about the number and the types of models in the data nor about the distribution of outliers in the data, we have to solve the exploration as a discrete search problem. To generate a hypothesis we draw a random *k*-tuple from a subset of all boundary elements, check whether it defines a model and whether the support of the model is large enough to accept it as a hypothesis. *The key idea of our exploration scheme is the dynamic, data-driven choice of the subsets from which we draw the k-tuples*. The following characterization of an efficient combinatorial search for new hypotheses can be used to define the subsets:

1. We do not want to re-discover what we have already discovered.
2. Since we do not know whether what we have already discovered is actually useful to explain the data, we want to be able to return to any part of the data in order to continue the exploration with a possibly different model type.

The authors of [CHE89,ROT93,XU93] implement only the first point by removing the supporting boundary elements of a model after the model has been detected. If this model is a false positive, *i.e.*, it is an accidental alignment of boundary elements, then their algorithms cannot recover because these boundary elements cannot be used to fit another model. In addition, the chance of detecting a false model in the next step increases due to the misclassification of boundary elements in the previous step. We avoid these problems by implementing both points. We add a temporal component to each boundary element. A boundary element that was recently used in a *k*-tuple or that was in the support of a recently detected hypothesis, is temporarily not allowed to be used in the search for a new hypothesis. This creates a dynamic that is driven by the data and the history of the exploration. Our exploration can be viewed as a masking technique in which the already detected hypotheses and the classified data define the mask which is applied to the data before the search for the next model starts. This enables us to detect models with large as well as with small support. In addition, the use of our data-driven subsets bears significant computational advantages.

A.2 Selection

The purpose of the selection is to decide which hypotheses are useful to explain the data and to remove the randomness which is present in the list of hypotheses produced by the exploration. It is necessary that the selection uses a global criterion to resolve the problems that arise from partially overlapping models of possibly different types. We argue that in most cases the globally simplest description of the data is the best one. The principle of simplicity has a long history in psychology (Gestalt principles), and the formalization of this principle in information theory led to the method of *Minimum Description Length* (MDL), which has recently found its way to computer science, including computer vision (see for example [PEN89]). According to the MDL principle, those models are selected from the set of hypotheses that describe the data with the shortest possible encoding. Models that provide an efficient encoding should encompass a large number of data points and have a high value of the goodness-of-fit measure. Where possible, a simpler model, *i.e.*, one with lower degrees of freedom, should be selected.

If we measure the benefit of using a certain subset of the hypotheses with an objective function, then the selection becomes a global, discrete optimization problem. More precisely, it is a Boolean optimization problem. If the exploration produces n hypotheses, then there exist 2^n different subsets. For a modest number of hypotheses it is already not tractable to evaluate the objective function for all subsets in order to find the best one. How can a practically acceptable solution with a reasonable computational complexity be obtained? There exist various discrete optimization approaches in the literature that can be employed to solve the selection. From the stochastic approach the simulated and microcanonical annealing are the most prominent ones. Their deterministic counterpart - mean field - annealing can be used as well. It has been shown in [HER93] that on combinatorial optimization problems the latter two are one to two orders of magnitude faster than simulated annealing. Hopfield and Tank [HOP85] proposed a neural network for solving these types of problems. The network, which implements a gradient ascent method, produces good results in cases for which correct solutions cluster, but does not allow recovery from local extrema. Pentland's modification [PEN89], which embeds Hopfield's neural network in a continuation method, can only alleviate but not remove this problem. Genetic algorithms [BHA89], which perform a global search by evaluating and reproducing populations of solution vectors distributed throughout the search space, can in principle avoid the problem of local minima, but their computational complexity makes them infeasible for practical applications (at least on a sequential machine). Under the assumption that the input can be described by the set of admissible models, it is argued in [LEO93] that a straight forward application of steepest ascent yields reasonable solutions, even though it is only a local optimization method.

We propose to use *tabu search* [GLO93] in connection with an objective function based on MDL for the selection. Tabu search is a general heuristic procedure for global optimization. Its strategy can be viewed as a smart extension of a steepest ascent method. Unlike the various annealing methods which try to exploit a questionable analogy with a physical phenomenon, tabu search looks more like an intelligent search which may in some sense imitate human behavior. Tabu search seems to be unknown in the computer vision community even though it has recently received a lot of attention in operations research because it outperforms the annealing techniques on many classical operations research problems [GLO93]. Our implementation of the selection confirms the superiority of tabu search over annealing tech-

niques. Tabu search is computationally more expensive than simple steepest ascent because it is a global optimization technique. However, if models of different types are involved, then the extra computational cost outweighs the sub-optimal solutions of a simple steepest ascent algorithm.

A.3 Fit

After the selection we have good estimates of the models that can be used to explain the set of boundary elements. Since it is our goal to accurately describe the set of boundary elements with parametric models, we use these estimates to initialize a robust fitting algorithm. In the classify-then-fit paradigm we would run an algorithm that fits a model to a fixed set of boundary elements, namely the boundary elements that support the initial estimate of the model. Since the classification of the boundary elements was based on an estimate of the model, it might also need to be improved in order to produce an accurate fit. The symbiotic relationship between classification and fitting suggests the use of a classify-and-fit paradigm, in which the intermediate results of the fitting are used to improve the classification of the boundary elements and the improved classification in turn helps to improve the fitting. In such a paradigm, the fitting procedure has to monitor the data constantly in order to change the classification according to the progress of the fitting.

The modified least squares algorithm presented in [STR94] follows the classify-and-fit paradigm; two extensions are needed in order to make it robust in the presence of multiple models. If the initial estimates of the models are good enough, then the modified least squares algorithm converges with high accuracy to the local maximum likelihood position of the models. Most other robust fitting techniques sacrifice accuracy for robustness and do not converge to a local maximum likelihood position, *e.g.* median methods.

It is possible that several estimates converge to the same model since the fitting can change the classification of the boundary elements. This makes a final selection necessary.

References

[ABD79] I.E. Abdou, W.K. Pratt, "Quantitative Design and Evaluation of Enhancement / Thresholding Edge Detectors", *Proc. IEEE*, Vol.67, No.5, pp.753-763, 1979.

[AGG89] J.K. Aggarwal, C.H. Chien, "3-D Structures from 2-D Images", in: J. Sanz (Ed.), "*Advances in Machine Vision*", Springer Verlag, New York, pp.64-121, 1989.

[AHL91] R.J. Ahlers, H.J. Warnecke, *"Industrielle Bildverarbeitung"*, Addison-Wesley Publishing Company, Bonn, Muenchen, New York, Sidney, Tokyo, 1991.

[AHO72] A.V. Aho, J.D. Ullmann, "*The Theory of Parsing, Translation, and Compiling*", Vol.1, Prentice-Hall, Englewood Cliffs, New Jersey, 1972.

[AME95] Amerinex Artificial Intelligence Inc., 409 Main Street, Amherst, MA 01002, USA, "*The KBVision System - User's Guide*", 1995.

[ASC92] P. Aschwanden, W. Guggenbühl, "Experimental Results from a Comparative Study on Correlation-Type Registration Algorithms", *Robust Computer Vision*, Wichman Verlag, Karlsruhe, 1992.

[ATT54] F. Attneave, "Some Informational Aspects of Visual Perception", *Psychological Review*, Vol.61, No.3, pp.183-193, 1954.

[BAE91] H. Baessmann, P.W. Besslich, "*Bildverarbeitung Ad Oculos*", Springer Verlag, Berlin, Heidelberg, New York, 1991.

[BAI83] M.L. Baird, "GAGESIGHT: A Computer Vision System for Automatic Inspection of Instrument Gauges", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.5, No.6, pp.618-621, 1983.

[BAL81] D.H. Ballard, C.M. Brown, "Generalizing the Hough Transform to Detect Arbitrary Shapes", *Pattern Recognition*, Vol.13, No.2, pp.111-122, 1981.

[BAR81] A. Barr, E.A. Feigenbaum, "*The Handbook of Artificial Intelligence*", Vol.1, Vol.2, Pitmann Books, London, 1981.

[BAT78] B.G. Batchelor, "*Pattern Recognition - Ideas in Practice*", Plenum Press, New York, London, 1978.

[BAT85] B.G. Batchelor, D.A. Hill, D.C. Hodgson, "*Automated Visual Inspection*", IFS, Bedford, UK and North-Holland, The Netherlands, 1985.

[BAT92] B.G. Batchelor, D. Braggins, "Commercial Vision Systems", in: C. Torras (Ed.), "*Computer Vision: Theory and Industrial Applications"*, Springer Verlag, New York, pp.405-452, 1992.

[BAT94] B.G. Batchelor, P.F. Whelan, "Machine Vision Systems: Proverbs, Principles, Prejudices and Priorities", *Proc. of Workshop in Machine Vision Applications, Architectures, and Systems Integration III, Boston*, Proc. Vol.SPIE-2347, pp.374-383, 1994.

[BAT95] B.G. Batchelor, P.F. Whelan, "Ethical, Environmental and Social Issues for Machine Vision in Manufacturing Industry", *Proc. of Conf. on Machine Vision Applications, Architectures and Systems IV, Philadelphia*, Proc. Vol.SPIE-2597, pp.2-17, 1995.

[BAT96] B.G. Batchelor, P.F. Whelan, "Machine Vision Systems: Proverbs, Principles, Prejudices and Priorities", *Proc. of SME Conf. on Applied Machine Vision*, Cincinnati, pp.7-19, 1996.

[BEN91] K. Bengoetxea, "Lighting Setup in the Automatic Detection of Ventral Skin and Blood Spots in Cod Fish Filet", in: L. Pau, R. Olafsson (Eds.), "*Fish Quality Control by Computer Vision*", Dekker, New York, pp.121-150, 1991.

[BHA89] B. Bhanu, S. Lee, J. Ming, "Adaptive Image Segmentation Using a Genetic Algorithm", *Proc. of Image Understanding Workshop*, pp.1043-1055, DARPA, May 1989.

[BHA94] B. Bhanu, "Special Issue on Innovative Applications of Computer Vision", *Machine Vision and Applications*, Vol.7, No.3, pp.125-218, 1994.

[BJO77] C.M. Bjorklund, T. Pavlidis, "On the Automated Inspection and Description of Printed Wiring Boards", *Proc. of the Int. Conf. Cybern. Soc.*, pp.690-693, 1977.

[BOD95] R. Bodington, "A Software Environment for the Automatic Configuration of Inspection Systems", *Proc. of 1st Intl. Workshop on Knowledge-based Systems for the (Re)Use of Program Libraries*, Sophia Antipolis, France, Vol.1, pp.100-108, 1995.

[BOL81] R.C. Bolles, M.A. Fischler, "A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", *Comm. ACM*, Vol.24, No.6, pp.381-395, 1981.

[BOW91] K.W. Bowyer, C.R. Dyer, "Aspect Graphs: An Introduction an Survey of Recent Results", *Int. Jour. Imaging Syst. Technol.*, Vol.2, pp.315-328, 1991.

[BOW94] K.W. Bowyer, C.R. Dyer, "Three-Dimensional Shape Representation", in: T.Y. Young, (Ed.), "*Handbook of Pattern Recognition and Image Processing: Computer Vision*", Vol.2, Academic Press, San Diego, London, 1994.

[BUN82] H. Bunke, "Attributed Programmed Graph Grammars and Their Application to Schematic Diagram Interpretation", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.4, No.6, pp.574-582, 1982.

[BUN92] H. Bunke, A. Sanfeliu, "Statistical and Syntactic Models and Pattern Recognition Techniques", in: C. Torras (Ed.), "*Computer Vision: Theory and Industrial Applications"*, Springer Verlag, New York, pp.215-266, 1992.

[BUR86] J.B. Burns, A.R. Hanson, E.M. Riseman, "Extracting Straight Lines", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.8, No.4, pp.425-455, 1986.

[CAL89] A. Califano, R.M. Bolle, R.W. Taylor, "Generalized Neighborhoods: A New Approach to Complex Parameter Feature Extraction", *Proc. of the CVPR89*, pp.192-199, 1989.

[CAN86] J. Canny, "A Computational Approach to Edge Detection", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.8, No.6, pp.679-698, 1986.

[CHA90] J.P. Chan, B.G. Batchelor, I.P. Harris, S.J. Perry, "Intelligent Visual Inspection of Food Products", *Proc. of the SPIE Conf. on Machine Vision Systems Integration in Industry, Boston*, Vol.1386, pp.171-179, 1990.

[CHE89] D.S. Chen, "A Data-driven Intermediate Level Feature Extraction Algorithm", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.11, No.7, pp.749-758, July 1989.

[CHI82] R.T. Chin, C.A. Harlow, "Automated Visual Inspection - A Survey", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.4, No.6, pp.557-573, 1982.

[CHI86] R.T. Chin, "Algorithms and Techniques for Automated Visual Inspection", in: T.Y. Young, K.S. Fu (Eds.), "*Handbook of Pattern Recognition and Image Processing*", Academic Press, San Diego, London, pp.587-612, 1986.

[CHI88] R.T. Chin, "Automated Visual Inspection - 1981-87", *Comp. Vision, Graphics, Image Processing*, Vol.41, pp.346-381, 1988.

[CHI92] R.T. Chin, "Automated Visual Inspection Algorithms", in: C. Torras (Ed.), "*Computer Vision: Theory and Industrial Applications"*, Springer Verlag, New York, pp.377-404, 1992.

[CLE89] V. Clement, M. Thonnat, "Handling Knowledge in Image Processing Libraries to build Automatic Systems", *Proc. of Int. Workshop on Machine Intelligence and Vision, Tokyo*, pp.187-192, 1989.

[CLE93] V. Clement, M. Thonnat, "Integration of Image Processing Procedures, Ocapi: A Knowledge-based Approach", *Comp. Vision, Graphics, Image Processing: Image Understanding*, Vol.57, No.2, pp.164-184, 1993.

[CLO95] R. Clouard, C. Porquet, A. Elmoataz, M. Revenu, "Why Building Knowledge-based Image Segmentation Systems is so Difficult", *Proc. of 1st Intl. Workshop on Knowledge-based Systems for the (Re)Use of Program Libraries*, pp.138-148, Sophia Antipolis, France, November 23-24, 1995.

[COW89] C.K. Cowan, A. Bergman, "Determining the Camera and Light Source Location for a Visual Task", *Proc. of IEEE Intl. Conf. on Robotics and Automation, Scottsdale*, Vol.1, pp.509-514, 1989.

[CRE93] D. Crevier, "Expert Systems as Design Aids for Artificial Vision Systems: A Survey", *Proc. of SPIE*, Vol.2055, pp.84-96, 1993.

[CRO83] G.R. Cross, A.K. Jain, "Markov Random Field Texture Models", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.5, No.1, pp.25-39, 1983.

[DAL91] W. Daley, R.A. Carey, "Color Machine Vision for Defect Detection (Algorithms and Techniques)", *Proc. of Intl. Robots and Vision Automation Conf., Detroit*, pp.5.29-5.42, 1991.

[DAN90] P.E. Danielsson, O. Seger, "Rotation Invariance in Gradient and Higher Order Derivate Detectors", *Comp. Vision, Graphics, Image Processing*, Vol.49, pp.198-221, 1990.

[DAR88] A.M. Darwish, A.K. Jain, "A Rule-based Approach for Visual Pattern Inspection", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.10, No.1, pp.56-58, 1988.

[DBO96] C. de Boer, A.W.M. Smeulders, "BESSI: An Experimentation System for Vision Module Evaluation", *Proc. of the 13th. Intl. Conf. on Pattern Recognition, Vienna*, Vol.3, pp.109-113, 1996.

[DEU78] E.S. Deutsch, J.R. Fram, "A Quantitative Study of the Orientation Bias of Some Edge Detector Schemes", *IEEE Trans. on Comp.*, Vol.27, No.3, pp.205-213, 1978.

[DOW86] E. Downing, K. Doyle, "Test and Inspection: II Lights, Camera, Inspection", *Instrum. Control Syst. - Ind. Process Control Mag.*, Vol.59, No.6, pp.71-80, 1986.

[DUD75] R.O. Duda, P.E. Hart, "Use of the Hough Transform to Detect Lines and Curves in Pictures", *Com. of ACM*, Vol.18, No.9, pp.509-517, 1975.

[DYE83] C.R. Dyer, "Gauge Inspection Using Hough Transforms", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.5, No.6, pp.621-623, 1983.

[EBE91] G. Ebenbichler, "*Erkennen von analog und digital dargestellten Uhrzeiten: Veränderungen der langsamen, negativen Hirnpotentiale und 99m-Tc HMPAO SPECT Studie*", PhD, University of Innsbruck, 1991.

[EVE79] S. Even, "*Graph Algorithms*", Computer Science Press, Rockville, 1979.

[FRA75] J.R. Fram, E.S. Deutsch, "On the Quantitative Evaluation of Edge Detection Schemes and Their Comparison with Human Performance", *IEEE Trans. on Comp.*, Vol.24, No.6, pp.616-628, 1975.

[FRE61] H. Freeman, "On the Encoding of Arbitrary Geometric Configurations", *IRE. Trans. on Electron. Comp.*, Vol.10, pp.260-268, 1961.

[FRE80] H. Freeman, I. Chakravarty, "The Use of Characteristic Views in the Recognition of Three-Dimensional Objects", in: E. Gelsema, L. Kanal, (Eds.), "*Pattern Recognition in Practice*", North Holland Publishing Company, Amsterdam, 1980.

[FRE89] H. Freeman, M.Y. Chiu, D.D. Dreyfuss, I. Gorog, "Is Industry Ready for Machine Vision?", in: H. Freeman (Ed.), "*Machine Vision for Inspection and Measurement*", Academic Press, San Diego, Inc., 1989.

[FU77] K.S. Fu, "Error-correcting Parsing for Syntactic Pattern Recognition", in: A. Klinger et. alt. (Eds.), "*Data Structure, Computer Graphics and Pattern Recognition*", Academic Press, New York, 1977.

[FU81] K.S. Fu, J.K. Mui, "A Survey on Image Segmentation", *Pattern Recognition*, Vol.13, pp.3-16, 1981.

[FU82a] K.S. Fu, "*Syntactic Pattern Recognition and Applications*", Prentice-Hall, Englewood Cliffs, New Jersey, 1982.

[FU82b] K.S. Fu, "A General (Syntactic-semantic) Approach to Picture Analysis", in: K.S. Fu, T.L. Kunii (Eds.), "*Picture Engineering*", Springer Verlag, New York, 1982.

[FU86] K.S. Fu, "Syntactic Pattern Recognition", in: T.Y. Young, K.S. Fu (Eds.), "*Handbook of Pattern Recognition and Image Processing*", Academic Press, San Diego, London, pp.84-117, 1986.

[FUK72] K. Fukunada, "*Introduction to Statistical Pattern Recognition*", Academic Press, New York, 1972.

[GAL90] L.J. Galbiati, "*Machine Vision and Digital Image Processing Fundamentals*", Prentice-Hall, Englewood Cliffs, New Jersey, 1990.

[GLO93] F. Glover, M. Laguna, "Tabu search", In: C.R. Reeves (Ed.),"*Modern Heuristic Techniques for Combinatorial Problems*", pp.70-150, Blackwell Scientific Publications, Osney Mead, Oxford OX2 0El, GB, 1993.

[GON78] R.C. Gonzalez, M.G. Thomason, "*Syntactic Pattern Recognition: An Introduction*", Addison Wesley, Reading, 1978.

[HAR68] P.E. Hart, N.J. Nilsson, B. Raphael, "A Formal Basis for the Heuristic Determination of Minimum Cost Paths", *IEEE Trans. Syst. Sci. Cybern.*, Vol.4, No.2, pp.100-107, 1968.

[HAR85] R.M. Haralick, L.G. Shapiro, "Image Segmentation Techniques", *Comp. Vision, Graphics, Image Processing*, Vol.29, pp.100-132, 1985.

[HAR86] R.M. Haralick, "Statistical Image Texture Analysis", in: T.Y. Young, K.S. Fu (Eds.), "*Handbook of Pattern Recognition and Image Processing*", Academic Press, San Diego, London, pp.233-280, 1986.

[HAR91] R.M. Haralick, G.L. Shapiro, "Glossary of Computer Vision Terms", *Pattern Recognition*, Vol.24, No.1, pp.69-93, 1991.

[HAR94] R.M. Haralick, "Performance Characterization in Computer Vision", *Image Understanding*, Vol.60, pp.245-249, 1994.

[HAT78] M. Hattori, M. Kawamura, "Automation of the Adjustment Process in Pattern Recognition (Meter Gauge)", *Proc. of 2nd Tech. Conf. on Rationalization of Product and Improvement of Production Technology*, pp.81-82, 1978.

[HEC69] H. Hecaen, "Aphasic, Apraxic and Syndromes in Right and Left Hemisphere Lesions", in P.J. Vinken, G.W. Bruyn, M. Critchley, J.A.M. Fredericks (Eds.), "*Handbook of Clinical Neurology*", Vol.4, pp.291-311, Wiley, Amsterdam, New York, 1969.

[HED89] K. Hedengren, "Methodology for Automatic Image-based Inspection of Industrial Objects", in: J. Sanz (Ed.), "*Advances in Machine Vision*", Springer Verlag, New York, pp.160-191, 1989.

[HER93] L. Herault, R. Horaud, "Figure-ground Discrimination: A Combinatorial Optimization Approach", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.15, No.9, pp.899-914, September 1993.

[HIL85a] A.G. Hill, "The Context of Automated Inspection", in: B.G. Batchelor, D.A. Hill, D.C. Hodgson, (Eds.), "*Automated Visual Inspection*", IFS, Bedford, UK and North-Holland, The Netherlands, 1985.

[HIL85b] A.G. Hill, "Economic Considerations", in: B.G. Batchelor, D.A. Hill, D.C. Hodgson, (Eds.), "*Automated Visual Inspection*", IFS, Bedford, UK and North-Holland, The Netherlands, 1985.

[HOL84] J. Hollingum, "*Machine Vision; the Eyes of Automation*", IFS, Kempston, 1984.

[HOP79] J.E. Hopcroft, J.D. Ullman, "*Introduction to Automata Theory, Languages and Computations*", Addison-Wesley, Reading, 1979.

[HOP85] J.J. Hopfield, D.W. Tank, "Neural Computation of Decisions in Optimization Problems", *Biological Cybernetics*, Vol.52, pp.141-152, 1985.

[HOU62] P.V.C. Hough, "Methods and Means for Recognizing Complex Patterns", *U.S. Patent* 3,069,654, 1962.

[HUN85] V.D. Hunt, "*Smart Robots: A Handbook of Intelligent Robotic Systems*", Chapman and Hall, New York, 1985.

[ISO94] International Organization for Standardization, "*ISO 9000: International Standards for Quality Management*", 4. Ed., Genève, 1994.

[JAI90] R.C. Jain, A.K. Jain, "*Analysis and Interpretation of Range Images*", Springer Verlag, New York, 1990.

[JAR78] J.F. Jarvis, "Automatic Visual Inspection of Western Electric Type 700 Connectors", *Proc. of IEEE Com. Soc. Conf. on Pattern Recognition and Image Processing, Troy*, pp.143-150, 1978.

[JAR80] J.F. Jarvis, "A Method for Automating the Visual Inspection of Printed Wiring Boards", *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.2, pp.77-82, 1980.

[KAM90] B. Kamgar-Parsi, B. Kamgar-Parsi, H. Wechsler, "Simultaneous Fitting of Several Planes to Point Sets Using Neural Networks", *Comp. Vision, Graphics, and Image Processing*, Vol.52, pp.341-359, 1990.

[KAU95] H. Kauppinen, O. Silven, "A Color Vision Approach for Grading Lumber", *Proc. of 9th Scandinavian Conf. on Image Analysis, Uppsala*, Vol.1, pp.41-48, 1995.

[KHO95] Khoral Research Inc., USA, "*KHOROS Program Services - User's Guide*", 1995.

[KIE92] P. Kierkegaard, "A Method for Detection of Circular Arcs Based on the Hough Transform", *Machine Vision and Applications*, Vol.5, pp.249-263, 1992.

[KIT81] L. Kittchen, A. Rosenfeld, "Edge Evaluation Using Local Edge Coherence", *IEEE Trans. Syst., Man, Cybernet.*, Vol.11, No.9, pp.597-605, 1981.

[LEA92] V.F. Leavers, "The Dynamic Generalized Hough Transform: Its Relation to the Probabilistic Hough Transform and an Application to the Concurrent Detection of Circles and Ellipses", *Comp. Vision, Graphics, Image Processing: Image Understanding*, Vol.56, No.3, pp.381-398, Nov. 1992.

[LEO93] A. Leonardis, "*Image Analysis Using Parametric Models: Model-Recovery and Model-selection Paradigm*", PhD thesis, University of Ljubljana, Faculty of Electrical Engineering and Computer Science, Slovenia, May 1993.

[LIU95] M.Y. Liu, D.W. Wang, H. Shi, "Machine Vision Inspection System for Automobile Gauge Panel", *Proc. of the IS&T/SPIE Symposium on Machine Vision Applications in Industrial Inspection III, San Jose, CA*, SPIE Vol.2424, pp.367-376, 1995.

[MAR82] D. Marr, "*Vision*", Freeman, 1982.

[MAR92] A.D. Marshall, R.R. Martin, "*Computer Vision, Models and Inspection*", World Scientific, 1992.

[MAT89] T. Matsuyama, "Expert Systems for Image Processing: Knowledge-based Composition of Image Analysis Processes", *Comp. Vision, Graphics, Image Processing*, Vol.48, pp.22-49, 1989.

[MAT92] Matrox (UK) Ltd., 6 Cherry Orchard West, Kembrey Park, Swindon, SN2 6UP, UK, "*Matrox Image Series - Product Overview*", 1992.

[MAT95] Matrox Electronic Systems Ltd., 1025 St. Regis Blvd., Dorval, Quebec, H4P 2T4, Canada, "*Matrox Inspector - Product Overview*", 1995.

[MAT96] Matrox Electronic Systems Ltd., 1025 St. Regis Blvd., Dorval, Quebec, H4P 2T4, Canada, "*Matrox Imaging Library (MIL) - Product Overview*", 1996.

[MIL95] H. Mili, F. Mili, A. Mili, "Reusing Software: Issues and Research Directions", *IEEE Trans. on Software Engineering*, Vol.21, No.6, pp.528-562, 1995.

[MIS83] S. Misra, C.A. Bennett, "Lighting for Visual Inspection Task", *Light. Des. Appl.*, Vol.13, No.1, pp.32-33, 1983.

[MOR92] S. Mori, C.Y. Suen, K. Yamamoto, "Historical Review of OCR Research and Development", *Proc. of the 11th Intl. Conf. on Pattern Recognition, The Hague*, pp.754-760, 1992.

[MUR95] H. Murase, S.K. Nayar, "Three-Dimensional Object Recognition from Appearance: Parametric Eigenspace Method", *Systems and Computers in Japan*, Vol.26, pp.45-53, 1995.

[NA95] A. Na, D. Zhao, M. Shridhar, "Vision-based Defect Detection Method for Fine-pitch Surface-mounted Devices", *Proc. of the IS&T/SPIE Symposium on Machine Vision Applications in Industrial Inspection III, San Jose, CA*, SPIE Vol.2423, pp.2-13, 1995.

[NAL93] V.S. Nalwa, "*A Guided Tour of Computer Vision*", Addison-Wesley Publishing Company, Bonn, Muenchen, New York, Sidney, Tokyo, 1993.

[NEV86] R. Nevatia, "Image Segmentation", in: T.Y. Young, K.S. Fu (Eds.), "*Handbook of Pattern Recognition and Image Processing*", Academic Press, San Diego, London, pp.215-245, 1986.

[NEW95] T.S. Newman, A.K. Jain, "A Survey of Automated Visual Inspection", *Comp. Vision, Graphics, Image Processing*, Vol.61, No.2, pp.231-262, 1995.

[NOB93] A. Noble, J. Mundy, "Toward Template-based Tolerancing from a Bayesian Viewpoint", *Proc. of the IEEE Conf. on Computer Vision and Pattern Recognition, New York*, pp.246-252, 1993.

[NOV95] J.L. Novack, "*The ISO 9000 Documentation Toolkit: 1994 revised ISO 9001 Standard*", Prentice Hall, Englewood Cliffs, New Jersey, 1995.

[PAV80] T. Pavlidis, "*Structural Pattern Recognition*", 2nd Edition, Springer Verlag, Berlin, 1980.

[PEA84] J. Pearl, "*Heuristics*", Addison-Wesley, Reading, 1984.

[PEN89] A.P. Pentland, "Part Segmentation for Object Recognition", *Neural Comp.*, Vol.1, pp.82-91, 1989.

[PEN95] J. Penix, P. Alexander, "Design Representation for Automating Software Component Reuse", *Proc. of 1st Intl. Workshop on Knowledge-Based Systems for the (Re)Use of Program Libraries*, Sophia Antipolis, France, Vol.1, pp.76-84, 1995.

[PET87] D. Petkovic, E.B. Hinkle, "A Rule-based System for Verifying Engineering Specifications in Industrial Visual Inspection Applications, *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol.9, No.2, pp.306-311, 1987.

[PET95] E.D. Peters, (Ed.), "*ISO 9000 Almanac 1994-1995*", Irwin, Homewood, 1994.

[PIN69] K.K. Pingle, "Visual Perception by a Computer", in: A. Grasselli (Ed.), "*Automatic Interpretation and Classification of Images*", Academic Press, New York, pp.277-284, 1969.

[POG85] T. Poggio, C. Koch, V. Torre, "Computational Vision and Regularization Theory", *Nature*, Vol.317, pp.314-319, 1985.

[POU89] D. Poussart, D. Laurendeau, "3-D Sensing for Industrial Computer Vision", in: J. Sanz (Ed.), "*Advances in Machine Vision*", Springer Verlag, New York, pp.122-159, 1989.

[PRA78] W.K. Pratt, "*Digital Image Processing*", John Wiley & Sons, New York, 1978.

[PRA91] W.K. Pratt, "*Digital Image Processing*", 2nd Edition, John Wiley & Sons, New York, 1991.

[PRE70] J.M.S. Prewitt, "Object Enhancement and Extraction", in: B.S. Lipkin, A. Rosenfeld, "*Picture Processing and Psychopictoris*", Academic Press, New York, pp.75-149, 1970.

[QUI68] M.R. Quillian, "Semantic Memory", in: M. Minsky (Ed.), "*Semantic Information Processing*", M.I.T. Press, Cambridge, 1968.

[ROD95] P.A. Roder, "Algorithm Design for Use in Cross-sectional X-ray Image Analysis of Solder Joints", pp.46-57, *Proc. of the IS&T/SPIE Symposium on Machine Vision Applications in Industrial Inspection III, San Jose, CA*, SPIE Vol.2423, 1995.

[ROS79] A. Rosenfeld, "*Picture Languages - Formal Models for Picture Recognition*", Academic Press, New York, 1979.

[ROS82] A. Rosenfeld, A.C. Kak, "*Digital Picture Processing*", 2nd Edition, Vols. 1 & 2, Academic Press, New York, 1982.

[ROT93] G. Roth, M.D. Levine, "Extracting Geometric Primitives", *Comp. Vision, Graphics, Image Processing: Image Understanding*, Vol.58, pp.1-22, July 1993.

[SAB94a] R. Sablatnig, W.G. Kropatsch, "Automatic Reading of Analog Display Instruments", *Proc. of the 12th Intl. Conf. on Pattern Recognition, Jerusalem*, October 9-13, pp.794-797, 1994.

[SAB94b] R. Sablatnig, W.G. Kropatsch, "Application Constraints in the Design of an Automatic Reading Device for Analog Display Instruments", *Proc. of the 2nd IEEE Workshop on Applications of Computer Vision*, pp.205-212, Sarasota, FLA, December 5-7, 1994.

[SAB95a] R. Sablatnig, C. Hansen, "Machine Vision for Automatic Calibration of Analog Display Instruments", *Proc. of the IS&T/SPIE Symposium on Machine Vision Applications in Industrial Inspection III, San Jose, CA*, SPIE Vol.2424, pp.356-366, 1995.

[SAB95b] R. Sablatnig, C. Hansen, "Automatic Calibration of Watermeters", in: F. Solina, W.G. Kropatsch, (Eds.), "*Visual Modules: Proc. of 19th ÖAGM and 1st SDRV Workshop*", Schriftenreihe der OCG, Vol.81, pp.231-239, Oldenburg, Wien, München, 1995.

[SAB95c] R. Sablatnig, "Automatische Ablesung von Wasserzählern zur Qualitätssicherung bei der Eichung", in: G. Sagerer, S. Posch, F. Kummert, (Eds.), "*Mustererkennung 95, 17. DAGM Symposium, Bielefeld*", Informatik aktuell, pp. 335-342, Springer Verlag, Berlin, 1995.

[SAB95d] R. Sablatnig, "Visual Inspection of Watermeters Used for Automatic Calibration", in: R.T. Chin, H.H.S. Ip, A.C. Naiman, T-C. Pong, (Eds.), "*Image Analysis Applications and Computer Graphics*", Lecture Notes in Computer Science, Vol.1024, pp.518-520, Springer Verlag, New York 1995.

[SAB95e] R. Sablatnig, "A Highly Adaptable Visual Inspection Concept by Separating Detection and Analysis Process", *Proc. of 1st Intl. Workshop on Knowledge-Based Systems for the (Re)Use of Program Libraries*, Sophia Antipolis, France, Vol.2, pp.25-27, 1995.

[SAB96a] R. Sablatnig, A. Leonardis, "A Modular Automatic Visual Inspection Concept based on Generic Detection", *Proc. of SME Conf. on Applied Machine Vision*, Cincinnati, Ohio, pp.339-358, 1996.

[SAB96b] R. Sablatnig, A. Leonardis, "New Concepts in Automatic Visual Inspection", in: N. Pavesic, H. Niemann, S. Kovacic, F. Mihelic (Eds.), "*Speech and Image Understanding, Proc. of the 3rd Slovenian-German Workshop and 2nd SDRV Workshop, Ljubljana*", pp.321-330, 1996.

[SAB96c] R. Sablatnig, "Flexible Automatic Visual Inspection based on the Separation of Detection and Analysis", *Proc. of 13th Int. Conf. on Pattern Recognition, Vienna*, Vol.3, pp.944-948, IEEE-Computer Society Press, 1996.

[SAB96d] R. Sablatnig, A. Leonardis, "A General Approach to Automatic Visual Inspection", in Lee G.K. (Ed.), "*Proc. of the ISCA 9th Intl. Conf. Computer Applications in Industry and Engineering, Orlando*", pp.56-59, 1996.

[SAB96e] R. Sablatnig, A. Leonardis, W.G. Kropatsch, "A Highly Adaptable Concept for Visual Inspection", in: M. Jamshidi, J. Yuh, P. Dauchez, "*Intelligent Automation and Control - Recent Trends in Development and Applications*", Vol.4, pp.651-656, TSI Press, Albuquerque, 1996.

[SAN95] C. Sanby, L.N. Wayne, "Machine Vision Inspection of Lace Using a Neural Network", *Proc. of the IS&T/SPIE Symposium on Machine Vision Applications in Industrial Inspection III, San Jose, CA*, SPIE Vol.2423, pp.314-322, 1995.

[SHI87] Y. Shirai, "*Three Dimensional Computer Vision*", Springer, Berlin, Heidelberg, New York, 1987.

[SIL86] O. Silven, T. Westman, S. Huotari, H. Hakalahti, "A Defect Analysis Method for Visual Inspection", *Proc. of the 8th Intl. Conf. on Pattern Recognition, Paris*, pp.868-870, 1986.

[SIL89] O. Silven, I. Virtanen, T. Westman, T. Piironen, M. Pietikainen, "A Design Data-based Visual Inspection System for Printed Wiring", in: J. Sanz (Ed.), "*Advances in Machine Vision*", Springer Verlag, New York, pp.192-212, 1989.

[SMI93] B. Smith, "Making War on Defects: Six-Sigma Design", *IEEE Spectrum.*, Vol.30, No.9, pp.43-47, 1993.

[SPR91] A.P. Sprague, M.J. Donahue, S.I. Rokhlin, "A Method for Automatic Inspection of Printed Circuit Boards", *Comp. Vision, Graphics, Image Processing: Image Understanding*, Vol.54, No.3, pp.401-415, 1991.

[STE94] C.V. Stewart, "A New Robust Operator for Computer Vision: Theoretical Analysis", *Proc. of the IEEE Conf. Computer Vision and Pattern Recognition*, pp.1-8, June 1994.

[STR94] M. Stricker, "A New Approach for Robust Ellipse Fitting", *Proc. of the 3rd Intern. Conf. on Automation, Robotics and Computer Vision (Singapore)*, Vol.2, pp.940-945, Nov. 1994.

[STR95a] M. Stricker, A. Leonardis, "From Edgels to Parametric Curves", *Proc. of the 9th Scandinavian Conf. on Image Analysis, SCIA'95*, pp.749-758, Uppsala, Sweden, June 6-9, 1995.

[STR95b] M. Stricker, A. Leonardis, "ExSel++: A General Framework to Extract Parametric Models", *Proc. of the 6th Intl. Conf. on Computer Analysis of Images and Patterns, CAIP'95*, pp.90-97, Prague, Czech Republic, September, 1995.

[THO95] M. Thonnat, S. Moisan, "Knowledge-based Systems for Program Supervision", *Proc. of 1st Intl. Workshop on Knowledge-based Systems for the (Re)Use of Program Libraries*, Sophia Antipolis, France, Vol.1, pp.4-8, 1995.

[TOD88a] J.D. Todd, "Advanced Vision Systems for Computer-Integrated Manufacture -Part 1", *Computer Integrated Manufacturing Systems*, Vol.1, No.3, pp.143-154, 1988.

[TOD88b] J.D. Todd, "Advanced Vision Systems for Computer-Integrated Manufacture -Part 2", *Computer Integrated Manufacturing Systems*, Vol.1, No.4, pp.235-245, 1988.

[TOR92] C. Torras, "Segmentation", in: C. Torras (Ed.), "*Computer Vision: Theory and Industrial Applications"*, Springer Verlag, New York, pp.59-95, 1992.

[TUC93] M. Tuceryan, A.K. Jain, "Texture Analysis", in: C.H. Chen, L.F. Pau, P.S.P. Wang, (Eds.), "*The Handbook of Pattern Recognition and Computer Vision*", World Scientific, River Edge, NJ, 1993.

[VER91] D. Vernon, "*Machine Vision*", Prentice-Hall, Englewood Cliffs, New Jersey, 1991.

[VGO91] L. Van Gool, P. Wambacq, A. Oosterlinck, "Intelligent Robotic Vision Systems", in: S.G. Tzafestas, (Ed.), "*Intelligent Robotic Systems*", pp.457-507, Decker, New York, 1991.

[WAN90] R. Wang, H. Freeman, "The Use of Characteristic-view Classes for 3D Object Recognition", in: H. Freeman, (Ed.), "*Machine Vision for Three-Dimensional Scenes*", Academic Press, San Diego, 1990.

[WES96] P.C. West, "Survey of Training by Learning for Machine Vision", *Proc. of SME Conf. on Applied Machine Vision*, Cincinnati, Ohio, pp.489-493, 1996.

[XU93] L. Xu, E. Oja, "Randomized Hough Transform: Basic Mechanism, Algorithms, and Computational Complexities", *Comp. Vision, Graphics, Image Processing: Image Understanding*, Vol.57, No.2, pp.131-154, March 1993.

[YAN95] Q. Yang, "Automatic Fruit Blemish Detection", *Proc. of 9th Scandinavian Conf. on Image Analysis, Uppsala*, Vol.1, pp.365-372, 1995.

[YOU95] M. Young, D. Argiro, S. Kubica, "Cantata: Visual Programming Environment for the Khoros System", *Computer Graphics*, Vol.29, No.2, pp.22-24, 1995.

[YUE90] H.K. Yuen, J. Princen, J. Illingworth, J. Kittler, "Comparative Study of Hough Transform Methods for Circle Finding", *Image and Vision Computing*, Vol.8(1), pp.71-77, 1990.

[ZEL82] H. Zeller, G. Doemens, "Industrial Application of Pattern Recognition", *Proc. of the 6th Intl. Conf. on Pattern Recognition, Munich*, pp.202-213, 1982.