ARTIFICIAL INTELLIGENCE II: Methodology, Systems, Applications
Ph. Jorrand and V. Sgurev (editors)
Elsevier Science Publishers B.V. (North-Holland)

Towards Computer Aided Vision System Configuration

Bernd Neumann

Universität Hamburg
Fachbereich Informatik
Schlüterstr.70
D-2000 Hamburg 13
West Germany

1. Introduction

Computer vision is a discipline which has raised high expectations from the very beginning (cf. [13]). But there is evidence that profitable applications are lagging behind. Tracing computer vision surveys through the years (e.g. [10] and [11]), one repeatedly finds statements to the effect that the field has an extremely rich potential and widespread industrial applications are imminent, in particular in process control and inspection tasks. The predictions did not come true. In the automotive industry, for example, the vast majority of robot manipulators perform their tasks without visual sensors. Automatic visual inspection is the exception rather than the rule. Furthermore, a considerable fraction of the existing computer vision applications are pilot projects. The techniques developed in such projects are frequently not applied to similar tasks because of economic reasons. This picture is also supported by observations concerning the fate of laboratory developments "*for industrial applications*". The number of systems developed for practical use is out of proportion to the number of systems actually employed. This is particularly true for so-called universal vision-systems which have been designed for a broad spectrum of applications.

There are several reasons for these difficulties. The most basic reason is discussed in [22]: We lack practical methods. Simple, cheap and fast methods often perform poorly, while more complex and powerful methods tend to be slow and expensive. As we have already accumulated a large repertoire of methods and there is little hope for radically new developments, Winkler sees a way out of this dilemma only by a combined development of dedicated hard- and software. Both should move towards each other to bring about cheap, fast and powerful processing methods.

This paper is directed at another cause for the scarcity of industrial vision systems: the configuration problem. This problem arises when the task of a vision system changes and it is necessary to reconfigure the hard- and software accordingly. For many systems this tends to be a costly process where a vision system expert is required and a lengthy trial-and-error procedure must be carried out. Of course, the same configuration problem occurs when the system is installed for the first time. But the costs are often hidden as most installations are delivered as turn-key systems.

A similar problem arises from unwanted changes or variations in a vision system, e.g. changes of illumination due to light source contamination, or variations of object surface properties. Small changes of this kind may easily cause unacceptable effects, e.g. rejection of objects in an inspection task. In this case a solution in terms of a systematic adaption procedure is not satisfactory. We should be able to predict vision system behaviour precisely enough to avoid performance degradation right from the start. Hence we call this the prediction problem. Both, the configuration problem and the prediction problem, are related: An accurate performance prediction can obviously alleviate the configuration task.

This contribution deals mainly with the configuration problem but also touches upon the prediction problem. Both have a common cause: the lack of formal, theoretical penetration of the expert knowledge required for configuring a vision system for a given application.

Hence we can expect improvements if we succeed in developing formal and systematic procedures for those areas where human expertise, ad-hoc solutions and trial-and-error procedures predominate up to now.

In the following sections we discuss the expert system approach to configuring a vision system. Expert systems for configuration tasks have been developed successfully for several applications. The idea of applying this approach to vision system configuration is quite young, however, and there is not much work so far directly related to this problem (cf. [5], [7] and [14]). The scientific basis, however, is well enough developed to permit a detailled discussion of the problem.

Section 2 will deal with configuration systems in general. As we point out the essential properties and difficulties, we shall lay the ground for a specific discussion of the vision system configuration problem in section 3. In section 4, finally, we shall report about a concrete system which has been developed in the author's group (cf. [14]). This system supports method selection and configuration for visual inspection tasks.

2. Configuration Expert Systems

An expert system is a special kind of knowledge-based system designed to take the place of a human expert. The basic building blocks of an expert system are well-known. There is a knowledge base which contains expert knowledge about the task domain (typically in terms of rules) as well as dynamic descriptions of the specific task and of computed results. The second major component is the problem solver (often called inference engine). It is independent of specific tasks and works by applying rules to descriptions in the knowledge base until the desired goal has been achieved.

We are interested in expert systems for configuration tasks – in short: configuration systems – and shall now briefly review the kinds of problems which are typical for such systems. Configuration systems belong to the class of construction systems. Construction systems have to build up the result from components, as opposed to diagnosis systems where the result is selected from a set of possible results. For the case of a configuration system the result is a configuration composed of components in such a manner that the task specification and other constraints are met. The configuration system XCON/R1 is a well-known example (cf. [6]). It can configure DEC computers from given components and has been successfully used by DEC for several years.

Another successful development is the SICONFEX system (cf. [4]) which supports the configuration of Siemens SICOMP computers. As the architectures of such systems become better understood (the two examples are quite dissimilar), configuration expert system shells become feasible. One such specimen is presented in [20]. It is designed to meet the needs of a broad field of technical configuration tasks and could possibly be used as the basis for a vision system configuration system.

The problems which are typical for configuration system design can be roughly subdivided into two parts: knowledge representation problems and control problems. We shall discuss the former first. A fundamental prerequisite for using expert system technology at all is formalizability of the relevant knowledge. Formalization means that knowledge is structured and represented in a computer according to fixed rules. As we examine the knowledge governing vision system configuration, we can immediately see that formalization is an issue. Computer vision is a formal discipline, but vision system configuration is not yet, as indicated in the introduction.

The components of a configuration constitute an important class of objects which impose representational requirements. They have to be represented in terms of properties which bear on the configuration task. The computer system components of XCON, for example, are characterized by properties such as size, current load, connector type, etc. For configuration tasks such properties may often be expressed in terms of constraints. Hence finding a valid configuration may be posed as a constraint satisfaction task.

Experience has shown (cf. [8]) that multiple levels of representation may be useful. For example, a spatial configuration could be initially determined using simple, approximate object shapes. In a second phase, the proposed solution could be verified using exact object shapes. This can also be viewed as bottom-up hypothesis generation followed by top-down verification. The idea will also be useful for vision system configuration as shown in the next section.

Task specification is another representational issue. In the case of XCON the task is represented mainly by a list of components which have to be configured, hence there are no major representational problems. For vision system configuration, however, a formal task specification may be quite difficult. Consider, for example, the inspection task "*verify the proper position of the safety pin*" or "*determine surface integrity of a disc brake*". Apart of each individual representation problem, vision system task specifications are also problematic because of the multitude of possible tasks. Hence one must either provide a rather general set of representational tools or be prepared to narrow down the task domain.

As we focus on control issues now, it is important to note that constructing a configuration may not be possible without backtracking. The task can be compared to a puzzle where local decisions cannot be tested for global compatibility. Hence it may be necessary to undo decisions and explore alternatives. Problems of this nature have been investigated in depth for planning tasks (cf. [12] and [16]). Concepts and solutions developed hereby are partially applicable to configuration tasks as the latter constitute a more general class of tasks.

One of the concepts which do carry over is hierarchical planning. For configuration problems this means that a configuration is represented in multiple levels of abstraction. The highest level contains the coarsest view, lower levels refine the components of the next-higher level. If the configuration space is organized skilfully according to the principle of modularity, one obtains a tree with strong intra-branch and weak inter-branch dependencies. Following the tree during the configuration process will alleviate the global conflict problem but may not necessarily avoid it completely. XCON is an exception: The task could be structured in such a way that no backtracking is necessary at all.

If backtracking cannot be avoided, one can still try to improve search efficiency using 'intelligent backtracking'. This is the customary name for procedures which help to avoid undoing more decisions than necessary during backtracking. One way to do this is to keep track of all dependencies between decisions (cf. [1]). Thus one can determine which decisions are affected by a conflict and which not.

3. Vision System Configuration

We now turn to vision system configuration and discuss possible approaches as well as specific difficulties associated with such approaches.

In figure 1 we show the essential components of a vision system for an inspection task. In this example, the length of a work piece is to be examined. The following discussion will refer to this task as an exemplary one. Other tasks such as completeness tests, object recognition, fault detection, etc. pose similar problems.

The objects which are to be configured can be subdivided into *devices* (illumination, sensor, processor, test environment) and *methods* (thresholding, component analysis, etc.). Configuration (as an activity) encompasses selection, composition and adaption of such objects. The *selection* problem lies at the heart of the configuration problem. It amounts to choosing suitable devices and methods for a given task with the help of general configuration knowledge. The *composition* problem refers to structural decisions, i.e. to ways of putting a system together from a given set of components. In the case of devices composition mainly refers to the spatial arrangement, in the case of methods to temporal order and logical dependency. In general, the composition of devices offers many degrees of freedom, whereas methods impose a considerable amount of structure once they are selected. Hence method composition cannot really be separated from method selection. The *adaptation* problem is due to the fact that devices and especially methods typically contain free parameters. Examples are thesholding, edge detection with variable window size, minimal values for edge

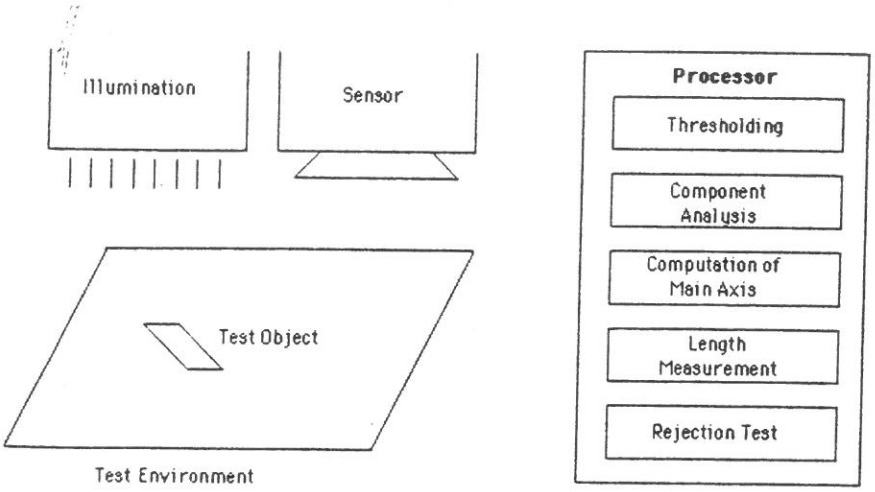

Figure 1: Components of a vision system for an inpection task

lengths or region sizes, light source power, etc. If the parameters are determined on the basis of the task specification, this part of the configuration process can also be considered as a selection (e.g. selection of a particular method instance). If there is a special evaluation criterion, however, which allows an iterative improvement of the configuration, parameter adaptation becomes a subtask in its own right. The evaluation of test runs could provide such a criterion, for example.

From now on we will refer to parameter adaptation in the latter sense. Furthermore, we will assume that selection and composition are combined in one process, hence a distinction will only be made between component selection and parameter adaptation. We now discuss formalization steps which are prerequisite for computer-aided component selection and for parameter adaptation, and examine to what extent this formalization is supported by the current state of the art.

Component selection

Computer-aided selection of devices and methods for a given task requires a formal representation of the following three components:

(i) task specification,

(ii) configuration knowledge,

(iii) devices and methods.

The *task specification* encompasses information about test object, test environment and test. This is the terminology of the VDI/VDE guidelines for automatic inspection (cf. [19]). The three components can be further expanded in considerable detail and provide a structured checklist for inspection task specifications. A similar approach could be used for other tasks.

For a configuration system it is useful to obtain a task specification by putting questions to the user. The kinds of questions asked (and the amount of information supplied by the user) greatly influence the resulting system complexity. In the simplest case, the task specification may be given in terms from which the configuration can be immediately obtained. For example, the user could be asked:

Can one use an array camera?

Is binary image processing possible?

Which features must be computed?

Such a task specification reduces the configuration system to a dialogue system where the user has to provide most of the expertise. A more interesting system results (cf. [14]) if the task is specified according to a task taxonomy as provided, for example, by [19] in a

crude form. Here the user indicates a task such as 'object recognition' and specifies the features, test object classes and tolerances. This information can be given without much image analysis expertise.

For more complex vision tasks and a more ambitious configuration system design it may be necessary to provide the properties of test objects and test locations in every detail, i.e. by specifying photometric surface properties, 3D shapes of objects and defects, etc. A task specification which is "complete" in this sense, is prerequisite for a *simulation* of the vision system. Simulation requires that the entire vision system be internally modelled by the configuration system. This approach has obvious advantages with respect to an exact prediction of the vision system performance. It is well supported by representation techniques developed in CAD and Computer Graphics (see [9] for a survey). In summary, one must say that little concrete work has been done on formal task specification, but the discipline is well organized and useful tools are available so that rapid advances seem possible.

The second problem component is *configuration knowledge*. By this we mean task independent expert knowledge which is employed for configuring vision systems. For example, we know that binary image processing may be possible if test object and test environmnent occupy non-overlapping sections of the greyvalue scale. We also know that the contrast between test object and environment is very large if backlighting is employed. These are examples of knowledge which can be represented in terms of productions or rules:

```
IF:   high contrast between test object and test environment is desired,
      and backlighting is possible,
THEN: select backlighting.
```

The ease with which such rules can be formulated is deceiving. All concepts and predicates used in the condition and action part of the rules must be carefully tuned with respect to each other to obtain a consistent and operational knowledge base. For example, one may wonder whether or not the consequence of selecting backlighting – high contrast – should be included in the action part of the rule. The freedom in designing such a knowledge base and the absence of a methodology are symptomatic for 'knowledge engineering', i.e. the process of transforming human expertise into a formal computer representation.

We now turn to the third problem component, the formal representation of *devices* and *methods*. They constitute the repertoire of the configuration system. Their representation must be tuned to the vocabulary used for task specification and configuration knowledge, as configuration rules establish a connection between the task on one hand and devices and methods on the other hand. As we pointed out earlier, the selection of a particular method typically implies strong restrictions on preceding and succeeding processing steps. Hence methods must be represented including interface conditions. The conditions – which correspond to the local constraints of a general configuration problem – mainly reflect compatibility requirements of data types.

Subroutine libraries which have been developed for interactive image processing systems are a useful basis for formalizing image processing methods. One example is the DIBIAS system (cf. [18]). In this system a composition of methods is achieved using a command language, but composition is also supported by information stored along with the subroutines. One may also start off with subroutine collections such as SPIDER (cf. [17]). SPIDER currently contains more than 400 FORTRAN subroutines. For a survey of interactive image processing systems see [3]. In summary, these developments are a useful basis for constructing a method base. Compatibility information with respect to data types may be readily obtained, whereas other important properties, in particular the suitability of a method for a given task, remains to be worked out.

Parameter adaptation

Parameter adaptation has been defined earlier as a configuration subtask where the vision system performance is improved iteratively by modifying free parameters. Parameter adaptation can be viewed as system fine-tuning and hence as complementary to compo-

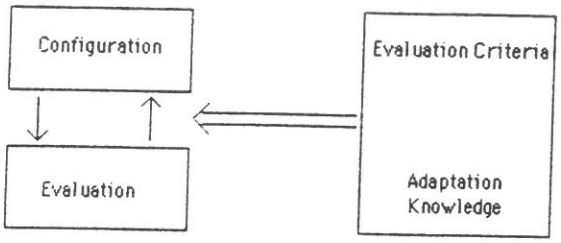

Figure 2: Parameter adaptation

nent selection. In addition, automatic parameter adaptation can be viewed as a method to compensate for unwanted changes as described in the introduction.

Parameter adaptation is a cyclic process with the structure shown in figure 2. A configuration is evaluated (downward arrow), then the parameters are modified based on this evaluation (upward arrow). The process is knowledge-based as it is controlled by evaluation criteria and adaptation knowledge.

An obvious evaluation method is trying out the configured vision system and evaluating the outcome. If test runs do not give the desired results, the parameters have to be modified using adaptation knowledge. We are interested in formalizing this process. In doing so, we have several options depending on the role which is to be played by the user on one hand and the concrete vision system on the other.

If one aims at parameter adaptation without human assistance, both evaluation and parameter modification must be formalized and all relevant knowledge must be formally represented. This method is called 'unsupervised automatic adaptation'. One can independently choose to carry out parameter adaptation in a simulated vision system. We then get 'adaptation by simulation'. Both methods can be combined and lead to adaptation systems with different merits, suitable for different applications.

The system developed by Liedtke and collaborators (cf. [5] and [2]) is an important example of unsupervised automatic adaptation using the real vision system. Their task is object recognition. As edge analysis and model matching is performed, the quality of the results is evaluated by demons and parameter modifications are suggested based on general adaptation knowledge.

The simulation approach is in its infancy. It is currently being investigated by several groups (cf. [15] and [21]), but concrete systems have not yet been presented. The advantages of simulation are evident: The performance of a vision system may be determined without the expense of installing a concrete system. Furthermore, simulation may be used for component selection as well as parameter adaptation.

4. BIKOS – A Configuration System For Visual Inspection

We now briefly describe the configuration system BIKOS which has been developed and implemented in the author's research group (cf. [14]). BIKOS configures vision systems for inspection tasks can only select methods. It is assumed that the physical components (devices) are fixed, and there is no parameter adaptation.

The performance of BIKOS is largely determined by a structured representation of image analysis methods. The structure is an AND/OR graph subdivided into levels of abstraction. An edge of the graph leading from a higher level to a lower level denotes either method refinement or method specialization. Figure 3 shows part of the method graph. Underlined methods are leaves of the graph and are called R-methods (realization methods). All other methods are abstract and are decomposed (not always shown in this this figure) into parts (Z-methods) or specializations (A-methods).

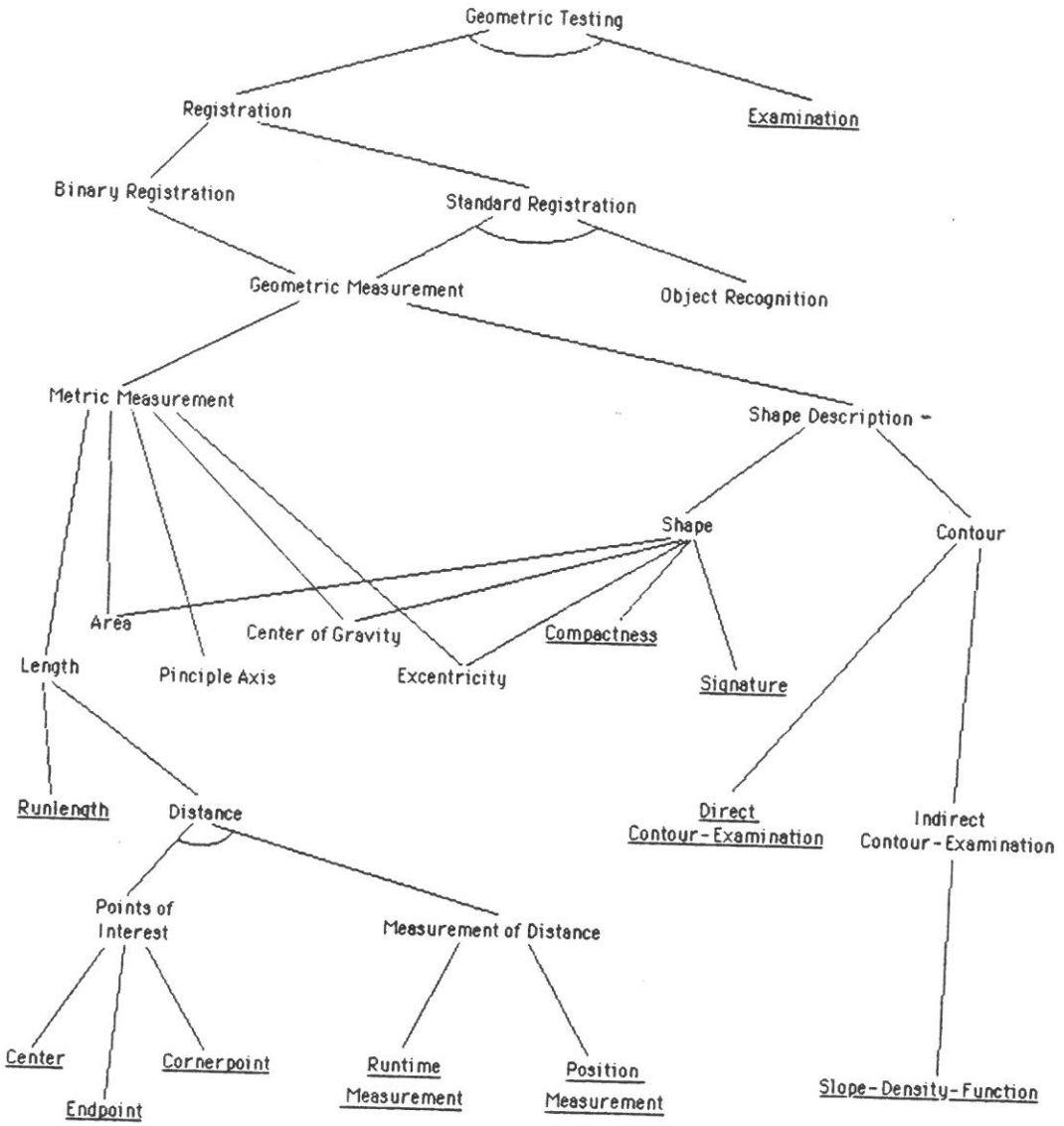

Figure 3: Part of the method graph

A method is a frame with the following structure:

METHOD

Nominal part:

[name]	method name
[class]	method class (Z, A, or R)
[optional]	optional method (yes or no)
[description]	short text describing the method
[parts]	list of partial methods for Z-methods
[specializations]	list of specializations for A-methods

Selection part:

[external conditions]	selection conditions referring to task
[internal conditions]	selection conditions referring to dependencies between methods

Action part:

[context activation]	information concerning context activation
[operators]	special functions called when activating the method

The selection part contains conditions which either refer to the task specification (e.g. 'little secondary-illumination') or to mutual constraints, in particular data type compatibility requirements. Facts which satisfy internal conditions are generated dynamically by method instantiation or instantiating configuration knowledge. The latter is represented in BIKOS in terms of criteria which serve to acquire, store or deduce knowledge about the task on hand. For example, there is a criterion 'secondary-illumination' which can satisfy the external condition of a method if instantiated accordingly.

CRITERION

[name]	secondary-illumination
[mode]	question
[text]	How strong is the secondary illumination?
[default]	average
[range]	little, average, strong

In this case the instantiation of the criterion leads up to a question. Other criteria deduce the desired information from existing data.

Questions are usually put forth in a task specification phase which precedes the configuration phase. The specification is carried out following the guidelines of [19]. Internally, this phase is controlled by an AND/OR task tree similar to the method tree.

The overall control of the configuration process is not predetermined by the knowledge structures presented so far but follows from meta-knowledge represented separately. The control knowledge of BIKOS currently realizes a depth-first search, i.e. the method graph is searched systematically beginning at its root until an acceptable configuration has been found. This is the order which takes the greatest advantage of the hierarchical structure given by the method graph, as most internal constraints are dealt with locally and backtracking can be kept within reasonable limits. Other search strategies can also be realized, however, e.g. following a user specified order.

BIKOS has been implemented in OPS5 on a VAX/780 and currently encompasses ca. 10.000 lines of source code.

5. Summary

Today's computer vision systems cannot be employed economically to such an extent as has been predicted. One of the reasons is the configuration problem: Installing a vision system and adapting it to a new task can only be done by experts and hence is costly. In this contribution we investigated the possibilities for using expert systems to configure computer vision systems. Configuration systems have been successfully developed for other applications, their properties and problems are sufficiently well understood. The configuration of vision systems can be subdivided into component selection and parameter adaptation. Both tasks require several kinds of formalized knowledge, part of which are already available from various sources. A first implementation of a configuration system for method selection has been presented which demonstrates the feasibility of this approach.

References

[1] de Kleer, J.: *An Assumption-Based TMS*, Artificial Intelligence 28/2 (1986), pp. 127-162

[2] Ender, M., Liedtke, C.-E.: *Repräsentation der relevanten Wissensinhalte in einem selbstadaptierenden regelbasierten Bilddeutungssystem*, to appear in: Hartmann, G. (ed.), Mustererkennung 86, Informatik Fachbericht, Springer 1986

[3] Haarslev, V.: *Interaktion in Systemen zur Bildfolgenauswertung basierend auf einem objektorientierten Ansatz*, Dissertation, Fachbereich Informatik, Universität Hamburg, 1986

[4] Haugeneder, H., Lehmann, E., Struß, P.: *Knowledge-Based Configuration of Operating Systems - Problems in Modeling the Domain Knowledge* in: W. Brauer, B. Radig (eds.), Wissensbasierte Systeme, Informatik Fachberichte 112, Springer 1985, pp. 121-134

[5] Liedtke, C.-E., Ender, M., Henser, M.: *Komponenten eines adaptiven Bildverarbeitungssystems zur Lageerkennung von Objekten*, in: H. Niemann (eds.), Mustererkennung 85, Informatik Fachberichte 107, Springer 1985, pp. 165-169

[6] McDermott, J.: *R1: A Rule-Based Configurer of Computer Systems*, Artificial Intelligence 19 (1982), pp. 39-88

[7] Neumann, B.: *Rechnergestützte Konfigurierung von Bildverarbeitungssystemen*, in: Proc. 1. Internationale Fachtagung "Automatische Bildverarbeitung", Kammer der Technik, DDR, 1985, P4/1- P4/4

[8] Raulefs, P.: *Knowledge Processing Expert Systems*, in: T. Bernold, G. Albers (eds.), Artificial Intelligence: Towards Practical Applications, North Holland 1985, pp. 21-32

[9] Requicha, A.A.G.: *Representations for Rigid Solids: Theory, Methods and Systems*, ACM Computing Surveys 12, 1980, pp. 437-464

[10] Rosen, C.A.: *Machine Vision and Robotics: Industrial Requirements*, in: G.G. Dodd, L. Rossol (eds.), Computer Vision and Sensor-Based Robots, Plenum Press, 1979, pp. 3-19

[11] Rossol, L.: *Computer Vision in Industry*, in: A. Pugh (ed.), Robot Vision, Springer 1983, pp. 11-18

[12] Sacerdoti, E.D.: *A Structure for Plans and Behavior*, American Elsevier Publ. Company, 1977

[13] Selfridge, O.G.: *Pattern Recognition and Modern Computers*, Western Joint Computer Conference 1955, pp. 91-93

[14] Syska, I.: *Ein Expertensystem-Ansatz für die automatische Konfigurierung von industriellen Bildverarbeitungsanlagen*, Diplomarbeit, Fachbereich Informatik, Universität Hamburg, 1986

[15] Strecker, H.: *private communication*, Philips Forschungslaboratorium, Hamburg, 1986

[16] Tate, A.: *A Review of Knowledge-Based Planning Techniques*, in: Proc. Expert Systems 85, M. Merry (eds.), Cambridge Univ. Press 1985, pp. 89-111

[17] Tamura, H., Sakane, S., Tomita, F., Yokoya, N., Kaneko, M., Sakaue, K.: *Design and Implementation of SPIDER - A Transportable Image Processing Software Package*, in: Computer Vision, Graphics, and Image Processing, Vol. 23, 1983, pp. 273-294

[18] Triendl, E., Fiedler, R., Helbig, H., Kritikos, G., Kübler, D., Lehner, M.: *Design of an Interactive Picture Processing System*. Proc. of the IEEE Computer Society Conference on Pattern Recognition and Image Processing, June 14-17, Las Vegas, Nevada 1982, pp. 534-536

[19] VDI/VDE-Handbuch Meßtechnik II: *Automatisierte Sichtprüfung - Beschreibung der Prüfaufgabe*. Beuth Verlag, 1985

[20] Vitins, M.: *A Prototype Expert System for Configuring Technical Systems*, in: Proc. Knowledge-Based Systems in Industry, I. Kriz (eds.), KLR 86-54C, Brown Boveri Research Center, Baden-Dättwil, Schweiz, 1986, pp. 146-161

[21] Weber, J.: *private communication*. URW GmbH, Hamburg, 1986

[22] Winkler, G.: *Industrielle Anwendung der digitalen Bildauswertung*, in: Informatik-Spektrum 8/4, Springer 1985, pp. 215-224