# Automatic Configuration of Medium-level Vision Routines Using Domain Knowledge

Tilo Messer
Bayerisches Forschungszentrum für Wissensbasierte Systeme
Orleansstr. 34, W-8000 München 80, Germany

Michael Schubert
Institut für Informatik der Technischen Universität München
Orleansstr. 34, W-8000 München 80, Germany

## Abstract

In this paper we present a knowledge-based approach for recognizing objects in a natural scene. The system generates a sequence of vision operations with respect to a query about a particular object. The resulting computer vision program detects the object in a grayscale image. We propose a three-stage procedure. First, we use a 2-D representation of real objects modelling them in an *object knowledge base*. A query to the object knowledge base about an object procures *evidence features* of the goal object. The second stage involves a set of the deduced evidence features and their values forming a *generic model* of the real object. The generic model serves as a specification for a vision program. Finally, a *configuration module* comprising of low-level and medium-level vision operations, segmentation algorithms, and morphological operations, is used to generate a vision program. The configuration process is supported by various rules. These rules are used to set appropriate parameter values for the vision operations and to control how vision operation are combined to form a program.

## Résumé

Nous présentons dans cet article une approche de la reconnaissance d'objets dans un environnement naturel à l'aide d'une base de connaissances. Un système qui génère une séquence d'opérations de vision grâce à des requêtes sur un objet a été développé. Le programme de vision par ordinateur final détecte les objets dans une image noir et blanc. Nous proposons un processus en 3 temps: nous utilisons d'abord une représentation en 2 dimensions d'objets réels, en les modélisant dans une *base de connaissances objet*. Une requête sur un objet, faite à la *base de connaissances objet*, déduit des *caractéristiques évidentes* (evidence features) à propos de l'objet recherché. Dans un deuxième temps, l'ensemble des *caractéristiques évidentes* ainsi déduites, et leurs valeurs forment un *modèle générique* de l'objet réel. Le *modèle générique* va servir en tant que spécification pour le programme de vision. Enfin, un *module de configuration* qui met en œuvre des opérations de vision de bas et moyen niveau, ainsi que des algorithmes de segmentation et des opérations morphologiques, génère un programme de vision. Des règles pour déterminer les paramètres adéquats pour chaque opération, et d'autres contrôlant l'assemblage de la séquence des opérations de vision assistent le processus de configuration.

**Keywords:** Configuration, Domain Knowledge, Evidence Features, Expert Systems, Generic Model, Image Understanding

## Introduction

Combining computer vision and expert system techniques leads to improvements in both automatic image interpretation and automatic vision program configuration. Automatic configuration implies the arranging of sequences of vision operations by means of rules which are activated by knowledge about various domains. Above all two domains have to be emphasized: the computer vision domain and the application domain. Research results concerning automatic vision system configuration contain, on one hand, interactive user consultation systems for developing vision programs and, on the other, automatic adaption of vision operations to new scene contents ([EnL 86, LiE 86]).

During the last 10 years, efforts have been made trying to create an automatic process for arranging vision programs. Sueda ([Sue 85], [Mat 86]) developed a consultation system for image processing. In his system, a step in the automatic vision program configuration is taken once appropriate commands have been automatically selected from a large number of available commands and appropriate arguments for these commands have been specified. With this problem, an user of an interactive image processing system is also confronted. The consultation system uses help facilities and restricted configuration knowledge for guiding the user to the appropriate choice. A further step towards automatic configuration has been taken by Sakane.

Nadif and Matsuyama: Sakane and Tamura ([SaT 85]) propose an automatic vision program generation system. They define an intermediate language to express a command sequence which is translated into SPIDER code ([TST 83]). Nadif ([NaL 84]) and Levine describe a rule-based segmentation system to create effective segmentation algorithms. Production rules, regarding the splitting and merging of regions and lines, serve as a basis for the inference mechanism. Matsuyama ([Mat 86]) uses a network arrangement to represent image-features and transfer-processes as parts of the knowledge needed for a segmentation process. The specification for the automatic image segmentation expert is given by the user. Tomita ([Tom 88]) combines vision program modules with machine learning techniques. The user can teach the system the way of recognizing objects. The system uses the built-up model to analyse images automatically and to recognize the expected objects. Vogt ([Vog 88]) also classifies objects by training sequences of morphological algorithms. Joo and Haralick ([JoH 89]) illustrate the ability of mathematical morphology to extract shape information from grayscale and binary images. They point out that the problem they want to solve is to reduce any machine vision task to a sequence of morphological operations. They propose four knowledge sources which influence the reduction: description of the input image, goal, morphological operations, and knowledge and reasoning. Syska and Neumann ([Sys 86], [Neu 87]) construct an image recognition system by combining the design, scope, and organization of the images component parts. The work of Risk and Boerner ([RiB 89]) also deals with the configuration of a complete vision system (X-ray inspection of cast metal parts). They argue that the result of the configuration process does not have to be revised. In case of wrong result, computer vision experts are asked to bring in their knowledge and experience to modify the knowledge base.

We have implemented a system called *BOA* for automatic configuration of vision programs. *BOA* is divided into three modules which consist of: the *object knowledge base*, the *generic model module*, and the *configuration module*. Extending the concept of existing systems our approach generates vision programs using a specification which results from a query to a knowledge base about a goal object. Examples of queries which may be put to the object knowledge base are: "Is an object x in the scene?", "Where is the object x?". The query can only refer to objects which are modelled in the object knowledge base. The query results in generic model features of the object. A set of pairs of generic model features with their values is a specification of the goal object. This specification is then used as the generic model of the object. Some of the specified features and values may be similar to those of other objects not asked for. In order to cope with the problem, a mechanism for refining the specification is provided. Finally, the specification is used in controlling the configuration of the vision program. Running this vision program on the input image results in instances of the goal object (Fig. 1). Instances are regions which are computed by the sequence of the morphological and mathematical set operations. Identification faults can be detected by matching the resulting instance with the previous generic model features. The refinement mechanism mentioned above can be used to get an instance which fits the generic model better.

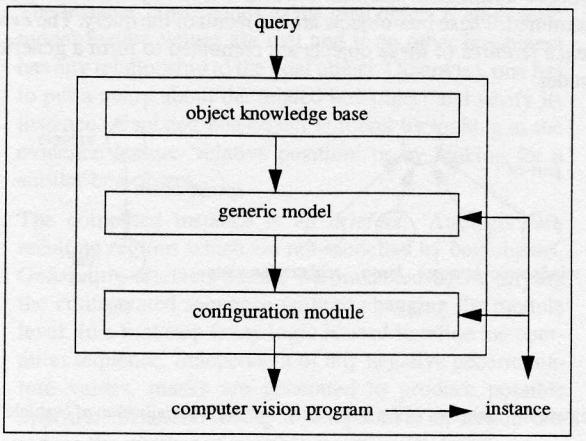



**Figure 1:** Architecture of the *BOA* System

The objective of our approach is to design a knowledge base, with appropriate facilities, to make the relationship clear and predictable which exists between the specification of the vision program and the result of its application to the input image.

The paper is structured as follows. First of all, the structure of the object knowledge base will be explained. Thereafter, a brief survey of the computed generic model features is given. The way these features are used to control the configuration process of the vision program is then described. Finally, a number of pictures illustrate some intermediate results.

## Object Knowledge Base

In order to deduce generic model features, an *object knowledge base* is built up. 3-D objects of the real world are modelled as 2-D objects in the object knowledge base. The transformation from 3-D to 2-D is done manually. Due to our special application domain we have to generate only very few 2-D objects for each real object[1]. The object knowledge base is built up by hierarchical taxonomy and a 'part-of' hierarchy. The objects in the object knowledge base are called *boa-objects*. Boa-objects can have boa-objects as their parts forming the part-of hierarchy (Fig. 2a). A boa-object can also be a specialization of another boa-object (Fig. 2b). A boa-object is essentially described as a set of evidence features.

Beginning with the boa-object which is the subject of the query, the control algorithm implemented in the object knowledge

1. We use traffic scenes for our experiments. The images of the scene are taken by a camera which is fastened in the front of a car and which is directed straight forward.

base runs through all boa-objects which are connected with the starting boa-object by specialization and 'part-of' relations. The process terminates when all reachable boa-objects have been examined. These boa-objects are the result of the query. The *evidence features* of these objects are combined to form a generic model.

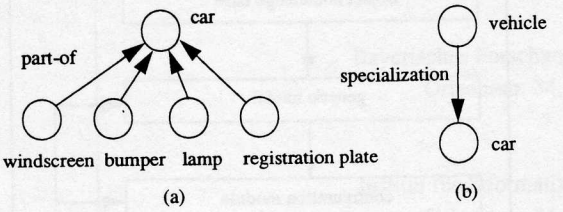



**Figure 2:** (a) parts of the boa-object 'car'; (b) specialization of 'vehicle'

Evidence features have already been used by Hoffman and Jain ([HoJ 88]) in order to manage the difficulties of representing objects and object parts in a fully automatic vision system[2]. Given two areas P and Q, examples of evidence features used by Hoffman and Jain would be:

- the diameter of P,
- the surface area of P, and
- the maximum distance between P and Q.

Examples of evidence features and possible values used in *BOA* are given below:

| *evidence feature* | *possible values or value type* |
|---|---|
| • shape: | line, rectangle, circle, triangle |
| • width: | number of pixels |
| • height: | number of pixels |
| • ratio height/width: | scalar |
| • radius: | number of pixels |
| • angle of the longer principal axis to baseline: | degrees |
| • grayscale: | value between 0 and 255 |
| • absolute reflectance: | high, medium, low |
| • relative reflectance: | lighter, darker, opposite |
| • absolute contrast: | high, medium, low |
| • relative position: | above, below, right-of, left-of, inside-of, single |
| • view: | frontal, side |

A table of boa-objects with their evidence features is built up. The entries of the table are, on one side, boa-objects resulting from a query to the object knowledge base about a real object, and, on the other side, the values of the evidence features of these boa-objects. An example of such a table is given in Fig.3.

| evidence feature \ boa-object | registration plate | bumper |
|---|---|---|
| shape | rectangle | rectangle |
| height | 9 | —— |
| width | 27 | —— |
| ratio h/w | 1 / 3 | 1 / 20 |
| grayscale | 210 | 185 |
| view | frontal | frontal |

**Figure 3:** Boa-objects and their evidence features.

Each evidence feature of a boa-object has no more than one value. If a query results in more than one boa-object it is obvious that there could be more than one entry for an evidence feature in the resulting table. In this case, the representation of boa-objects and their evidence features enables us to build up relationships between various boa-objects. The relationships are built by examining the values of the evidence features of the boa-objects. Various boa-objects can be connected if the values of some evidence features meet certain conditions. This connection leads to two advantages, the first is one concerning the configuration process and the second is to do with the interpretation of the computed instance of the generic model. The former advantage lies in the ability to use the condition *similar* in the configuration process. Such a relationship allows a much more detailed specification of the configuration process. The latter advantage lies in making it possible for a verifying mechanism to recognize possible identification faults. The configurated vision program could produce a resulting instance which cannot be identified as an instance of the goal object. Exploiting the relationship between various boa-objects can make it possible to avoid identification faults. Thus we now want a proper definition for the terms *similar* and *non-similar*.

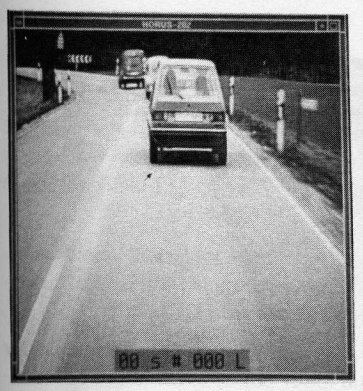
Two boa-objects are *similar* if they have at least one evidence feature in common with equal values. For example, the boa-objects 'registration plate' and 'bumper' are similar because they have the same shape (evidence feature 'shape' has the value 'rectangle') (Fig. 3).[3]

Formal definition:

Let $O_1$ and $O_2$ be two boa-objects, $\{e_1^1, \ldots, e_n^1\}$ and

2. Hoffman and Jain classify 3-D objects by a learning approach using evidence features.

3. If the table of boa-objects and evidence features is treated as a table of a relational database, the procedure of getting similar objects could be interpreted as a join operation.



original grayscale image
512 x 512 pixels
8 bits grayscale

**query**: where is the registration plate?

**object knowledge base**

boa-objects inspected by the control algorithm

resulting boa-objects

| | registration plate |
|---|---|
| shape | rectangle |
| height | 9 |
| width | 27 |
| ratio h/w | 1 / 3 |
| grayscale | 210 |
| view | frontal |

**generic model**

**configuration module**

composition rule:

if the *value of the generic model feature width exists*
then the *feature width is a lower bound of the height for a opening operation with a complement operation.*
and *connect this opening operation*

composition meta rule:

if *two or more positive feature vectors exist*
then *select one positive feature vector*
and *create a new branch in the configurated sequence.*

**sequence of vision operations**
the patterned circles are intermediate results by applying a

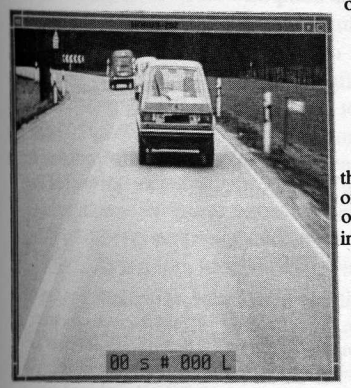
mask operation

opening

complement

result

original image



the result of the application of the configured vision operations sequence to the input image (black)

Figure 4: a process activated by a query to *BOA*

*BOA* produces an instance: When may *BOA* come to the conclusion that the computed instance corresponds to the goal object?

1. The instance corresponds to the goal object if all generic model feature values are met and if no other boa-object has any relationship to the goal object. Otherwise, one has to put a query about the related boa-object and verify its instance. A related boa-object is found by looking at the evidence feature 'relative position' or by looking for a similar boa-object.
2. The computed instance is an *artefact*. 'Artefacts' are resulting regions which are not modelled by boa-objects. Generating artefacts should be prevented by modifying the configurated sequence without changing the module level. In a first step fuzzy logic is used to refine the operation sequence. Independent of any negative generic feature values, masks are generated to produce possible opposite instances. Using a complement operation can reduce the number of possible resulting artefacts[5].

BOA does not produce any instance.

3. The sequence of vision operations has to be modified. We follow the strategy that the masks used in the configurated sequence should be modified. Either the mask of the closing operations are too restricted or the masks of the opening operations connected with an complement operation are too large.

## Experimental Results

The *BOA* system is implemented using NEXPERT Object for the representation of the object knowledge base and the processing of the generic feature models. The configuration module is implemented in IF/Prolog. The vision operations are taken from the image processing tool HORUS ([Eck 88]). The images of the traffic scenes which are the basis of the experiments were put at our disposal by BMW[6]. The image size is $512 \times 512$ pixels and the gray level is 8 bits.

As an example, intermediate results during the configuration process for detecting and identifying a registration plate are demonstrated. Fig. 5 shows the original input image which is preprocessed and segmented, as shown in Fig. 6.

A closing operation fills up small holes in the binary image (a mask of $3 \times 9$ pixels is used) (Fig. 7). The size of the mask is orientated by the evidence feature 'ratio of width and length' of the boa-object 'registration plate'.

5. If it is assumed that a value of a generic model feature has the value 10 taken from a range of possible widths from 0 to 100, then the complementary value for this range of possible values is defined as 90. Assuming the possible range remains constant, it is obvious that, e.g., the value 55 cannot have a significant complementary value.

6. We thank BMW AG, Munich, making the images available to us. The images are parts of some image sequences.

$\{e_1^2, \ldots, e_m^2\}$ the evidence features of $O_1$ and $O_2$ respectively, $v(e)$ the value of an evidence feature $e$.

$O_1$ and $O_2$ are *similar* if

$\exists e_i \in \{e_1^1, \ldots, e_n^1\} \cap \{e_1^2, \ldots, e_m^2\} : v(e_i^1) = v(e_i^2)$

Two boa-objects are *non-similar* if they have no evidence feature in common or, if each common evidence feature has a different value for the two boa-objects.

Formal definition:

$O_1$ and $O_2$ are *non-similar* if

$\{e_1^1, \ldots, e_n^1\} \cap \{e_1^2, \ldots, e_m^2\} = \emptyset$ or if

$\forall e_i \in \{e_1^1, \ldots, e_n^1\} \cap \{e_1^2, \ldots, e_m^2\} : v(e_i^1) \neq v(e_i^2)$.

## Generic Model

The *generic model* of a query consists of all evidence feature values of the deduced boa-objects. We call them *generic model features*. The generic model features describe a prototype of the goal object. They are composed of *generic feature vectors* referring to each boa-object. This way of structuring fits well with the composition rules of the configuration module. The rules are activated by the values of the generic model features. The order of composition of the vision operations is also influenced by the number of generic feature vectors.

Two categories of generic feature vectors exist: *positive generic feature vectors* and *negative generic feature vectors*. Positive generic feature vectors contain the values of the generic model features which contribute information about objects to be detected and identified to the configuration process. Negative generic feature vectors contain generic model feature values describing boa-objects which are *not intended* to be detected and identified. Negative generic feature vectors are used to control the configuration process in order to prevent detecting and identifying objects, esp. those which are similar boa-objects.

## Configuration Module

The configuration module contains a collection of vision operations. For our purpose morphological operations and mathematical set operations are predominantly used. Examples of morphological operations are:

- dilation, erosion
- opening, closing
- morphological sieve operation
- operations using letters of the Golay alphabet
- operations producing a skeleton of a region

Examples of mathematical set operations are:

- intersection, union, complement

Low-level vision operations and segmentation operations are applied to the grayscale input image in order to get a binary image. We do not concentrate on this part of the configuration module because it is not the topic of this paper.

Morphological operations for binary images are configurated. Composition rules and composition meta rules control the configuration. An example of a composition rule is:

- If *the value of the generic model feature 'width' exists*
  then *this value is a lower bound of the height of a mask for an opening operation*
  and *connect the opening operation with an complement operation.*

Knowledge about the appropriate application of the morphological vision operations is expressed by the composition rules. Due to the fact that the parameters concerning masks are almost the only once which can be set[4] to control the configuration process, a lot of rules exist concerning mask creation. The values of the generic model features are the principal values used to control the specification.

Composition meta rules represent knowledge about how to arrange operations or the order of sequences of operations. If it is necessary to consider values of a generic model feature which compete with each other during the configuration process, operation sequences are run concurrently and the instances are combined resulting by their application to the image with an union operation. Negative generic feature vectors activate composition meta rules concerning the complement operation. These types of rules use the generic feature values as knowledge sources. An example of a composition meta rule is:

- If *two or more positive generic feature vectors exist*
  then *select one positive generic feature vector*
  and *create a new branch in the configured sequence.*

After the configuration process is finished an *instance of the generic model* is computed. Fig. 4 shows the whole process in simplified, schematic form.

To make sure whether the computed result is an instance of the generic model, a matching process compares the generic model feature values with the corresponding values of the resulting instance. If the values fit, the matching process is successful. Even in this case it is not conclusive that the computed instance corresponds to the instance of the goal object. Why? There are three problems which have to be solved when the process from the query to the instance comes full circle:

4. Another variable parameter is the number of iterations done in a morphological operation. BOA does not yet take into consideration morphological operations which use letters of the golay alphabet as structuring elements. Input and output images are not treated as parameters which can be used for the knowledge-based configuration process.

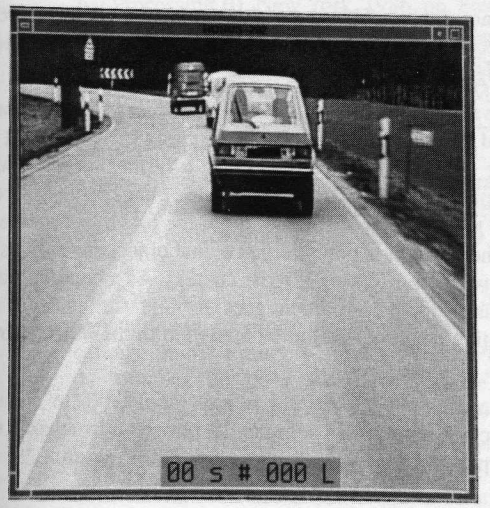



Figure 5: Original grayscale image: the given task is to detect the registration plate (marked black).

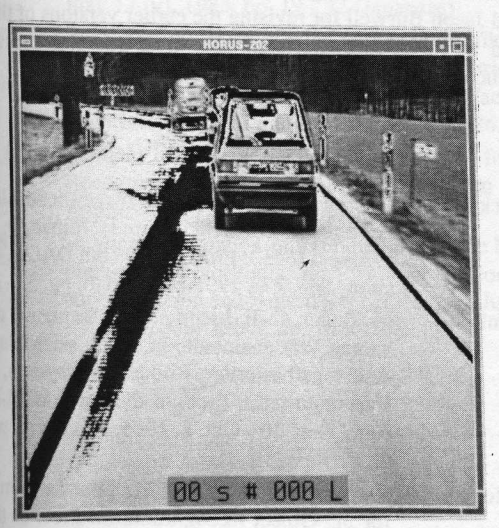



Figure 6: Step 1: segmented by a threshold operation

The following step removes long vertical regions by running an opening operation using a mask measuring $20 \times 1$ pixels. The width of the registration plate is a lower bound for the height of the mask. As a result, long vertical regions (Fig. 8) are removed from the image Fig. 7 by a complement operation (Fig. 9).

Finally an opening operation selects those regions which are larger than the appropriate mask ($9 \times 27$ mask) (Fig. 5).

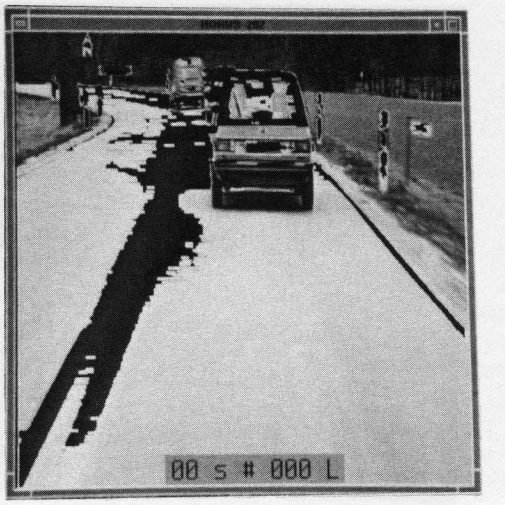



Figure 7: closing operation with mask $3 \times 9$ in order to fill up holes

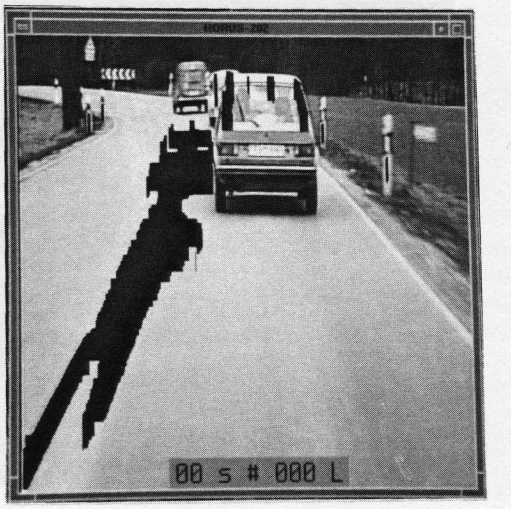



Figure 8: opening operation with mask $20 \times 1$.

If the goal object is a 'bumper' the following result demonstrates the effect of instances produced by other boa-objects (e.g. 'registration plate') or non boa-objects (artefacts). The structure of the operation sequence corresponds to that for detecting the registration plate but of course uses different masks (Fig. 10).

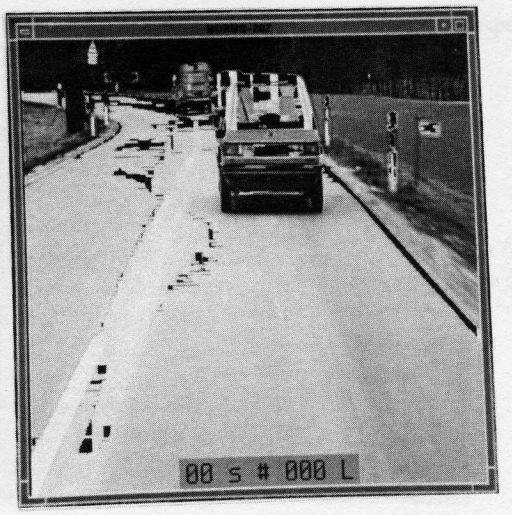



**Figure 9:** complement of the closing and opening operations.


**Figure 10:** Resulting instances including artefacts and other boa-objects than the goal object (here: bumper).

## Conclusion

In this paper, we have discussed a knowledge-based approach for detecting and identifying objects in a grayscale image by generating a sequence of vision operations and applying them to the image. Our approach is restricted to scenes including objects which have simple shape and topological relations. This is due to the simple segmentation operation and the use of morphological algorithms. This restriction, however, could be prevented by including other vision operations which are considered in the configuration process.

If we applied the system to image sequences identification faults could be reduced. Assuming, a resulting instance is not unique to be identified as the goal object, a further instance, which is computed in a following image of the image sequence, could be used to clarify the identification of the previous instance.
Furthermore, it is possible that a goal object can only be identified when considering it in its context. As a consequence, contextual knowledge which goes beyond the evidence feature 'relative position' should be modelled in the object knowledge base.

In conclusion, the system is well applicable to scenes with objects of a restricted complexity, but could be adapted to more complex scenes by modifying parts of the system.

## Acknowledgements

We gratefully acknowledge the constructive and critical comments of Bernd Radig. Many thanks also go to Elisabeth Ober and Jason Burwell for revising the earlier versions of the paper and to Jean-Marc Steffann who translated the abstract into french.

## References

[Eck 88] Wolfgang Eckstein: Das ganzheitliche Bildverarbeitungssystem HORUS, in: H. Bunke, O. Kübler, P. Stucki (eds.): Proc. of the 10th DAGM-Symposium, Sep. 27 - 29, 1988, Zürich, pp. 53 - 59.

[EnL 86] M. Ender, C.-E. Liedtke: Repräsentation der relevanten Wissensinhalte in einem selbstadaptierenden regelbasierten Bilddeutungssystem, in: G. Hartmann (ed.): Proc. of the 8th DAGM-Symposium, Sep. 30 - Oct. 2, 1986, Paderborn, pp. 219 - 223.

[HoJ 88] Richard Hoffman, Anil K. Jain: Learning Rules for 3D Object Recognition, in: Proc. of the Conference on Vision and Pattern Recognition 88, June 5 - 9, 1988, Ann Arbor, pp. 885 - 892.

[JoH 89] Hyonam Joo, Robert M. Haralick: Applications of Mathematical Morphology to Machine Vision, in: H. Burkhardt, K.H. Hoehne, B. Neumann (eds.): Proc. of the 11th DAGM-Symposium, Oct. 2 - 4, 1989, Hamburg, pp. 1 - 27.

[LiE 86] C.-E. Liedtke, M. Ender: A Knowledge-based Vision System for the Automated Adaption to New Scene Contents, in: Proc. of the 8th International Conference on Pattern Recognition 86, Oct. 27 - 31. 1986, Paris, pp. 795 - 797.

[Mat 86] T. Matsuyama: Expert Systems for Image Processing: An Overview, in: I.T. Young et al. (eds.), Sig-

nal Processing III, New York, 1986, pp. 869 - 872.

[NaL 84] A.M. Nadif, M.D. Levine: Low Level Image Segmentation: An Expert System, in: IEEE Transactions on Pattern Analysis and Machine Intelligence PAMI-6 (5), 1984, pp. 555 - 557.

[Neu 87] Bernd Neumann: Towards Computer Aided Vision System Configuration, in: Ph. Jorrand, V. Sgurev (eds.): Artificial Intelligence II: Methodology, Systems, Applications, Sep. 16 - 19, 1986, Varna, 1987, pp. 385 - 393.

[RiB 89] I.C. Risk, H. Börner: VIDIMUS: A Vision System Development Environment for Industrial Applications, in: W. Brauer, C. Freksa (eds.): Proc. of the Kongreß für Wissensbasierte Systeme 1989, Oct. 16 - 17, 1989, München, pp. 477 - 486.

[SaT 85] K. Sakaue, H. Tamura: Automatic Generation of Image Processing Programs by Knowledge-Based Verification, Proc. of the Conference on Computer Vision and Pattern Recognition, June 19 - 23, 1985, San Fransisco, pp. 189 - 192.

[Sue 85] N. Sueda: An Expert System for Image Processing, in: Image Technology and Information Display 17 (9), 1985, pp. 19 - 22 (Japanese).

[Sys 86] Ingo Syska: Ein Expertensystemansatz für die automatische Konfigurierung von industriellen Bildverarbeitungsanlagen, Diplomarbeit, Fachbereich Informatik, Universität Hamburg, 1986.

[Tom 88] Fumiaki Tomita: Interactive and Automatic Image Recognition System, in: Machine Vision and Applications 1, 1988, pp. 59 - 69.

[Vog 88] R. Vogt: Automatic Configuration of Simple Morphological Algorithms, in: Proc. of the Conference on Vision and Pattern Recognition 88, June 5 - 9, 1988, Ann Arbor, pp. 760 - 765.