Knowledge Based Image Understanding by Iterative Optimization

H. Niemann[a,b], V. Fischer[a,c], D. Paulus[a], J. Fischer[a]
[a]Lehrstuhl für Mustererkennung (Informatik 5)
Universität Erlangen-Nürnberg
Martensstraße 3, 91058 Erlangen, Germany;
[b]Forschungsgruppe Wissensverarbeitung
Bayerisches Forschungszentrum für Wissensbasierte Systeme
Am Weichselgarten 7, 91058 Erlangen, Germany
[c]IBM Deutschland GmbH, Institut für Logik und Linguistik,
Vangerowstraße 18, 69115 Heidelberg, Germany

This paper will be printed in the Proceedings KI'96
September 1996, Dresden, Germany

Knowledge Based Image Understanding by Iterative Optimization

H. Niemann[a,b], V. Fischer[a,c], D. Paulus[a], J. Fischer[a]
[a]Lehrstuhl für Mustererkennung (Informatik 5)
Universität Erlangen-Nürnberg
Martensstraße 3, 91058 Erlangen, Germany;
[b]Forschungsgruppe Wissensverarbeitung
Bayerisches Forschungszentrum für Wissensbasierte Systeme
Am Weichselgarten 7, 91058 Erlangen, Germany
[c]IBM Deutschland GmbH, Institut für Logik und Linguistik,
Vangerowstraße 18, 69115 Heidelberg, Germany

Abstract In this paper knowledge based image interpretation is formulated and solved as an optimization problem which takes into account the observed image data, the available task specific knowledge, and the requirements of an application. Knowledge is represented by a semantic network consisting of concepts (nodes) and links (edges). Concepts are further defined by attributes, relations, and a judgment function. The interface between the symbolic knowledge base and the results of image (or signal) processing and initial segmentation is specified via primitive concepts.

We present a recently developed approach to optimal interpretation that is based on the automatic conversion of the concept oriented semantic network to an attribute centered representation and the use of iterative optimization procedures, like e.g. simulated annealing or genetic algorithms. We show that this is a feasible approach which provides 'any–time' capability and allows parallel processing. It provides a well–defined combination of signal and symbol oriented processing by optimizing a heuristic judgment function.

The general ideas have been applied to various problems of image and speech understanding. As an example we describe the recognition of streets from TV image sequences to demonstrate the efficiency of iterative optimization.

Key Words: semantic network, iterative optimization, knowledge based image analysis

1 Introduction

The general goal of automatic image interpretation is to extract from an image or image sequence the information relevant to perform a well–defined task in an *optimal* manner. It is not necessary to extract *all* information contained in the images but to provide the *relevant* information in a format which suites the subsequent usage, either by man or by machine. Usually, information is required in some symbolic and condensed form, not as arrays or subarrays of pixels.

[1] This work was funded partially by the German Research Foundation (DFG) under grant number SFB 182. Only the authors are responsible for the contents.

[2] The related work on *speech* understanding using iterative optimization was supported by the Real World Computing Partnership (RWCP).

Interpretation of sensory inputs can only be done with respect to an internal model of the environment, where in principle the observations may in turn modify the internal model. Hence, all interpretation is model–based or knowledge–based.

In this paper we present an approach to represent a model of the task domain or the a priori knowledge about it in a semantic network and to compute an interpretation which is optimal with respect to a given judgment function. From among the different approaches to optimization we describe combinatorial optimization techniques. We point out how knowledge based processing interfaces to data driven segmentation and preprocessing, but techniques for this are not the topic. Finally, we demonstrate the feasibility and efficiency of our approach by application.

Related work on applications of image interpretation is covered, for example, in [1, 2], some general books on image interpretation are, for example, [3, 4], and some books on knowledge representation and use are, for example, [5, 6].

2 Overview

The general goal of image understanding is the computation of a symbolic description $\mathcal{B}$ of an image $\boldsymbol{f}$ or image sequence $\boldsymbol{f}_\tau$ which

- optimally fits to the observed data (or pixels),
- is maximally consistent with internally represented task–specific knowledge,
- and best suits the demands of a user as defined by an explicit model of the task–domain.

Hence, it is an *optimization problem* and should be formulated (see Section 3) and solved (see Section 4) as such.

Two main phases of processing are assumed: an initial phase of mainly data–driven processing and a phase of mainly model–driven processing. However, it is not assumed in general that these two phases are strictly sequential in the sense that the first phase must be finished before the second may start. Rather it is assumed that the timing of the phases, the data used by them, and the order of switching between phases is determined by a control strategy implemented by a control module or control algorithm.

The data–driven phase of processing consists of *preprocessing* and *initial segmentation*. The goal of preprocessing is to improve the image quality in order to obtain better results during subsequent processing. In general it transforms an image $\boldsymbol{f}$ into another image $\boldsymbol{h}$. The goal of initial segmentation is to decompose the image into *segmentation objects* O like lines or regions and to obtain their attributes and relationships. We assume that no explicitly represented task–specific knowledge is used, but only general knowledge valid for (almost) all images, for example, about color, geometry, or image formation.

The main topic of this paper is an approach to knowledge representation in the concepts C of a semantic network as well as its use for image understanding, and the formulation of image understanding as the computation of a best scored instance $I^*(C_g)$ of a goal concept C_g.

3 Framework for Knowledge Based Processing

Initial Description According to the above processing model, data driven image processing and initial segmentation is viewed as a sequence of operations, that transforms an image $\boldsymbol{f}$ into an *initial description* $\mathcal{A}$ which in general is a network $\langle O \rangle$ of *segmentation objects* O. In our applications we mainly use straight and circular lines for segmentation. The initial (symbolic) description defines the *interface* to knowledge based processing.

In general, a segmentation object has a *type* T_O, for example, indicating that it is a straight line, or a circle and it has a *name* identifying the object, for example, the number of a straight line. In addition, it may have attributes, parts, structural relations among its attributes and parts, and it should have a judgment $\mathbf{G}$ which in general is a vector of real numbers. An *attribute* A has a type T_A, a value which may be a real number or a symbol from a finite terminal alphabet, and a judgment which is a real number. Examples of attributes are the length of a line, the angle between a line and the x-axis, or the area of a region. A *part* P of a segmentation object is itself a segmentation object and provides a decomposition into simpler components. For example, the 'right angle' has as parts two lines. A *structural relation* S expresses a constraint between attributes on objects (A_O), parts (A_P), and concretes (A_K, see below), of a segmentation object, for example, in a right angle the angles between each of the two lines and the x-axis are constrained. It is useful to consider fuzzy relations having an associated real number which measures the degree of fulfilment of the relation. Briefly, a segmentation object is a structure

$$
\begin{aligned}
O = [D : T_O, \quad & // \ \texttt{name}\,, \\
(A : (T_A,\ \mathbb{R} \cup V_T))^*, \quad & // \ \texttt{attribute}\,, \\
(P : O)^*, \quad & // \ \texttt{part}\,, \\
(S(A_O, A_P, A_K) : \mathbb{R})^*, \quad & // \ \texttt{relation}\,, \\
\mathbf{G} : \mathbb{R}^n] \quad & // \ \texttt{judgment}\,.
\end{aligned} \tag{3.1}
$$

The notation $(A : \ldots)^*$ indicates that there may be an arbitrary number (including zero) of attributes in an object. The initial description of an image is a network of segmentation objects $\mathcal{A} = \langle O \rangle$.

Concept For knowledge representation we employ a suitably defined version of a semantic network [7] because it allows well–structured representation, efficient utilization, different types of optimization procedures, explanation tools, and was shown to provide excellent results in various applications.

Basically, a formalism for knowledge representation in an image analysis system should allow one to represent in the computer a certain *conception* denoting an entity in the real (physical) world, for example, a 'highway', a 'car', an 'accident on a highway', and so on. We represent a conception by a recursive structure

C and call this internal representation a *concept*

$$\begin{aligned}
C = \big(D : T_C,& \qquad && \texttt{// name}\\
(P_{ci} : C)^*,& && \texttt{// context-indep. part}\\
(P_{cd} : C)^*,& && \texttt{// context-dep. part}\\
(K : C)^*,& && \texttt{// concrete}\\
(V : C)^*,& && \texttt{// specialization}\\
(L : I)^*,& && \texttt{// instance}\\
(A : (T_A \mapsto F))^*,& && \texttt{// attribute, computed by } F\\
(S(A_C, A_P, A_K) \mapsto F)^*,& && \texttt{// relation, computed by } F\\
(\mathbf{G} \mapsto F)\big)& && \texttt{// judgment, computed by } F
\end{aligned} \tag{3.2}$$

A detailed description of a concept in this sense is given in [7] and related work is found, for example, in [8, 6]. Some basic properties are illustrated in Figure 1.

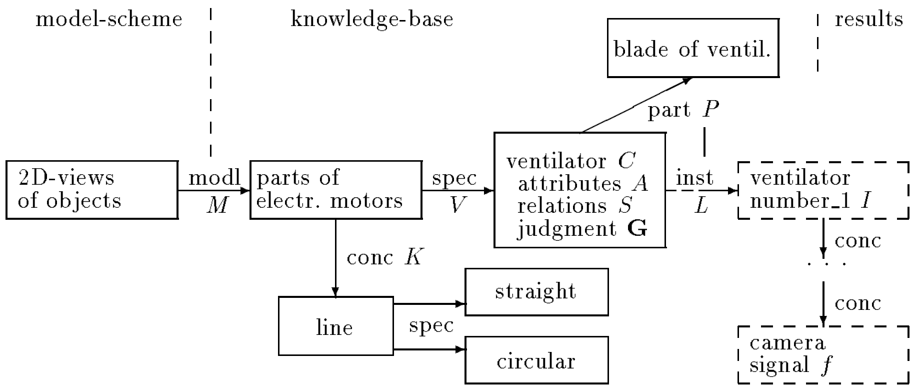

Figure 1. Examples of concepts linked to other concepts and modeling industrial parts.

The main defining components of a concept are its context–*in*dependent and context–*de*pendent *parts* P_{ci}, P_{cd} and *concretes* K. The distinction between parts and concretes allows one to represent relations between different *levels of abstraction* or different conceptual systems. For example, in Figure 1 the 'parts of electric motors' are determined from 'lines' which are in a different conceptual system (or closer to the level of pixels); they are therefore related by a concrete–link. We omit the model scheme (Figure 1) from further discussion here and refer to [9, 10].

The distinction between context–independent and –dependent parts allows one to handle context–dependent relations between objects and their parts. With respect to the usage of a knowledge base this means that a context–independent part may be detected or infered *without* having the superior concept, whereas a context–dependent part can only be infered *after* having the superior concept.

In order to have compact knowledge bases and to define hierarchies of conceptions it is possible to introduce a concept as the *specialization* V of some other (more general) concept.

According to the above definition a concept C has a set of *attributes* (or features, properties) A each one referencing a function F which can compute the value of the corresponding attribute.

There may be *structural relations* S between the attributes A_C of a concept or of attributes A_P and A_K of its parts and concretes. Since in image processing noise and processing errors are inevitable, relations usually can only be established with limited precision. Therefore, each relation references a function F computing a measure of the degree of fulfilment of this relation, for example, in the sense of a fuzzy relation.

In the same class of objects there may be quite different types of objects. For example, there are chairs with three and with four legs, and a chair may have an armrest or not. One possibility is to define separate concepts for each type, but in order to have compact knowledge bases it is convenient to introduce a more flexible definition of a concept. Therefore a concept is allowed to have different so called sets of modalities H_i with the implication that each individual set may define the concept. Furthermore, obligatory and optional parts are introduced.

Instance and Modified Concept The occurrence of a specific object in an image is represented by an *instance* $I(C)$ of the corresponding concept C. The relation between the concept and its instance is represented by a link L from C to $I(C)$. An instance is represented by a structure identical to (3.2) except that references to functions are replaced by the actual values computed by those functions from the image.

There may be the situation that some instances have been computed and allow the restriction of attribute values of a concept C which cannot yet be instantiated. In this case a so called *modified concept* $Q(C)$ is created which allows the propagation of constraints both in a bottom–up and top–down manner. For example, the detection of one 'wheel' of a 'car' constrains both the location of the car (bottom–up) and of the other wheels (top–down). This way *constraints* are propagated bottom–up and top–down.

A *judgment* $\mathbf{G}$ of an instance or a modified concept is a vector of numbers computed by F, measuring the degree of confidence in the actual occurrence of the instance and its expected contribution to the success of analysis.

Knowledge–Based Processing The available task-specific knowledge is represented in a *model* $\mathcal{M}$ which is a network of concepts $\mathcal{M} = \langle C \rangle$. Hence, knowledge about objects, events, tasks, actions, and so on is represented homogeneously by concepts related to each other by the various links.

In particular we assume that the goal of image analysis is itself represented by one or more concepts which we denote as the *goal concepts* C_{g_i}. The description of an image then is represented by an instance $I(C_g)$ of the goal concept. Since every concept has an attached judgment $\mathbf{G}$, there is also a judgment $\mathbf{G}(I(C_g))$ of the goal concept. Now it is natural to request the computation of an *optimal* *instance* $I^*(C_g)$ of a goal concept and define knowledge based processing as the optimization problem

$$I^*(C_g) = \max_{\{I(C_g)\}} \{\mathbf{G}(I(C_g)|\mathcal{M}, \mathcal{A})\} \ , \tag{3.3}$$

$$\mathcal{B}(\boldsymbol{f}) = I^*(C_g) \ . \tag{3.4}$$

The essential assumptions are that it is possible to

– compute a sufficiently reliable initial segmentation $\mathcal{A}$,
– acquire the relevant task-specific knowledge in the model $\mathcal{M}$,
– specify judgments $\mathbf{G}$ which are adequate for the task domain.

It has been demonstrated in several applications that this can be done, see for example, [11, 12].

Facing both the large amount of data and the limited processing time in most image understanding tasks, the exploitation of parallelism provides a promising way to compute an optimal instance $I^*(C_g)$ just in time with the sensory input. Whereas parallel algorithms for preprocessing and segmentation have nearly become a state of the art and may be found in various textbooks [13], parallel knowledge based processing is much less investigated. While the former algorithms often make use of the local nature of pixel–oriented computations, and therefore allow the use of simple data partitioning techniques, the parallelization of knowledge based processing is more difficult, since usually it requires the identification of inferences that may be executed simultaneously.

Most parallel semantic network systems employ an isomorphic mapping between the processors of a parallel hardware and the nodes and links of a knowledge base, which turned out to be a feasible approach if both concepts and inferences are simple [14]. However, since in our formalism a concept may have an arbitrary number of attributes and structural relations, complex concepts may become a bottleneck in parallel instantiation. Therfore, we employ an attribute centered representation of a semantic network, where each computation needed during instantiation is represented by a node of a directed acyclic graph [15, 16] that may be mapped to a multiprocessor system for purposes of parallel processing.

The automatic construction of the graph from the concept centered definition given by equation (3.2) is an important prerequisite to preserve the advantages of the well–structured knowledge representation by semantic networks. In our formalism this is possible, since the use of knowledge during instantiation only relies on the syntax of the network language, and not on the content of the knowledge base or intermediate results of analysis.

Since a concept usually is stored exactly once in the knowledge base, but it may be necessary to create several instances for one concept during analysis (as an example consider the wheels of a car), transformation of the network starts with the computation of the number of instances needed for instantiation of a goal concept C_g. This is achieved by a top–down expansion of the goal concept. Then, the expanded network is refined by the determination of dependencies between subconceptual entities. These are obtained from an examination of the interface to the procedural knowledge F, where a list of arguments is specified for each procedure. For each attribute, structural relation, and judgment of the expanded network a node v_k is created, and the name of the structure is attached to v_k as well as the corresponding procedures for the computation of a value or a judgment. If a node v_l is referenced via its name in the argument list of the procedures attached to v_k, a directed link $e_{lk} = (v_l, v_k)$ is created, expressing the fact that the computation of v_l must finish before the computation of v_k may start. Nodes without predecessors represent attributes that provide an interface to the initial segmentation, and nodes without successors usually represent the judgments of goal concepts.

4 Optimal Instantiation

As a prerequisite for goal–directed processing, we assume that a heuristic judgment function is available, which allows the quantitative treatment of alternative, uncertain, or imprecise results, that may arise during segmentation as well as during knowledge based processing (cf. Section 3). Results of initial segmentation in our applications are either scored heuristically or forwarded to knowledge based processing without a judgment of quality.

During knowledge based processing we have to distinguish between the judgment of an instance $I(C)$ of a concept C and the judgment of a state of analysis. The score of an instance in principle is always based on the quality of the match between data and model or on the quality of fulfilment of fuzzy inference rules. The score of a state of analysis is always based on the current estimate of the quality of a solution provided by a goal concept, see eq. (4.4) for combinatorial optimization. Therefore, instantiation is always directed towards optimal instantiation of a goal concept as requested in (3.3).

The attribute centered representation of a semantic network presented originally was developed to speed up analysis by the parallel computation of instances. Besides instantiation on the *network level* there is a *control level* (e.g. a search tree) that deals with the problem of finding an optimal interpretation, and in face of the required processing rates it seems reasonable to employ parallel processing on this level, too.

Parallel graph search algorithms, which are treated, for example, in [17], provide an obvious solution and are widely used in solving standard problems from artificial intelligence, like e.g. the Traveling–Salesman–Problem. However, since in our application state space search demands the collection of instances and modified concepts into nodes of the search graph, the control algorithm given above requires a large amount of communication between processors if the search space is explored in parallel. We therefore developed an *iterative optimization algorithm* which facilitates an efficient realization on a parallel hardware. In addition it provides an approximate interpretation at every iteration and therefore supports the fast computation of suboptimal interpretations, which may be used by other processing modules if less processing time is available (any–time property). For purposes of explanation, the algorithm may be divided into two stages:

– During bottom-up instantiation, values and judgments are computed for each attribute, structural relation, link or concept of the attribute network. Initially this will lead to many competing interpretations having low values of judgment.
– Interpretations obtained from bottom-up instantiation are iteratively improved by applying a combinatorial optimization procedure. Ideally, this will lead to a unique interpretation having a high value of judgment.

Bottom-up instantiation starts with the computation of attributes that provide an interface to the initial segmentation, and finishes with the judgment of goal concepts. Parallel bottom–up instantiation maps each node of the attribute network onto a processor and computes simultaneously those nodes, whose predecessors have already finished execution. We thereby obtain instances for the goal concepts, each provided with a judgment $\boldsymbol{G}(I(C_g))$. From the best scored instances we create a vector

$$\boldsymbol{g} = (\boldsymbol{G}(I^*(C_{g_1}), \ldots, \boldsymbol{G}(I^*(C_{g_n})) \tag{4.1}$$

representing the final result of a single iteration step.

Since results from inital segmentation are usually erroneous and due to the presence of ambiguities in the knowledge base (e.g. sets of modalities), there is a need for an efficient processing of competing hypotheses that are created during instantiation. Because the simultaneous treatment of competing hypotheses would result in both a large amount of communication between processors and a combinatorial explosion of intermediate results on the higher levels of the network, an iterative processing is employed.

The computation of instances (Figure 2) and their judgment is completely determined by assigning to each interface (attribute) node A_i a (possibly empty) subset $\{O_j^{(i)}\}$ of segmentation objects and selecting for each concept node C_k a unique modality $H_l^{(k)}$ (cf. Section 3). This allows us to characterize the *current state of analysis* by a vector

$$\boldsymbol{r}_c = \left[(A_i, \{O_j^{(i)}\}) \mid i = 1, \ldots, m\ ;\ (C_k, H_l^{(k)}) \mid k = 1, \ldots, n\right]\ , \tag{4.2}$$

where m is the number of interface nodes and n the number of concepts having more than one modality. Hence, we also may rewrite (4.1) as a function of the current state $\boldsymbol{r}_c$ by

$$\boldsymbol{g}(\boldsymbol{r}_c) = (\boldsymbol{G}(I^*(C_{g_1}), \ldots, \boldsymbol{G}(I^*(C_{g_n})|\boldsymbol{r}_c) \tag{4.3}$$

and compute an optimal state of analysis by treating $\boldsymbol{r}_c$ as the current state z_c of a combinatorial optimization problem.

Figure 3 gives a general algorithm for solving the problem of optimal instanti-

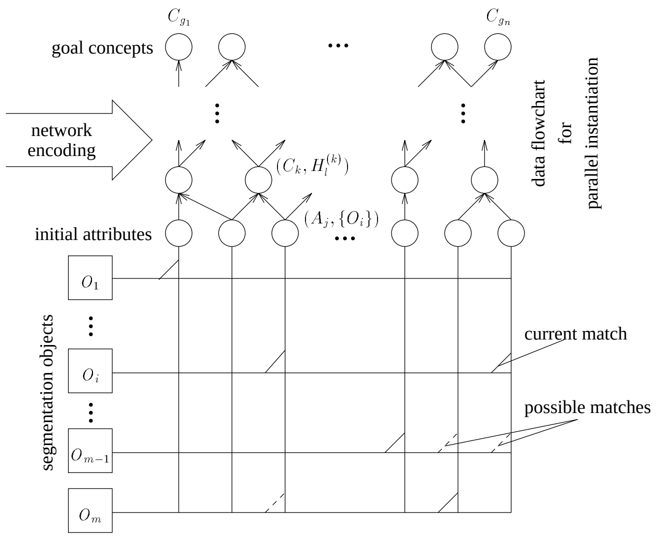

Figure 2. The principle of bottom–up instantiation. Hypotheses resulting from actual matches between initial attributes and segmentation objects are propagated through the data flowchart, resulting in a judgment of goal concepts.

```
given: state space $Z = \{z_1, z_2, \ldots\}$
define a control sequence $(T_k)$ with the following properties:
    $T_k > 0 \wedge \forall k \geq 0 : T_k \geq T_{k+1} \wedge \lim_{k\to\infty} T_k = 0$
create an initial state $z_0$ and compute $\phi_0 := \phi(z_0)$
$z_c := z_0$; $\phi_c := \phi_0$
FOR computing time available
    create a new state $z_n$ out of $z_c$
    compute $\phi_n := \phi(z_n)$
    compute the acceptance criterion $a(z_n)$
    IF $a(z_n) = \mathtt{T}$
    THEN let $z_n$ become the current state:
         $z_c := z_n$, $\phi_c := \phi_n$
report the current state $z_c$ as the solution
```

Figure 3.: Algorithm for the minimization of a cost function by combinatorial optimization.

ation (3.3–3.4) by combinatorial optimization. The parts printed sans serif have to be adapted for optimal instantiation.

For the design of the required cost function ϕ, we assume that an error–free segmentation would support a single interpretation, and therefore results in an ideal judgment $\boldsymbol{G}(I^*(C_{g_i})) = 1.0$ for the correct goal concept. On the other hand, at the same time this segmentation would give no evidence to any other goal concept, i.e. $\boldsymbol{G}(I^*(C_{g_j})) = 0.0$. We would thus obtain the i–th unit vector $\boldsymbol{e}_i$ as an ideal result of instantiation, if the i–th goal concept provides the desired symbolic description.

We approximate this behaviour and define

$$\phi(\boldsymbol{r}_c) = \min_{1\le i\le n}\{(\boldsymbol{e}_i - \tilde{\boldsymbol{g}}(r_c))^2\}, \qquad \tilde{\boldsymbol{g}}(\boldsymbol{r}_c) = \frac{\boldsymbol{g}(\boldsymbol{r}_c)}{||\boldsymbol{g}(\boldsymbol{r}_c)||} \tag{4.4}$$

as a cost function, that computes the minimum distance from $\boldsymbol{g}(\boldsymbol{r}_c)$ to the unit vectors $\boldsymbol{e}_i$. Note that this function provides a problem independent measure of costs, since no assumptions on the contents of the knowledge base are made.

Monte-Carlo-Methods The creation of a new state $\boldsymbol{r}_n$ and the choice of an acceptance criterion are motivated by the analogy between the annealing of a solid in condensed matter physics and the optimization of a system with many independent variables (cf. "simulated annealing algorithm" [18]).

In combinatorial optimization the temperature of a solid corresponds to the value T_k of a control sequence, and the cooling is achieved by a monotone decrease of T_k. The creation of a new state is performed by applying a small random perturbation to the current state $\boldsymbol{r}_c$. A new state $\boldsymbol{r}_n$ is generated from the current state $\boldsymbol{r}_c$ by first selecting a tuple from among $(A_i, \{O_j^{(i)}\})$ or $(C_k, H_l^{(k)})$ in (4.2) with equal probability, and then exchanging the term $\{O_j^{(i)}\}$ or $H_l^{(k)}$ by a possible alternative, again with equal probability for each alternative.

If the new state generated this way has *lower* cost than the current state, it is accepted. In addition, a new state with *higher* cost may also be accepted. The acceptance of new states with *higher* costs is necessary as well to allow the escape from local minima of the cost function. However, accepting *all* states would prevent the algorithm from convergence, and therefore an acceptance criterion $a(\boldsymbol{r}_n)$ is needed to decide which new state should be accepted or rejected. The general algorithm given in Figure 3 is specialized to different optimization procedures according to the different criteria

$$a(\boldsymbol{r}_n) = \begin{cases} T : q \le \exp(-1/T_k \cdot (\phi_n - \phi_c)) & \text{simulated annealing ,} \\ T : (\phi_n - \phi_c) \le T_k & \text{threshold acceptance ,} \\ T : \phi_n \le T_k & \text{great deluge ,} \\ T : \phi_n \le \phi_c & \text{stochastic relaxation ,} \\ F : \text{else .} \end{cases} \tag{4.5}$$

Whereas stochastic relaxation does not allow to escape from local minima, and therefore may be appropriate for certain cost functions only, the other algorithms derived from equation (4.5) can guarantee convergence into a global minimum, provided the decrease of the control parameter is sufficiently slow. Since this usually results in a large number of iterations and a large amount of computing time, there is a strong interest to speed up the optimization by a parallel exploration of the search space.

Genetic Algorithms Different from the optimization techniques introduced in the previous section, genetic algorithms perform a parallel exploration of the search space by the use of a set of current states or current solutions of the image interpretation problem. The set of solutions is called a *population*, the members of a population are referred to as *organisms*, and the population used in the k–th iteration of the algorithm is called the k–th *generation* [19]. Adopting the notation of (4.2) we denote the set of current states, that is, the current population, by

$$R_c = \{\boldsymbol{r}_{c,1}, \ldots, \boldsymbol{r}_{c,\mu}, \ldots, \boldsymbol{r}_{c,p}\} \quad , \tag{4.6}$$

and an organism of this population by

$$\boldsymbol{r}_{c,\mu} = \Big[(A_i, \{O_j^{(i)}\})_\mu \mid i = 1, \ldots, m \;;\; (C_k, H_l^{(k)})_\mu \mid k = 1, \ldots, n\Big] \quad . \tag{4.7}$$

Costs have to be assigned to each newly created organism. In our application this demands the bottom–up instantiation of the data flowchart for each newly created organism; they constitute the temporary population R_t. Therefore, especially in the presence of a large knowledge base, it is desirable to keep $|R_t|$ small to perform as many iterations as possible. Thus, we decided to create only *one* new organism in each step of iteration ($|R_t| = 1$), and to employ a multipoint crossover operator to make use of the full size of the current population R_c, instead of a pair of organisms as provided by a single application of the operator. Hence, genetic otpimization in our case is performed by the following three steps: First, a new organism is created by multipoint crossover; this selects from among the tuples $(A_i, \{O_j^{(i)}\})_\mu$, $\mu = 1, \ldots, p$ one tuple with a probability proportional to the fitness of that organism and makes this tuple an element of the new organism in R_t; this is repeated for all tuples $(A_i, \{O_j^{(i)}\})_\mu$, $i = 1, \ldots, m$ and all tuples $(C_k, H_l^{(k)})_\mu, k = 1, \ldots, n$. Second, each tuple of the new organism is mutated by a small probability (we use $p_m = 5\%$). Third, the new population R_n is obtained from the best organisms in $R_c \cup R_t$.

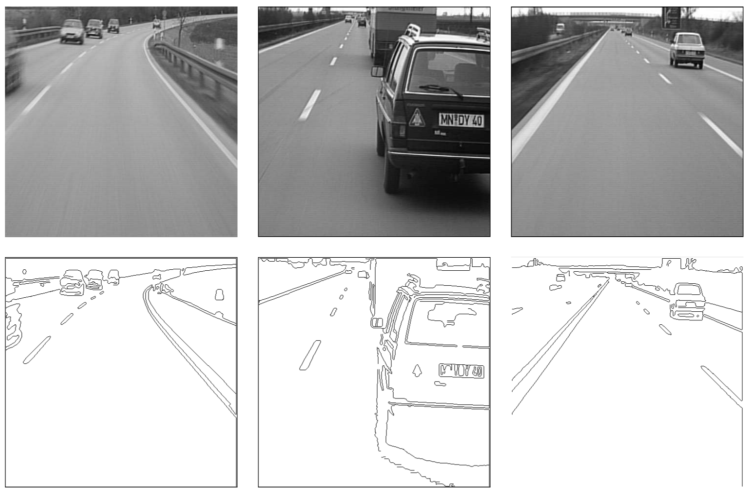

Figure 4. Gray–level images (top) and segmentation (bottom) from three different traffic scenes.

5 An Application

In this section we consider the problem of recognizing streets from TV image sequences recorded on a moving car. In the experiments described here, we used three image sequences, each consisting of 30 gray–level images, taken at video rate from a camera fixed in a moving car, see Figure 4 for an example from each sequence.

The goal of analysis was to obtain a description of the road and its markers, not to compute a complete interpretation in terms of cars, trucks, and traffic signs present in the scenes. Task–specific knowledge was developed in [20] and is represented in the semantic network formalism described in Section refA0301. An overview of the knowledge base is depicted in Figure 5. Encoding of the network into the attribute centered representation described results in a data flowchart consisting of 120 nodes, eleven of them representing initial attributes. On top of the flowchart, there is a single node for the computation of a judgment of the goal concept (in this case: "Street").

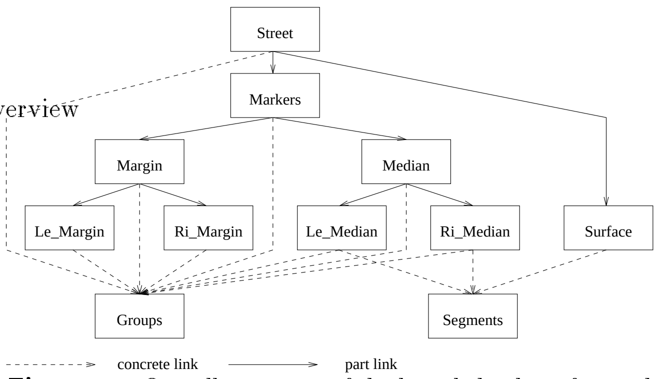

Figure 5.: Overall structure of the knowledge base for road detection.

Data driven processing yields segmentation objects of regions for the roadway as well as for groups of regions that are considered as road markers.

In [16] we examined the various Monte–Carlo–methods given by the acceptance criteria (4.5) and the different versions of genetic algorithms. The speed of convergence $\nu(n) = P\{\phi_n - \phi_{min} \leq \varepsilon\}$, which gives the probability to reach a nearly optimal solution in n iterations was used for the evaluation of all methods. For the results shown in Figure 6, the size of the population was $|Z_c| = 16$, and $|Z_t| = 1$, see Section 4.

The threshold acceptance and great deluge algorithm perform worse than stochastic relaxation and are therefore omitted in the discussion. However, it was

observable that all deterministic acceptance criteria perform significantly better than simulated annealing, which is in accordance to the results reported in [21, 22] for other optimization problems.
The computing time measured on a 1–master/p–slave configuration of coupled workstations (HP 735) using PVM is shown in Figure 7. The final result (computed from segmentation results in Figure 4) is given in Figure 8, which again shows one image out of each of the three image sequences. The grey value coding of the street markers and of the street surface show that those regions were interpreted correctly.

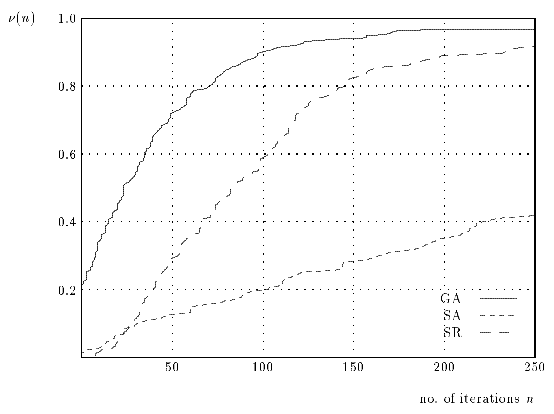

Figure 6.: Probability of convergence $\nu(n)$ after n iterations for the simulated annealing algorithm (SA), the stochastic relaxation algorithm (SR), and the multi point genetic algorithm (GA).

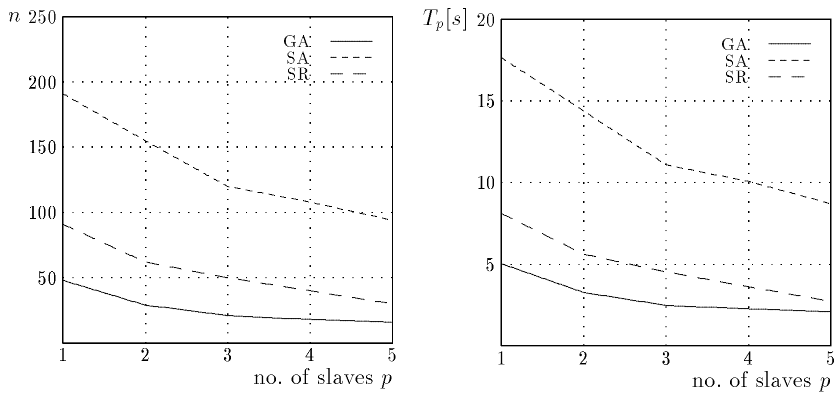

Figure 7. Number n of iterations (left) and computer time T_p (right) on a master/slave configuration of workstations.

In [23] the approach presented here for image understanding has also been successfully applied to *speech* analysis; this shows that the approach is useful for knowledge based pattern analysis in general.

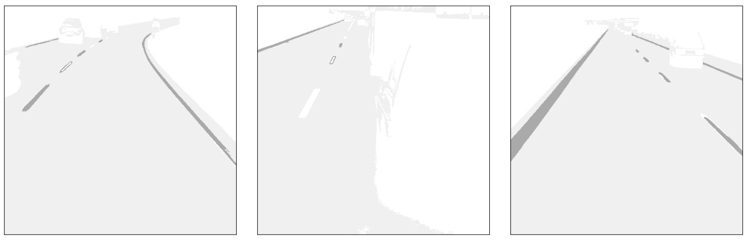

Figure 8. Result of interpretation by iterative optimization.

6 Conclusion and Outlook

We defined knowledge based image interpretation as the problem to compute an optimal instance of a goal concept. The two main processing steps are the data–driven computation of an initial segmentation and the model–driven interpretation by instantiation of the goal concept. We presented an approach to knowledge representation which started form a concept based representation in a semantic network and proceeded to an attribute based representation.

For further work we will consider the implementation of an active vision task by concepts of a knowledge base of the type introduced in Section 3. This requires a careful, precise, and explicit modeling of the goal of active vision. As an example we plan to consider the closed loop of sensing and acting (camera and robot arm) in order to find and grasp a specified object which may be invisible from the current camera position.

The above mentioned task requires consideration of another point for future work, that is real time performance. The parallel implementation as well as the any–time capability of iterative optimization seem to be useful steps in this direction.

References

1. T. Matsuyama and V. Hwang. *SIGMA: A Knowledge–based Aerial Image Understanding System.* Plenum Press, New York, 1990.
2. I. Masaki, editor. *Vision–Based Vehicle Guidance.* Springer, Berlin, 1992.
3. O. Faugeras. *Three–Dimensional Computer Vision.* Artificial Intelligence Series. The MIT Press, Cambridge, MA, 1993.
4. H. Niemann. *Pattern Analysis and Understanding.* Springer Series in Information Sciences 4. Springer, Berlin, 2. edition, 1990.
5. P. Krause and D. Clark. *Representing Uncertain Knowledge.* Intellect Books, Oxford, 1993.
6. J.F. Sowa, editor. *Principles of Semantic Networks.* Morgan Kaufmann, San Mateo, Calif., 1991.

7. H. Niemann, G. Sagerer, S. Schröder, and F. Kummert. ERNEST: A semantic network system for pattern understanding. *IEEE Trans. on Pattern Analysis and Machine Intelligence*, 9:883–905, 1990.
8. A. Kobsa. The SB-ONE knowledge representation workbench. SFB 314 (XTRA), Memo Nr. 31, Univ. des Saarlandes, FB 10, Saarbrücken, F. R. of Germany, 1989.
9. D. Paulus, A. Winzen, and H. Niemann. Knowlege based object recognition and model generation. In *Proc. Europto 93, Computer Vision for Industry,* pages 382–393, München, 1994. SPIE Proc. No. 1989–47.
10. A. Winzen. Automatische Erzeugung dreidimensionaler Modelle für Bildanalysesysteme. Dissertation, Technische Fakultät, Universität Erlangen–Nürnberg, Erlangen, (1994).
11. H. Niemann, H. Bunke, I. Hofmann, G. Sagerer, F. Wolf, and H. Feistel. A knowledge based system for analysis of gated blood pool studies. *IEEE Trans. Pattern Analysis and Machine Intelligence*, 7:246–259, 1985.
12. H. Niemann, H. Brünig, R. Salzbrunn, and S. Schröder. A knowledge-based vision system for industrial applications. *Machine Vision and Applications*, 3:201–229, 1990.
13. H. Burkhardt, Y. Neuvo, and J. Simon, editors. *From Pixels to Features II. Parallelism in Image Processing.* North–Holland, Amsterdam, 1991.
14. L. Shastri. *Semantic Networks: An Evidential Formalization and its Connectionist Realization.* Research Notes in Artificial Intelligence. Pitman and Morgan Kaufmann Publishers, Inc., London and San Mateo, Calif., 1988.
15. V. Fischer and H. Niemann. Parallelism in a semantic network for image understanding. In A. Bode and M. Dal Cin, editors, *Parallel Computer Architectures. Theory, Hardware, Software, Applications*, volume 732 of *Lecture Notes in Computer Science*, pages 203–218. Springer-Verlag, Berlin, 1993.
16. V. Fischer. Parallelverarbeitung in einem semantischen Netzwerk für die wissensbasierte Musteranalyse. Dissertation, Technische Fakultät, Universität Erlangen–Nürnberg, Erlangen, 1995.
17. B.W. Wah, G. Li, and C. Yu. Multiprocessing of combinatorial search problems. In [24], pages 103–145. 1990.
18. N. Metropolis, A. Rosenbluth, M. Rosenbluth, A. Teller, and E. Teller. Equation of state calculations for fast computing machines. *Journal of Chemical Physics*, 21(6):1087–1092, 1953.
19. L. Booker, D. Goldberg, and J. Holland. Classifier systems and genetic algorithms. *Artificial Intelligence*, 40(1–3):235–282, 1989.
20. S. Steuer. Erstellung eines ersten Modells in ERNESTzur Identifikation der Straße und der Position des Kamerafahrzeugs im statischen Bild. Technical Report 3.2.B1 Projekt MOVIE, Bayerisches Forschungszentrum für Wissensbasierte Systeme (FORWISS) und Bayerische Motorenwerke AG (BMW AG), München, 1991.
21. G. Dueck and T. Scheuer. Threshold accepting: A general purpose optimization algorithm appearing superior to simulated annealing. *Journal of Computational Physics*, 90(1):161–175, 1990.
22. G. Dueck. New optimization heuristics: The great deluge algorithm and the record-to-record-travel. *Journal of Computational Physics*, 104(1):86–92, 1993.
23. V. Fischer, J. Fischer, and H. Niemann. An algorithm for any-time speech understanding. In *German Slovenian Workshop on Image and Speech Understanding,* to appear, Ljubljana, 1996.
24. V. Kumar, P. Gopalakrishnan, and L. Kumar, editors. *Parallel Algorithms for Machine Intelligence and Vision.* Springer-Verlag, New York, 1990.

Here comes the llnc info!