Available online at www.sciencedirect.com
ScienceDirect
Image and Vision Computing 26 (2008) 82–101
www.elsevier.com/locate/imavis

On scene interpretation with description logics

Bernd Neumann [a,*], Ralf Möller [b]

[a] Universität Hamburg, Vogt-kölln-Str. 30, 22527 Hamburg, Germany
[b] Technische Universität Hamburg-Harburg, Harburger Schlossstr. 20, 21079 Hamburg, Germany

Received 15 July 2004; received in revised form 18 August 2005; accepted 7 August 2007

Abstract

We examine the possible use of description logics (DLs) as a knowledge representation and reasoning system for high-level scene interpretation. It is shown that so-called aggregates composed of multiple parts and constrained primarily by temporal and spatial relations can be used to represent high-level concepts such as object configurations, occurrences, events, and episodes that are required in an application context. Scene interpretation is modelled as a stepwise process which exploits the taxonomical and compositional relations between aggregate concepts while incorporating visual evidence and contextual information. It is shown that aggregates can be represented by concept expressions of a description logic which provides a concrete-domain extension for quantitative temporal and spatial constraints. The analysis reveals that different kinds of representation constructs have to be carefully selected in order to provide for the required expressivity while retaining decidability in general as well as practical support from description logic system implementations in particular. Reasoning services of the DL system can be used as building blocks for the interpretation process, but additional information is required to generate preferred interpretations. A probabilistic model is sketched which can be integrated with the knowledge-based framework.

Keywords: Scene interpretation; Description logics; High-level vision

1. Introduction

Interpreting a visual scene is a task which in general resorts to a large body of prior knowledge and experience of the viewer. Consider an every-day street scene as illustrated in Fig. 1.

Based on common-sense knowledge and experiences, we recognise that two persons are engaged with garbage collection while a third person is distributing mail. With visual evidence as sparse as a single snapshot, we obtain an interpretation which extends over time, supplements invisible objects outside the field of view, ignores uninteresting details, provides an estimate of daytime and season, and may even include assumptions about the intentions and emotions of the people in the scene. It is evident that scene interpretation is a knowledge-intensive process which is decisively shaped by the way common-sense knowledge and experiences are brought to bear.

While people seem to perform scene interpretations without effort, this is a formidable and as yet unsolved task for artificial vision systems. One reason is the often still unsatisfactory performance of low-level vision, in particular segmentation, tracking, 3D analysis, object recognition, and categorisation. Often it is argued that the problem of complex scene interpretation cannot be tackled before reliable low-level results are available. However, low-level vision is not always the bottleneck. As the above example suggests, an even more important role may be played by high-level knowledge and experiences. Given suitable high-level knowledge structures, far-reaching interpretations may be obtained including propositions about parts of the scene for which there is no direct evidence at all.

* Corresponding author. Tel.: +49 40 42883 2451; fax: +49 40 42883 2572.
E-mail addresses: neumann@informatik.uni-hamburg.de (B. Neumann), r.f.moeller@tu-harburg.de (R. Möller).

0262-8856/$ - see front matter © 2007 Elsevier B.V. All rights reserved.
doi:10.1016/j.imavis.2007.08.013

Fig. 1. Street scene for scene interpretation.

Furthermore, high-level knowledge may provide top-down guidance to facilitate and improve low-level processes. This has been known for a long time (e.g. [1]), but there are few examples (e.g. [2]) where vision systems exploit high-level knowledge – beyond single-object descriptions – for low-level processing and decisions.

In view of the importance of knowledge for scene interpretation, it is useful to be aware of the rich body of research on knowledge representation and knowledge-based system methodology when designing a scene interpretation system. For an overview see the corresponding sections in AI textbooks such as [3–5]. Out of the many aspects of past and ongoing developments in knowledge representation, the following seem to be particularly significant for scene interpretation.

1. Knowledge representation needs a sound formal basis when the body of knowledge becomes large and diverse. Many of the early representation formalisms such as semantic networks, early frame languages and rule systems suffer from the lack of precise semantics in the sense that the correct use of represented knowledge is partly based on intuitive notions which do not necessarily provide a consistent basis for large-scale knowledge processing.
2. Knowledge representation systems may provide standardised inference services, which can be used (and reused) for application development. Typical inference services are consistency checking, inheritance, instance classification, and model construction, but many more have been proposed and investigated, for example pattern matching services [6]. Inference services are interesting for scene interpretation as they may provide important functionality for the interpretation process in terms of existing software with well-defined properties.
3. There is a growing body of research about spatial and temporal knowledge and related reasoning services [7–9]. Space and time play a dominant role in visual scenes, and one may hope that spatial and temporal reasoning services provide useful support for scene interpretation. However, it is conspicuous that so far only few examples exist where spatial and temporal reasoning services have been integrated into a vision system [10–12]. One of the problems seems to be the mismatch between the quantitative spatial and temporal information arising from low-level vision and the mostly qualitative nature of spatial and temporal reasoning services.
4. Description logics (DLs) constitute a family of knowledge representation formalisms which have obtained much attention in the last decade. DLs provide object-oriented knowledge representation similar to frame systems used in many knowledge-based application systems, but based on formal semantics. DLs realise a subset of First Order Predicate Calculus. The subset is generally chosen as to guarantee the decidability of consistency checking and other key inference services. Furthermore, recent developments of sophisticated optimisation techniques have led to implemented DL systems which combine an expressive representation language with highly efficient services. Baader et al. [13] provide an excellent overview of the state-of-the-art of DL methodology.

In this contribution we report about an approach to using a DL for high-level scene interpretation. The insights and results are primarily based on long-standing work both on high-level vision and on formal knowledge representation in the Cognitive Systems Laboratory at Hamburg University, but certainly also try to reflect the development of the two fields in their respective research communities. The

organisation of the following sections roughly mirrors the corresponding research history.

In Section 2 we examine the conceptual structures which are needed to represent knowledge for high-level vision. The guiding scenario is a living-room, observed by a stationary smart-room camera. A typical scene is table-laying, when one or more human agents place dishes onto the table and the system has the task to recognise table-laying occurrences. Laying a table is, of course, only an exemplary task, and the goal is to develop a methodology which is applicable to high-level scene interpretation in greater generality. For example, based on this methodology, it should also be possible to recognise interesting occurrences in traffic scenes (as a possible task of a driver assistance system), team behaviour in soccer (or robocup) games, criminal acts in monitoring tasks, etc. Occurrences, object configurations and other high-level structures can be represented by aggregates which are introduced informally as representational units. Compositional and taxonomical hierarchies of aggregate concepts are proposed as the main structures of a high-level conceptual knowledge base. The aggregate structure represents the representational requirements which must be met by a DL system.

In Section 3 we discuss requirements for the interpretation process within the conceptual framework introduced before. In high-level vision, interpretation tasks may be highly context-dependent, involving prior information from diverse sources. Scene evidence may be incomplete, in particular in evolving time-varying scenes. Hence hypothesis generation and prediction become important issues. However, it is known that in the end, a valid scene interpretation must be a "model" (in the logical sense) of the conceptual knowledge and the scene data.

After having discussed knowledge-representation requirements for high-level scene interpretation, we examine the potential of DL systems for this task. In Section 4 we give an introduction to the family of DLs and the conceptual expressions which can be formulated. As an extension important for scene interpretation, symbolic reasoning may be augmented by predicates over concrete domains such as real numbers representing temporal or spatial coordinates. We also introduce inference services offered by DL systems. They provide benefits both for knowledge base maintenance and application development.

In Section 5 we examine the use of DL knowledge representation and inference services for scene interpretation. It is shown that the representational requirements for high-level vision aggregates can in fact be met by description logic representation constructs. Regarding inference services for scene interpretation, logical model construction – which is a service provided by modern DL systems such as RACER or FaCT – is in principle a candidate. However, scene interpretation requires that the logical models not only satisfy all constraints expressed by conceptual knowledge and visual evidence, but also be most "plausible" or "preferred" with respect to a measure. Furthermore, the interpretation process must be flexible to adapt to a given focus of attention and other situational context. While this poses requirements that cannot be met by existing DL systems, such an integrated interpretation process appears to be realisable in principle.

In Section 6 we briefly describe ongoing work towards an interpretation system where probabilistic information guides the interpretation process within the conceptual framework of a formal knowledge representation system.

Section 7, finally, summarises our findings and suggests directions for further research. One of the major impediments for decisive progress appears to be the prevailing segregation of the respective research communities of Computer Vision and Knowledge Representation. So far, the Computer Vision community has not succeeded in attracting significant attention of the Knowledge Representation community for research into high-level vision. But this is not really surprising in view of the enduring predominance of lower-level vision research.

2. Conceptual structures for high-level scene interpretation

In this section we first explain what we mean by "high-level interpretation". We then propose conceptual structures which can describe such "interpretations". We introduce "aggregates" as representational units for object configurations, occurrences, episodes and other concepts which occur in high-level interpretations. We also discuss the interface between conceptual high-level descriptions and the data provided by lower-level processes.

2.1. High-level interpretations

We define high-level scene interpretation as the task of "understanding" a scene beyond single-object recognition. In a knowledge-based framework, a high-level interpretation is determined by constructing a description of the scene in terms of concepts provided in a conceptual knowledge base (Fig. 2). A scene is assumed to be a connected region of the four-dimensional space–time continuum.

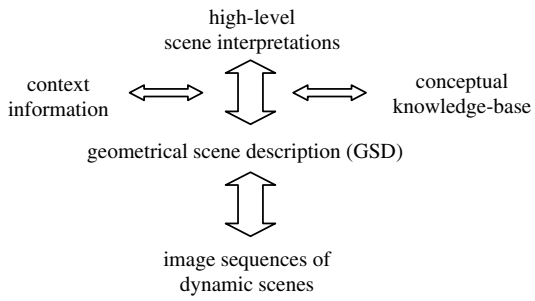

Fig. 2. Knowledge-based framework for high-level scene interpretation.

Our guiding example is a table-laying scene in a living-room where table-laying actions are observed over a certain time interval. We do not commit ourselves to a particular camera setup but simply assume that visual evidence is associated with the scene.

In order to be able to focus on high-level interpretation we will bypass lower-level image analysis issues and assume that a partial geometrical reconstruction of the scene in terms of objects and their properties is available which will constitute the input to high-level interpretation. This intermediate representation, called Geometrical Scene Description (GSD), has been introduced in earlier work [14] as a convenient separation between high-level and lower-level processes. In this work, however, we assume that high-level and lower-level processes will be able to interact. In fact, it is one of the goals of high-level processes to provide expectations and support for lower-level processes. Hence a GSD is not assumed to be complete and correct in any sense. In particular, objects in the GSD need not be fully classified, may be missing or may represent multiple scene objects. Imperfections at the level of the GSD will be a touchstone for robust high-level interpretation.

What are the requirements for describing scenes at a "high" conceptual level? From the examples given earlier we gather several characteristics. High-level scene interpretations typically

- involve several objects and occurrences;
- depend on the temporal and spatial relations between parts of the scene;
- describe scenes in qualitative terms, omitting geometrical detail;
- exploit contextual information;
- include inferred facts, unobservable in the scene;
- are based on conceptual knowledge and experiences about the world.

Consider, for example, the table-laying scene with a snapshot shown in Fig. 3. A high-level interpretation would express that a person is placing a cover onto a table. This is a qualitative summary of several individual occurrences involving different objects. The scene has a characteristic spatio-temporal structure. The final spatial configuration is described by the term "cover" referring to a priori knowledge about dish arrangements. Similarly, there is a typical temporal structure of the scene. For example, usually we would expect that the plate is placed before the saucer and the cup. Further expectations may arise from context information. If we know that it is early in the morning, we might infer that a breakfast table is laid and someone may intend to have breakfast soon.

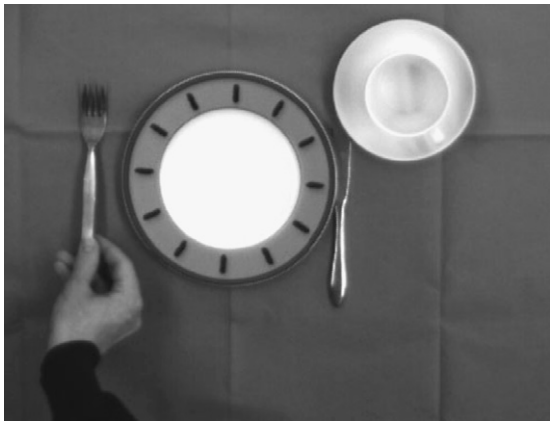

Fig. 3. Snapshot of a table-laying scene.

From the example, it is apparent that a scene interpretation may involve many conceptual levels above the level of single-object recognition, corresponding to different degrees of abstraction. At a low abstraction level we may talk about placing a fork beside a plate. At a higher level we may say that the table is laid for breakfast. It will be the task of the conceptual knowledge base to provide the corresponding conceptual structures.

Intuitively, we may think of the elements of a high-level scene interpretation as "occurrences". The term emphasises the general case of a time-varying scene (ranging from simple object motions to large-scale episodes), but is not meant to exclude concepts for stationary situations such as a cover configuration on a table.

It has been mentioned that an interpretation should exploit contextual information. As "context" of a scene we denote any information at any abstraction level which is relevant for the interpretation of that scene but not observable. For vision, spatial, and temporal context are particularly important. Spatial context is understood to influence the interpretation of a scene via spatial constraints. For example, context information about the location of the table border will constrain expected cover locations. Similarly, temporal context provides temporal constraints, for example, knowing the daytime may exclude certain interpretations such as "breakfast-table". The example suggests that it may be more appropriate to change certainty values rather than exclude an interpretation altogether. Uncertainty management as an extension of a logic-based framework will be addressed in Section 6.

In general, context may be provided in terms of diverse kinds of information. For example, it may be known by verbal communication that the table is being laid. This top-down information may facilitate a detailed scene analysis and interpretation. Context may also be given in terms of known intentions of agents. For example, if it is known that an agent intends to have breakfast, but the table is covered with other items, say books, then it may be expected that the agent will clear the table and then place dishes.

Another kind of context may be given by attention-directing queries in the sense that only specific information about the scene is interesting. In the smart-room setting of

our example scenario, for example, attention may be directed by queries of a human user such as "Is there a plate on the table?". The query will restrict the space of interesting interpretations to those which include a plate.

Similarly, context may also be provided in terms of the task for which some information is important and other is not. For example, a navigating agent may only be interested in obstacle avoidance and hence be interested in object geometry irrespective of object classes.

In our approach, context will be represented by factual knowledge about a scene beyond the evidence from visual sensors and general conceptual knowledge.

2.2. Aggregates

We turn now to the task of describing occurrence concepts in a knowledge-representation framework. This will be done initially in a frame-based notation. In Section 5, we will rephrase the frame-based models as conceptual expressions of a description logic.

The main conceptual entities are called aggregates. An aggregate consists of a set of parts tied together to form a concept and satisfying certain constraints. There are no a priori assumptions about dependencies between parts or specific reasons to combine them in an aggregate. We simply assume that one is interested to recognise an aggregate as a whole.

As an example, consider the occurrence of placing a cover on a table. Fig. 4 shows the corresponding conceptual model. It is a crude conceptual description of a scene where a plate, a saucer and a cup are placed onto a table to form a cover. The place-cover aggregate includes a table top, three transport occurrences and a cover configuration as parts (the spatial constraints expressed by cover are not shown here). Parts are assumed to be existentially quantified. Furthermore, there are time marks which refer to the beginning and ending of the place-cover occurrence. In the constraints section, there are identity constraints, such as pc-tp1.tp-ob = pc-cv.cv-pl, which relate constituents of different parts to each other (the plate of the transport sub-occurrence is identical with the plate in the cover), and qualitative constraints on the time marks associated with sub-occurrences. For example, pc-tp3.tp-te $\geqslant$ pc-tp2.tp-te denotes that the cup transport should end after the saucer transport. Aggregates involving mobile objects typically require that the objects fulfill certain temporal and spatial constraints. In high-level scene interpretation, we typically consider aggregates where spatial and temporal constraints express qualitative relations and dependencies between parts. But it is also possible to express crisp quantitative relations as, for example, between rigidly connected parts.

The example shows that an aggregate may have other aggregates as parts. Hence a compositional hierarchy is induced. The hierarchy is built on top of primitive occurrences which are generated as part of the GSD which will be discussed further down.

In many cases, the parts of an aggregate will have disjoint primitives and the compositional hierarchy will be a tree. But there may also be aggregate concepts which allow shared primitives. For example, one might define a "coffee-cover" as a cover with a coffee pot within reach. Clearly, one coffee pot may be part of several coffee-covers. Another example is the arrangement of covers in Dijkstra's Dining Philosophers Problem where five covers are arranged along a circular table, each cover sharing a fork with the neighbour to the left and right. We propose that, in general, aggregates may share primitives and disjointedness must be expressed by constraints. A more detailed discussion of the ontological assumptions in the aggregate hierarchy will be postponed until Section 5.

As indicated by the "parents" slot, aggregates are also embedded in a taxonomical hierarchy which is the usual organisational form for concepts at different abstraction levels.

Note that scene objects such as plate, saucer etc. are considered as aggregates composed of (i) a physical object or "body" in the 3D world and (ii) a "view" which is the visual evidence of the object in the camera view. As an example, Fig. 5 shows the conceptual model of a plate in a scene, where plate body and plate view are combined as an aggregate.

The constraints section contains constraints which relate the parts to each other, e.g. ensuring that the view is compatible with the 3D shape of the physical object (which is, of course, not trivial). In our approach, we do not intend an axiomatisation of depiction as investigated, for example, in [15,16], but prefer a model where the association of views with physical bodies can be learnt by experience. Hence constraints should be thought of as capturing experiences rather than laws of depiction.

name:	place-cover
parents:	:is-a agent-activity
parts:	pc-tt :is-a table-top
	pc-tp1 :is-a transport with (tp-obj :is-a plate)
	pc-tp2:is-a transport with (tp-obj :is-a saucer)
	pc-tp3 :is-a transport with (tp-obj :is-a cup)
	pc-cv :is-a cover
time marks:	pc-tb, pc-te :is-a timepoint
constraints:	pc-tp1.tp-ob = pc-cv.cv-pl
	pc-tp2.tp-ob = pc-cv.cv-sc
	pc-tp3.tp-ob = pc-cv.cv-cp
	...
	pc-tp3.tp-te ≥ pc-tp2.tp-te
	pc-tb ≤ pc-tp3.tb
	pc-te ≥ pc-cv.cv-tb

Fig. 4. Conceptual model of a place-cover scene.

name:	plate
parents:	:is-a scene-object
parts:	pl-body :is-a body with pl-body-preds
	pl-view :is-a view with pl-view-preds
constraints:	(constraints between pl-body-preds and pl-view-preds)

Fig. 5. Conceptual model of a plate in a scene.

Note that the aggregate and its parts are embedded in distinct taxonomical hierarchies: scene objects, bodies, and views. Only aggregates with locally coherent parts will be modelled with a view, for example a candlestick. Aggregates with disjoint mobile parts, such as a cover, will in general not be described by views at the aggregate level.

The main motivating criterion for defining an aggregate is to provide a coherent description of entities which tend to co-occur in a scene. This is regardless of whether the entities are visible or not. In fact, aggregates provide the means to hypothesise parts without evidence. As an extreme example, aggregates may include mental states of agents along with occurrences in a scene, in particular desires or emotional states. The aggregate in Fig. 6 is a sketch of an ''intended place-cover'', specifying an agent along with the place-cover occurrence and a desired cover configuration as the mental state of the agent. Relational descriptions including mental states have also been used in [17] as a basis for situation semantics.

The view concepts associated with physical object concepts refer to the interface between high-level and lower-level vision, as instances of view concepts are provided by lower-level processes. The next subsection deals with this interface.

2.3. Interfacing high-level and lower-level representations

The main task of the interface between high-level and lower-level vision is to ground symbols of symbolic descriptions in data structures provided by lower-level vision processes. It is assumed that, from below, the scene is described in terms of segments or blobs, each endowed with a rich quantitative description. As mentioned before, a similar scene description, denoted Geometrical Scene Description (GSD), has been introduced in earlier work [14]. Here, we do not require that objects of the GSD have been preclassified, but only postulate that view classes can be distinguished, e.g. ''disk-shaped'', in addition to the detailed quantitative description. A single view instance may be related to several object concepts, hence unambiguous recognition solely based on views may not be possible.

This way, the task of object recognition, i.e. assigning a view instance to a particular class of scene objects, is integrated into the scene interpretation framework (and thus can be influenced by contextual information, among others). In our approach, we do not attempt to model preference information pertaining to ambiguous classification situations within the logic-based framework. While this might be formally possible, for instance, using default reasoning techniques [18,19], the need to formulate explicit conditions for each default decision does not seem to be a feasible basis for a general scene interpretation framework. Instead, we will commit preference decisions to probabilistic methods which can be integrated with the logic-based framework at such decision points. A sketch of a probabilistic approach for preference decisions is given in Section 6.

name:	intended-place-cover
parents:	:is-a intended-action
parts:	ipc-pc :is-a place-cover
	ipc-ag :is-a agent
	ipc-cv :is-a cover
constraints:	ipc-ag.desire = ipc-cv
	(and other constraints)

Fig. 6. Conceptual model of an intended action.

In addition to instances of object views, qualitative relations between object views are computed, for example topological relations such as ''touch''. There is a large set of relations which can in principle be computed from the GSD. From a cognitive perspective, qualitative predicates over distances and angles between suitable reference features, as well as temporal derivatives of distances and angles, are of primary importance. For example, qualitative spatial relations such as ''right-of'' or ''parallel-to'' are of this kind.

In general, it may not be feasible to compute distances and angles between all pairs of objects. Utility measures and focus of attention come into play as well as verification requests of higher-level interpretation processes. It is therefore useful to think of instances of qualitative relations in terms of information which can be provided on demand.

In dynamic scenes, object motion and time-dependency of relations play an important part. The interface provides instances of views of *primitive occurrences* which are the basic building blocks for occurrences such as ''place-cover'' and other higher-level concepts. A primitive occurrence is defined as a conceptual entity where a qualitative relation is true over a time interval. Typical primitive occurrences are:

- object motion,
- straight object motion,
- approach or depart segment of an object motion relative to a second object,
- turning object motion,
- upward or downward motion.

If a predicate over a perceptual primitive is true throughout a scene, one usually does not talk about an occurrence. We will use the term *primitive relationship* instead, well aware that there is no inherent representational difference between a constancy which happens to change within the duration of a scene and one which does not.

3. Requirements for the high-level scene interpretation process

In this section we identify requirements which must be met by a high-level scene interpretation process. Further down, these requirements will be compared with existing inference services of DL systems.

3.1. Context-based interpretation

An interpretation of a scene is a partial description in terms of instances of concepts of the conceptual knowledge base. It is partial because only parts of the scene and a subset of the concepts are interesting in general, depending on the pragmatic context. This principle is well known from work on Active Vision [20] and knowledge-based attention mechanisms [21]. In our knowledge-based framework, we allow an interpretation to be incomplete in three respects:

(i) Objects need not be identified as parts of an aggregate. In particular, view objects may remain "unrecognised", i.e. not assigned to a scene-object aggregate.
(ii) An object need not be assigned to the most-specific concept describing this object.
(iii) Aggregates need not be instantiated at the parts level.

Context information can enter the interpretation process in terms of instantiated aggregates which constrain other possible scene objects. For example, if the context of a breakfast scene is given, it is assumed that a corresponding aggregate is instantiated and possible parts – such as the occurrence "laying-the-breakfast-table" – are expected as constituents of the interpretation. Context-based instances are often not fully specified, with properties left open or partially constrained. For example, the begin and end times of an instance of "laying-the-breakfast-table" may initially be loosely constrained to the typical morning hours, e.g. to times between 6 and 11 a.m.

Spatial and temporal context play a special part in scene interpretation, since spatial and temporal constraints provide important coherence in visual aggregates. The constraints section in aggregates will contain predominantly spatial and temporal constraints. In view of interpretation tasks under varying contextual conditions it is highly desirable that temporal and spatial constraints can be propagated between all constraint variables. For example, if a plate is interpreted as part of a cover, the plate location constrains other part locations, restricting possible choices and possibly even causing top-down guided image analysis in restricted areas.

As a consequence of context information, scene interpretation may be performed under diverse boundary conditions and the interpretation process must be influenced accordingly, in particular regarding the order in which possible hypotheses are tested. Hence one of the requirements for interpretation services must be flexibility to adjust to varying contexts.

3.2. Navigating in hallucination space

An interpretation may reach far beyond visual evidence, for example, by including predictions about the temporal development of a dynamic scene or expectations about invisible objects. Hence instantiations with incomplete or no visual evidence are more the rule than the exception. This is aptly expressed by the sentence "Vision is controlled hallucination" attributed to Max Clowes (1971).

Considering the potentially large space of possible hallucinations and the flexibility required for varying contexts, it is useful to model the interpretation process as an incremental construction process with the goal to create and verify any instance which may be useful for the overall goals of the vision system. We know that logically, an instance of an aggregate C can only be verified if it is asserted by context information or its parts can be verified under the constraints specified in the concept definition of C. This recursive definition may eventually bring scene objects and hence visual evidence of the GSD into play. But scene objects cannot be verified – logically – from visual evidence alone as shown in Fig. 5, but would require assertions about the corresponding physical object. Hence logical verifiability cannot be a criterion for accepting an instance in an interpretation. However, it can be assured that interpretations are consistent with evidence and conceptual knowledge. Unfortunately, the space of consistent interpretations may be huge and the knowledge-representation framework does not offer a suitable criterion for preferring one consistent interpretation over the other. Hence additional information is required, for example in terms of likelihoods of interpretations. We will discuss preference measures for guiding the interpretation process in Section 6.

In [22] a repertoire of three basic interpretation steps has been identified: aggregate instantiation, instance refinement and instance merging. For clarity, it is useful to further distinguish two variants of instance refinement: instance specialisation and instance expansion. The interpretation steps are designed to move around freely in hallucination space, i.e. to allow the construction of any consistent interpretation. In the following, the four kinds of interpretation steps will be described.

Aggregate instantiation is the act of inferring an aggregate from parts, also known as part–whole reasoning. Given instances of (not necessarily all) parts of an aggregate and satisfied constraints, we want to establish an instance of the aggregate. The question when evidence in terms of parts justifies aggregate instantiation is, of course, related to the verification question raised above, and we note that aggregate instantiation requires guiding information.

The second kind of interpretation step is *instance specialisation*. Specialisation means tightening properties and constraints, either along the specialisation hierarchy or by checking objects for possible roles in aggregates. Hence instance specialisation steps are predetermined by the structure of the specialisation hierarchy and the aggregate definitions. As above, it must be noted that the conceptual structures do not specify preferred choices if alternative specialisations are possible.

The degree to which instances should be specialised depends on the overall task of the vision system, and no generally valid rule can be given. On the other hand, we know from Cognitive Science that "natural kinds" play an important role in human thinking and communication. Roughly, a natural kind is a concept which describes essential visual properties of its instances [23]. In our domain, "plate" is a natural kind whereas "dish" is not. Asserting natural kinds could be a useful guiding goal for specialisation steps.

Instance expansion is the step of instantiating the parts of an aggregate if the aggregate itself is already instantiated. Logically, asserting an aggregate instance would generally imply the assertion of parts instances. But for a task-oriented and context-dependent interpretation it is useful to be able to suppress details. Hence it will not be required that parts are instantiated if an aggregate is instantiated. A typical reason for instance expansion is the need to connect higher-level aggregates to visual evidence.

The fourth kind of interpretation step, *instance merging*, is required because of the distributed nature of interpretation activities. New instances may be generated at any level and in any branch of the compositional hierarchy depending on visual evidence, context information and current interpretation state. Hence different sequences of interpretation steps may lead to identical instances which must be merged. This will happen in particular when instantiations are initiated both bottom-up and top-down, for example caused by visual evidence on one side and strong context-based expectations on the other. In our domain, context information such as "the table is laid" may have led to the top-down instantiation of a cover and its parts. Visual evidence about a plate and other items must then be merged with these instances. Again, we note that there may be many choices, and guiding information is needed.

3.3. Scene interpretation as model construction

As shown by Reiter and Mackworth [24] and further elaborated in [25,26,15], image interpretation can be formally described as constructing a partial model. "Model" is used here in the logical sense and means a mapping from the symbols of logical formulae into a domain such that the formulae are true.

Applied to scene interpretation, there are three sets of formulae, (i) generic knowledge about the world, (ii) knowledge about a specific scene in terms of visual evidence and context, and (iii) propositions which are generated as the scene interpretation. Model construction means connecting constant, predicate and function symbols of the formulae with corresponding individuals, predicates and functions of a real-world domain. The fact that the third set of formulae, the scene interpretation, is not given but incrementally constructed, is one of the differences to the notion of interpretation as used in formal knowledge representation.

The constructed model is "partial" in that neither all possible nor all implied conceptualisations of the scene must be expressed as formulae, and in particular that image analysis must not be perfect.

In addition to these general properties of a model, Schröder [15] postulates that two requirements must be fulfilled. First, it must be possible to extend the partial model to a complete model. This ensures consistency of any scene interpretation since it is always part of a model. Second, disjunctions must be resolved. This ensures completeness with respect to specialisation.

It is interesting to transfer Schröder's criteria for scene interpretation into the conceptual framework introduced above, although this is not (yet) formulated in a precise logical language. Scene interpretation as outlined in Sections 3.1 and 3.2 is an interpretation in the logical sense, i.e. the scene interpretation process determines a mapping from symbolic expressions into the real world, by connecting symbolic constants to individual entities in the scene via sensory input and computational procedures. For example, instantiating a place-cover aggregate connects the corresponding formula with the real-world scene via spatio-temporal constraints and whatever visual evidence is related to the occurrence.

The mapping is a model, if it causes all symbolic expressions of the conceptual knowledge and the scene-specific knowledge to become true. For example, if a plate is on the table in the scene, then a corresponding symbolic relation ON should hold for symbolic tokens PLATE1 and TABLE1 assigned to the scene objects. This is the case if the real-world meaning is correctly represented by the computational procedure which determines the ON-relation for the scene object.

Sometimes, our intuitive notions may differ from what is being computed and one might argue that in those cases a vision system does not compute a model. Discrepancies may range from obvious mistakes (e.g. interpreting a shadow as a physical object) to disputable propositions where even people might disagree (e.g. calling a spatial relation "near"). For a formal analysis, it is therefore useful to avoid references to intuition and accept the operational semantics realised by the conceptual models and computational procedures. In this sense consistent scene interpretations always correspond to logical models.

Schröder's consistency requirement makes sure that a partial model is always the kernel of a potentially complete model. In our framework, requirements for scene interpretations have been introduced without this condition, and it is not apparent at this stage how one could ensure that a partial scene interpretation remains consistent if it is completed by further image analysis. As an example, imagine a scene where a plate is placed onto an empty table. The vision system may come up with the interpretation of "table-laying" including predictions about future actions. The continuation of the scene, however, may show that the plate is picked up again

and put elsewhere. Hence the premature interpretation cannot be completed to be consistent with the scene.

In view of the fact that visual evidence is ambiguous as a rule (and not as an exception), we expect that Schröder's consistency requirement cannot be met in practice. Rather, we must be prepared to (i) withdraw an interpretation if it becomes inconsistent with additional information, and (ii) provide guiding information which helps to select between multiple possible models.

Let us now consider Schröder's specialisation requirement which calls for interpretations without unresolved disjunctions. Disjunctions occur naturally in conceptual descriptions where choices are left open, for example, when a concept may be specialised further according to the taxonomy, or when a property may have several values. Requiring interpretations without disjunctions is equivalent to enforcing interpretations at the lowest possible abstraction level. This is clearly not the right answer for all vision tasks and pragmatic contexts which one can think of. For example, in an obstacle avoidance task the vision system could well do without the most-specific classification of obstacles as long as their geometry is recognised properly.

In summary, we see that model construction, although the right logical framework for scene interpretation, leaves several questions unanswered regarding a practically useful interpretation process. These questions will be brought up again when we examine DLs for possible interpretation services, and will also be addressed in Section 6.

4. Knowledge representation and reasoning with description logics

Description logics (DLs), also called terminological logics, originated from the work of several researchers who tried to replace the intuitive semantics of semantic networks and frame systems by a formal logic-based semantics [27–30]. It was soon realised that semantic networks and frames do not require full first-order logic, but fragments suffice for typical representation and reasoning tasks. Moreover, since inference problems are found to be decidable in these fragments, reasoning can be operationalised by sound, complete, and terminating algorithms. This is a clear advantage compared to theorem provers for full first-order logic or theorem provers for Horn clauses with function symbols (e.g. PROLOG). DLs have taken a remarkable development as both solid theoretical foundations and successful operational systems have been achieved. The interest of using DL systems for practical applications is due to several attractive aspects.

- The family of DLs comprises a variety of representation languages ranging from languages with polynomial complexity such as CLASSIC [31] to highly expressive languages which – in the worst case – are no longer polynomial, such as $\mathrm{SHIQ}(\mathrm{D}_n)^-$ [32,33].
- DL systems offer various kinds of inference services which can be used for application development. Systems are available off the shelf and are based on international standards for web-based system development (e.g. OWL [34]). An excellent presentation of the history and current state of DL technology is offered in [13]. One example for a current DL system is RACER [35]. RACER supports the logic $\mathrm{SHIQ}(\mathrm{D}_n)^-$ and provides extensive support for OWL.
- The representation language is object-based and supports frame-like representations.

For the purpose of this contribution it is useful to introduce DLs in terms of a repertoire of language features which are potentially important for scene interpretation, rather than focussing on particular DLs. In Section 5, we will then examine how to meet the knowledge-representation requirements for scene interpretation. Unfortunately, not all features of the repertoire can be combined in a single language without losing decidability, so a careful analysis is necessary and a restricted use may be imposed. Note that decidability was not an issue in related work such as [36].

4.1. Syntax and semantics of description logics

Knowledge representation in DLs is based on unary predicates called concepts (or concept terms), binary predicates called roles (or role terms), and so-called individuals. A concept is interpreted in a Tarski-style set-theoretical semantics as a set of elements from a domain of discourse (also called universe), a role is interpreted as a set of pairs of elements from the domain, and an individual denotes an element of the domain. The elements in the second position of a role pair are called role fillers. Functional roles which map each first argument into at most one role filler are called features.

Building Blocks. For each application one has to fix a set of concept names (so-called atomic concepts), a set of role names (also called atomic roles), and a set of individuals. Names can be used to build complex concept and role terms. This is accomplished with the help of operators whose meaning is precisely defined in terms of the set-theoretical semantics. Below we present the language for building complex concept terms. We rely on a notation with the following abbreviations (possibly used with index):

C	concept term
R	role term
F	feature term
I	individual
CN	concept name
RN	role name
n	natural number

In the following we introduce concept terms that have been analysed in the description logic literature. Later on we focus on specific description logics such as CLASSIC or SHIQ.

C → CN	concept name
top	universal concept (containing all other concepts)
bottom	empty concept
(not C)	negation of a concept
(and $C_1 \ldots C_n$)	intersection of concepts
(or $C_1 \ldots C_n$)	union of concepts
(some R C)	existential quantification
(all R C)	value restriction
(at-least n R C)	qualified at-least number restriction
(at-most n R C)	qualified at-most number restriction
(exactly n R C)	qualified exact number restriction
(same-as F_1 F_2)	feature (chain) agreement
(subset R_1 R_2)	role-value map
(one-of $I_1 \ldots I_n$)	singleton set

For the roles used in qualified number restrictions, additional restrictions apply: they must be neither transitive nor must there exist a transitive subrole (see below). Role terms may be formed as follows:

R → RN	role name
top	universal role (containing all other roles)
bottom	empty role
(inverse R)	inverse role
(and $R_1 \ldots R_n$)	intersection of roles
(or $R_1 \ldots R_n$)	union of roles
(compose $F_1 \ldots F_n$)	feature chain
(compose $R_1 \ldots R_n$)	role composition

The concept expressions involving roles may require some explanations. The value restriction (all R C) denotes a class of objects where all role fillers of R, if there are any, belong to the concept C. Hence (and plate (all has-shape oval)) describes plates whose shapes are oval (but specific instances of oval shapes are not necessarily known). To express that a candlestick must have at-least one candle, one can use the existential role restriction (and candlestick (some has-candle candle)). Several forms of number restrictions can be used to further restrict the role-fillers for a class of objects. For example (and candlestick (at-least 2 has-candle candle)(at-most 2 has-candle candle)), describes the candlesticks with exactly two candles.

With so-called feature (chain) agreements one can describe elements of the domain which possess the same fillers for (possibly different) feature chains. Consider the definition of a cover which requires that plate and saucer have the same colour. This restriction could be expressed as

```
(same-as    (compose has-plate has-colour)
            (compose has-saucer has-colour))
```

Feature chain agreement is one of those constructs which cannot be combined with other critical constructs without jeopardising decidability. In particular, feature chain agreement is part of the CLASSIC language, however it cannot be used in a language as expressive as $\mathrm{SHIQ}(\mathrm{D}_n)^-$ [37].

Another critical construct not supported in $\mathrm{SHIQ}(\mathrm{D}_n)^-$ is a role-value map (subset with role chains). In general, this construct cannot be integrated even into the (less expressive) CLASSIC language without losing decidability [38]. But as can be seen from the previous example and some other examples shown below, both constructs appear to play a natural role in human concept formation.

$\mathrm{SHIQ}(\mathrm{D}_n)^-$ is an example of a DL language that does not only support the description of abstract objects (in the universe) but also supports additional domains with objects for which, for instance, an order is defined and certain algebraic operators (functions) such as addition and multiplication are specified. An additional domain plus a set of predicates syntactically constructed with reference to a set of predefined operators is called a concrete domain.

Concrete domains were introduced with the language ALC(D) [39]. The (D) part stands for concrete domains. The language ALC [40] comprises the first eight concept constructors from the grammar shown above. Another important extension of DLs in terms of predicates over concrete domains was established by Baader and Hanschke [37] with the language ALCFP(D) (i.e. ALC with feature agreements (same-as), feature composition, and concrete domains). The integration of concrete domain predicates allows to include predicates which are evaluated outside the description logic reasoner. Examples of concrete domain predicates interesting for scene interpretation are inequalities over real numbers, Allen's interval calculus [41], or the RCC-8 calculus about spatial regions [42].

At the time of writing RACER is the only optimised DL system which supports concrete domains with the language $\mathrm{SHIQ}(\mathrm{D}_n)^-$. In particular, the RACER concept language offers operators for forming concepts based on predicates involving (in)equalities over the integers and the reals. The following shows the syntax for concrete-domain concept expressions (CDCs) which extend the list of concept terms presented earlier (symbols on the left-hand side of grammar rules are treated as nonterminals as usual). AN denotes an attribute name which specifies an integer- or real-valued variable.

```
CDC    → (a AN) (an AN)          attribute filler exists restriction
         (no AN)                 complement of attribute filler exists restriction
         (min AN integer)        integer predicate exists restriction
         (max AN integer)
         (equal AN integer)
         (> aexpr aexpr)         real predicate exists restriction
         (⩾ aexpr aexpr)
         (< aexpr aexpr)
         (⩽ aexpr aexpr)
         (= aexpr aexpr)
aexpr  → AN
         real
         (+ aexpr2)              sum expression
         aexprl
aexprl → AN
         real
         (* real AN)             product expression
aexpr2 → aexprl
aexpr2 → aexprl aexpr2
```

It can be seen that concrete domain predicates offer an interesting way to integrate quantitative data from low-level vision with symbolic reasoning in high-level vision. As an example, we could define an integer-valued attribute `size` for the number of pixels of a plate-view and express a conceptual restriction on the size of the plate-view by means of the concept expression:

```
(and (min size 13) (max size 20))
```

Conceptual knowledge. The language for building concept terms as introduced above can be used to describe subsets of the universe. Concept definitions and logical relationships between concepts are introduced by so-called terminological axioms. The general forms of terminological axioms are given as follows (for definitions, C_1 is a concept name):

`(equivalent` C_1 C_2`)` (identity relationship between the sets associated with C_1 and C_2)
`(implies` C_1 C_2`)` (subset relationship between the sets associated with C_1 and C_2)
`(disjoint` $C_1 \ldots C_n$`)` (the sets associated with $C_1 \ldots C_n$ are disjoint)

Similar to concept definitions, relationships between roles can be enforced:

`(equivalent` R_1 R_2`)`
`(implies` R_1 R_2`)`

In addition, in the language $\mathrm{SHIQ}(D_n)^-$ roles may be declared to be functional or have other properties such as transitivity or symmetry. For historical reasons, a set of axioms is referred to as a TBox (terminological box).

It is apparent that *n*-ary predicates (or in set terminology: *n*-ary relations) cannot be directly represented. However, there is a well known way around by reifying *n*-tuples. Let

$$R \subseteq C_1 \times C_2 \times \ldots \times C_n$$

be an *n*-ary relation. Define C as the set of all *n*-tuples of R, and R_i as the binary relation between an *n*-tuple and its *i*th component.

$$R_i \subseteq C \times C_i, \quad i = 1 \ldots n$$

The concepts C and $C_1 \ldots C_n$ together with the roles $R_1 \ldots R_n$ represent the *n*-ary relation R. The disadvantage of using reified relations of this kind is that it is possible to introduce multiple names (aggregate individuals) for the same tuple. In the following we will see that this problem can be pragmatically solved in our context. Reification will be used extensively for defining concepts for high-level scene interpretation which typically relate many components to each other.

Assertional knowledge: So far, we have presented constructs for representing conceptual knowledge in a TBox. DL syntax also includes constructs for representing factual (assertional) knowledge about individuals. This body of knowledge is called an ABox. Let IN, IN1, and IN2 be individual names, then the following constructs express concept membership and role membership, respectively:

```
(instance IN C)        IN is instance of C
(related IN1 IN2 R)    IN1 is related to IN2 via role R
```

The following ABox constructs are provided for concrete domain extensions:

```
(constrained IN AN ON)
```

A concrete domain object ON is the filler for an attribute AN with respect to an individual IN.

`(constraints` *constraint-expr1…constraint-exprN*`)`

Constraint expressions describe relationships between objects of a concrete domain.

A knowledge base is a pair of TBox and ABox. Practical systems such as RACER support multiple knowledge bases. In particular, one TBox can be referred to by multiple ABoxes. ABoxes will be used for representing possibly different interpretation results (see below).

4.2. Reasoning services of description logics

In addition to providing the framework for knowledge bases, a DL system offers specific kinds of reasoning services. They are logical inferences based on the formal semantics, similar to inferences in first-order predicate

logic. From an application-oriented point of view, the reasoning services are useful for two main purposes, (i) organising and maintaining a potentially large knowledge base, and (ii) providing complete and correct procedures as building blocks for application systems.

Typical reasoning services of a DL system determine

- whether a concept is satisfiable (i.e. consistent),
- whether a concept is subsumed by another concept,
- whether two concepts are disjoint,
- whether a TBox is coherent (i.e. contains no inconsistent concept names),
- what are the parents (children) of a concept,
- whether an ABox is consistent w.r.t. a TBox,
- whether an individual is an instance of a concept,
- what are the most-specific atomic concepts of which an individual is an instance,
- what are the instances of a concept,
- what are the individuals filling a role for a specified individual,
- what pairs of individuals are related by a specified role
- answers for general queries about tuples of individuals (mentioned in ABoxes) that satisfy certain predicates (so-called conjunctive queries).

It can be shown that, in general, all of these services can be reduced to consistency checking of an ABox w.r.t. a TBox. Hence, in implemented DL systems, a premium is on efficient and optimised algorithms for consistency checking. One way to do this is by model construction as this is an elegant way to define an algorithm for proving satisfiability. Many DL systems are based on model construction techniques (they use so-called tableau provers). This is interesting because model construction has been shown to be one of the building blocks for the logical paraphrase of scene interpretation (Section 3.3).

Usually, inference services of DL systems are based on the open-world assumption (OWA) as opposed to the closed-world assumption (CWA). Employing the CWA means that if a fact does not follow from a knowledge base, then the negation is assumed to hold. As a consequence of the OWA in DL systems, inferences are only drawn to the extent that they are not affected by additional information. This precludes intuitive inferences which might be useful for scene interpretation. For example, if there is evidence for two dinner covers on a table, the interpretation of a ''dinner-for-two'' cannot be logically inferred as additional covers may be added to the knowledge base. Some form of closed-world reasoning was already supported in the CLASSIC system [31] (see also [43] for a theoretical investigation of the approach taken in CLASSIC). However, CLASSIC does not offer an expressive query language for finding individuals based on closed-world assumptions. The DL system RACER indeed now supports a very expressive query language for ABoxes (conjunctive queries) that also provides CWA-based operators (see below for the use of queries to formalise image interpretation).

In the DL literature, there are also investigations of so-called non-standard inference services which have been intoduced mainly in support of knowledge engineering, for example providing normalised forms for concept definitions. Some of the non-standard inferences may also be interesting for scene interpretation, for example, the generalisation operation LCS which computes the most-specific concept subsuming several specified concepts [44]. However, due to space restrictions we cannot report on details here.

5. Scene interpretation with description logics

We now examine in detail how scene interpretation – according to the ideas and requirements put forth in the previous sections – can be supported by knowledge representation and reasoning with a DL system. We will first deal with representational requirements and then with the interpretation process.

5.1. Representing aggregates with DL concepts

The main representational unit which has been identified for conceptual knowledge representation is an aggregate. An aggregate expresses the properties and constraints which make a particular set of objects worth being recognised as a whole. As shown in Section 2, aggregates can be described informally by frames, and it is straightforward to translate basic frame notation into DL notation: slot identifiers become role names, concept expressions for slot values become role-value restrictions, and the whole frame is represented as a union of role restrictions.

The assignment of role names deserves some consideration. One might be tempted to represent all roles connecting an aggregate to parts with a single role type ''has-part'' (or some other standard name). This would ignore that, in general, parts ''play different roles'' in an aggregate, and unwanted inheritance relations may result if these roles are not distinguished. It is useful to think of an aggregate as a reified n-ary relation where the roles relate components to corresponding positions in the n-tuples, as pointed out in Section 4.1. Hence role names within an aggregate should in general be distinct.

On the other hand, there may be aggregates related to one another by specialisation, for example ''cover'' and ''breakfast-cover''. Here, parts in different aggregates could play identical roles and should have identical names so that the specialisation relation between ''cover'' and ''breakfast-cover'' can be deduced.

There exists a considerable body of research on reasoning with part–whole relations, see [45] for an overview. The first integration of part–whole reasoning in a DL is reported in [46] where CLASSIC is extended to include ''physical whole–part relations'' between a composed object (such as a stereo system) and its components. Whole–part (and part–whole) relations enjoy a special status and are used to induce transitive contains and may-contain relations as well as several special inference mechanisms. Our approach also

exploits the special status of roles relating parts to aggregates when a part–whole interpretation step is performed (see Section 5.2).

Part–whole relationships differentiated according to distinct mereological categories (such as component – composite, stuff – object) and their realisability in DLs are investigated in [47] and [48]. Sattler shows that various kinds of part–whole relationships can be modeled in SHIQ, but that there are properties (such as part–whole inheritance) which cannot be achieved in full generality without losing decidability. In our approach any special part–whole semantics are modeled based on the specific roles introduced in the aggregates. In particular, one can achieve that a certain part cannot be shared among two different aggregates by making the respective roles subroles of the role `has-exclusive-part` and constraining this role as follows:

```
(all has-exclusive-part (at-most 1 (inverse has-exclusive-part)))
```

Thus, we need number restrictions and inverse roles for expressing exclusive use of parts in compositions. If roles of aggregates are not declared subroles of `has-exclusive-part` they allow for part sharing as in the coffee pot example mentioned above.

In order to function within a vision system, individuals in the ABox of a DL system must interface to lower-level vision. Mechanisms to feed concrete data into the ABox are common-place for DL applications, so this is no serious challenge. In the framework presented in Section 2, lower-level processes will supply data for instances of view concepts which are modelled as parts of scene objects. Also, context information may be entered into the ABox in terms of instantiated aggregates.

Representing the constraints section of aggregates is a more difficult issue. In the following simple example a DL concept is defined for a cover consisting of a plate, a saucer near the plate, and a cup on the saucer.

The requirement that the saucer is located near the same plate as referred to by the role cv-pl is expressed by the `subset` construct which relates the filler of the role cv-pl and the filler of the role chain (compose cv-sc near). The requirement that the cup is located on the same saucer as referred to by the role cv-sc is expressed in a similar way.

While `same-as` can be added to inexpressive description logics such as CLASSIC or even ALCFP(D), it cannot be added to SHIQ without, in general, losing decidability of main inference problems. However, considering the example discussed in Fig. 7, it becomes clear that, due to the nature of the things to be modelled, `same-as` is not adequate and `subset` must be used in the domain model. This does not only hold for spatial constraints but also for temporal constraints as the next example demonstrates.

```
(equivalent cover
    (and configuration
        (exactly 1 cv-pl plate)
        (exactly 1 cv-sc (and saucer (some near plate)))
        (exactly 1 cv-cp (and cup (some on saucer)))
        (subset cv-pl (compose cv-sc near))
        (subset cv-sc (compose cv-cp on))))
```

Fig. 7. DL concept for a simple cover.

A special task of the constraint section of an aggregate is to express spatial and temporal constraints. In principle, this could be done in a manner similar to the example in Fig. 7 where the symbolic roles "near" and "on" do the job. For example, in a (simplified) place-cover aggregate one could express the temporal "before" relation between the place-saucer and the place-cup occurrences as follows:

Note that in description logics one can define concepts (in the TBox) but one cannot define relations using axioms. Some properties of spatial relations such as the transitivity of the relation `inside` are not reflected by just using simple role names. This may or may not be a problem in applications. In any case, a concept definition as in Fig. 8 implies that qualitative temporal and spatial relations needed for conceptual modelling (such as "on" or "before") must be instantiated bottom-up by processes outside of the DL system. Assuming separate control structures of high-level and low-level processes, this would lead to bottom-up computation of a potentially very large number of pairwise spatial and temporal relations, from which only a small number may play a part in a high-level interpretation.

By integrating quantitative computations into the high-level concepts, a more efficient and also more transparent solution may be achieved. This can be made possible by concrete-domain concept terms as introduced in Section 4.1.

Four temporal constraints are specified:

(i) The end of the `place-saucer` occurrence must be before the end of the place-cup occurrence.
(ii) The begin of the `place-cover` occurrence is the minimum of the begins of its constituent occurrences.
(iii) The end of the `place-cover` occurrence is the maximum of the ends of its constituent occurrences.
(iv) The overall duration must not exceed a given maximal duration.

The constraints involve attributes relating an occurrence to its begin and end time, expressed in terms of values of the concrete domain of integers. Different from the first formulation with qualitative roles, the content of the constraints is now part of high-level concepts. This opens up the way for flexible interpretation strategies where constraints are propagated in order to restrict possible instan-

```
(equivalent place-cover
    (and agent-activity
        (exactly 1 pc-tp1 (and transport (some tp-obj plate))
        (exactly 1 pc-tp2 (and transport
                          (some tp-obj saucer)
                          (some before (and transport (some tp-obj cup)))
                          (exactly 1 pc-tp3 (and transport (some tp-obj cup))
        (subset pc-tp3 (compose pc-tp2 before))))
```

Fig. 8. Simplified DL concept for place-cover.

tiations at choice points. In particular, constraints pertaining to hypothesised objects without visual evidence can be used to constrain lower-level processes. For example, if evidence for a plate has led to instantiating a cover, spatial constraints between plate and missing cover parts, such as cup and saucer, can be exploited for top-down guided image analysis at the constrained locations. Our approach differs from temporal or spatial logic approaches in that it does not attempt to integrate inherent properties of space and time into the symbolic realm, but rather exploits the computational facilities of a metric space. The need for a metric space between signal and symbol processing has also been pointed out in [49].

Note that the minim and maxim operators are not part of a regular DL syntax. But the intended semantics can also be expressed by a disjunction of inequalities between pairs of variables.

Analysing the representation constructs that are needed for this kind of modelling approach, there is bad news again. The example discussed in Fig. 9 requires role composition (compose). This cannot be offered in a DL with expressive concrete domains (such as the one used in the example) and with expressive concept constructors. Indeed, role composition is excluded from the DL SHIQ$(D_n)^-$ supported by RACER for decidability reasons (this is what the minus sign actually indicates).

In summary, we have shown that the basic structure of an aggregate as introduced in Section 2 can, in principle, be modelled by a DL system using the following scheme:

```
(equivalent <concept-name>
  (and <parent-conceptl> ... <parent-conceptN>
    (<number-restrictionl>  <role-namel>  <part-conceptl>)
    ...
    (<number-restrictionK>  <role-nameK>  <part-conceptK>)
    <constraints between parts>))
```

```
(equivalent place-cover
        (and agent-activity
            (exactly 1 pc-tp1 (and transport (some tp-obj plate))
            (exactly 1 pc-tp2 (and transport (some tp-obj saucer))
            (exactly 1 pc-tp3 (and transport (some tp-obj cup))
            (<= (compose pc-tp2 tp-end) (compose pc-tp3 tp-end))
            (= pc-beg (minim (compose pc-tp1 tp-beg)
                             (compose pc-tp2 tp-beg)
                             (compose pc-tp3 tp-beg)))
            (= pc-end (maxim (compose pc-tp1 tp-end)
                             (compose pc-tp2 tp-end)
                             (compose pc-tp3 tp-end)))
            (<= (- pc-end pc-beg) max-duration))))
```

Fig. 9. DL concept of place-cover with temporal constraints.

Currently, at the TBox level, the expressivity of DL systems cannot be made as high as required to allow for concise and intuitive formulations of constraints between parts. The problems are due to decidability problems in the general case. Note that in the DL community representation problems such as the ones discussed above are deemed special cases, and developing the corresponding decidability proofs and optimised implementations is usually considered too much work.

It might be possible to consider a combination of Temporal Logics and DL for expressing temporal constraints (see, for instance [50]). A careful analysis is required to determine if adequate expressivity can be achieved without losing decidability. Unfortunately, practical systems for temporal description logics seem to be out of reach at the current state of the art. Note that some properties of temporal constraints such as transitivity are indeed reflected by the approach proposed here due to the transitivity of the $\leqslant$ predicate used in concrete-domain expressions (see Fig. 9).

So if a reduction to decidable inference problems implemented in practical inference systems such as RACER is to be used, expressivity w.r.t. constraints between parts must be reduced and, hence, TBox axioms will be "too weak" in a sense. Additional representation constructs must be used instead to express required constraints. In subsequent sections we will explain how the expressive RACER query (and rule) language allows us to cope with this situation appropriately. Before this can be explained, however, we

consider how scene interpretation processes can be modelled using standard inference services as explained above.

5.2. Supporting the scene interpretation process with a DL system

Support of the interpretation process has already been an important aspect for choosing particular constraint representations in the previous subsection. We now examine in more generality how the interpretation process can be supported by reasoning services of a DL system. As pointed out earlier, the use of DL reasoning services would offer two main advantages:

1. The formal semantics of a DL language helps to avoid misunderstandings often arising if knowledge bases and inference procedures are constructed intuitively.
2. Correct inference procedures may obviate the need for developing parts of application-specific programs.

Looking at the list of services presented in Section 4.2 we see that the first group deals with concept terms only and is mainly useful for the construction and maintenance of a knowledge base. The key inference service of this group is a satisfiability test from which all other concept-related services can be derived, for example concept subsumption which tests whether one concept is more general than another, and concept classification which determines the parent concepts for a given concept term.

The second group deals with ABoxes and TBoxes together and hence is more directly relevant for scene interpretation. It should be clear from the preceding that the TBox of a DL takes the role of the conceptual knowledge base and the ABox of a container for concrete scene data. Referring to Fig. 2, the ABox contains (i) visual evidence in terms of the GSD, (ii) context information in terms of partially specified concept instances, and (iii) the high-level scene description generated by the interpretation process. A DL system always checks consistency of the ABox w.r.t. the TBox, hence the ABox formally corresponds to a (partial) model of the TBox and – given its role in the scene interpretation framework – is a (partial) scene interpretation. We conclude that DL consistency checking can be used to ensure consistent scene interpretations.

Another key inference service is the instance check which determines whether an individual is an instance of a given concept w.r.t. the current ABox and the TBox. The most-specific atomic concepts of which an individual is an instance can be derived by instance classification (which, internally, is based on instance checks). The set of most-specific atomic concepts computed by instance classification is also known as the set of direct types. If the direct types are computed for all individuals in advance, this is known as ABox realization.

At first glance, instance classification appears to be an inference service which is immediately applicable for scene interpretation. Given an image segment represented as an individual in an ABox, this service would deliver the most-specific concept applicable to this individual. But this will not work in general because of two main reasons:

(i) Scene interpretation (and image interpretation in general) cannot be solely modelled as deduction. It is well known that image evidence is generally not conclusive regarding a classification because of the many-to-one nature of the imaging process. Hence an inference service which infers a class membership cannot solve the full interpretation problem. As elaborated earlier, it appears to be more adequate to model image interpretation as a (logical) model-construction task.
(ii) Individuals do not yet exist for aggregates which must be discovered. Hence instance checking cannot be applied. As a work-around, tentative aggregate instants could be created. However, this would turn interpretation into a top-down trial-and-error procedure which cannot be efficient in general. However, if aggregate individuals are determined by part–whole reasoning as described below, they result from educated guesses based on parts, and a classification step could become obsolete in many cases.

We now turn to the interpretation steps identified in Section 3. The first kind is aggregate instantiation, also known as part–whole reasoning. Given an individual in an ABox, what are the possible aggregates supported by this individual, and which aggregate should be chosen first? Assuming that aggregates are modelled by DL concepts as explicated above, we can exploit the special syntax of aggregate concept definitions which allows to identify parts by specific roles. The idea behind this syntactic construction is to provide a way for distinguishing roles which model spatio-temporal co-occurrence which are typical for the parts of an aggregate. This way we can identify the concept terms which describe the respective role fillers, and what remains to be done for part–whole reasoning is instance checking of the individual against each concept term. This can be done with a readily available reasoning service. A concrete solution for part–whole reasoning in RACER will be presented in the next subsection.

However, no support can be given for the strategic decision which aggregate – out of possibly many candidates – should be tried first. This requires a preference measure which is outside the scope of current DL systems. It must be expected that uneducated choices will lead to backtracking and hence inefficiency of the interpretation process. The development of a preference measure for part–whole reasoning must be considered a prerequisite for the employment of DL systems in practical scene interpretation applications.

The second kind of interpretation step required for scene interpretation is instance specialisation. One of the main advantages of a DL system is the specialisation network automatically generated for all concept definitions. Hence all specialisations of a given (atomic) concept can be efficiently retrieved. To compute the possible specialisations

of an individual, the most-specific atomic concepts of which an individual is an instance (i.e. the so-called direct types) can be determined by a service called instance classification, and then more specific concepts can be found by consulting the specialisation hierarchy. In general, there will be alternative choices, and it is useful to have guidance for a "best" choice. As with part–whole reasoning, such guidance is outside the scope of current DL systems.

Instance expansion is a step applied to instantiated aggregates and causing its parts to be instantiated. This operation is completely determined by the concept definition of the aggregate, and extending an existing DL system to include this new service should be possible without serious problems.

The fourth kind of interpretation step needed for scene interpretation is instance merging. As pointed out earlier, this step is typically required when a top-down generated hypothetical instance has to be connected with bottom-up evidence. Formally, the reasoning service required here is to determine whether it is consistent with the TBox and the current ABox to unify the descriptions of two individuals. Unification requires specialising the role fillers of the individuals until the most general common representation is found. This must be applied recursively to the instances of parts of aggregates and terminates at the level of instances of primitive concepts.

As observed for other kinds of interpretation steps, DL systems do not offer guidance when alternative choices are possible and an order of preference becomes important.

In summary, the logical structure of DL concepts can be exploited for selecting interpretation steps which necessarily lead to a logical model, i.e. to a description consistent with visual evidence, context and conceptual knowledge. This can be a considerable benefit. But in general, there are many models, and degrees of freedom are left open regarding choices among alternatives. The decisive question is which model to prefer in the face of several possiblities. From an answer to this question one can expect criteria regarding the preferred order for interpretation steps and other choices. Our understanding of vision suggests that these choices are critical for a practically useful performance of vision systems. In Section 6 we will sketch a probabilistic approach for determining preferred interpretations.

5.3. Scene interpretation using RACER's query and rule language

In previous sections we have discussed how knowledge necessary for scene interpretation could be modelled using TBoxes and ABoxes of a DL. One important insight was that the TBox language provided by current DL systems such as RACER appears to be "too weak" as important constructs such as feature chains, subset, and same-as are not provided. In this section we show how we can compensate for the deficiencies using a sophisticated query language for ABox individuals. Recently, the RACER query language nRQL (new RACER Query Language) has been developed [51]. It provides an extension to existing ABox query services in terms of query expressions with variables. In the following we will show how nRQL can be conveniently used to support the scene interpretation process.

The retrieval operator of nRQL has the general format

```
(retrieve <list-of-objects>  <query-body >)
```

where the list-of-objects may contain variables (beginning with "?") and individuals. The query-body is essentially a boolean combination of possible ABox entries with individuals replaced by variables, augmented by some additional constructs. A query can be seen as a template which is applied to the ABox and delivers all variable bindings satisfying the template.

As an example for the use of nRQL in our image interpretation scenario, let us assume that the current ABox contains various plates, cups and saucers. The following query will retrieve all combinations of parts which satisfy the aggregate definition of a cover given in Fig. 7.

```
(retrieve (?x ?y ?z) (and  (?x plate)
                           (?y saucer)
                           (?z cup)
                           (?x ?y near)
                           (?z ?y on)))
```

Note that the same-as relation can be expressed by using the same variable name. The result of the query is a list of all possible bindings of the variables to individuals of the ABox. For the fictitious ABox of this example, the result could be

```
(((?x plate1) (?y saucer3) (?z cup2))
((?x plate4) (?y saucer2) (?z cup4)))
```

indicating two combinations of plate, saucer and cup which satisfy the constraints of the cover definition.

This opens up an interesting way to support part–whole reasoning for scene interpretation. The query mechanism can be used to efficiently retrieve combinations of ABox individuals which justify the assertion of an aggregate instance. Furthermore, such queries can be automatically generated from the aggregate definitions. To establish an aggregate for each set of bindings retrieved by the query, a new individual must be entered into the ABox as an instance of the aggregate concept and related to the retrieved individuals via the roles of the aggregate concept. For the first set of bindings shown above, the new ABox entries would be:

```
(instance cover1 cover)
(related cover1 plate1 cv-pl)
(related cover1 saucer3 cv-sc)
(related cover1 cup2 cv-cp)
```

As a convenient service of the DL system, the new individual `cover1` will be automatically classified w.r.t.

all TBox concepts and implicit subsumption by other concepts – e.g. by specialisations of cover – will be discovered. It is obvious to see that with this approach there will be just one representative for each tuple of bindings, and hence, possible problems with reification can be avoided at a pragmatical level.

RACER also offers a simple rule language. Given a certain query as the antecedent of a rule, additional constraints with respect to existing ABox individuals can be automatically asserted. Asserting constraints to existing ABox individuals is the key to compensate for the restricted expressivity in TBoxes (see the analysis presented above). Since we do not deal with infinite structures but a finite number of explicitly mentioned individuals in an ABox to which rules are applied, the rule language can support variables, and role composition as in Fig. 9 is no problem. We only provide a sketch here due to space limitations (see also the nRQL manual for details about firerule).

```
(firerule   (and(?x place-cover)
                (?x ?y pc-tp2)
                (?x ?z pc-tp3))
            (constraints
                (⩽ (tp-end ?y) (tp-end ?z))
                 ...
                (⩽ (- (pc-end ?x) (pc-beg ?x)) max-duration)))
```

The control strategy must ensure that rules do not trigger the introduction of new aggregates but just add constraints (which may rule out some hypothetical ABox due to unsatisfiability).

In order to be able to assert aggregate instances also in cases of partial evidence, it is necessary to provide "partial queries" for subsets of parts, in addition to the "complete query" for all parts of the aggregate. For the cover in our example, one could generate queries involving any two of the three parts of a cover. For aggregates with many parts, the number of possible queries could become very large, however, and additional considerations are required to control query invocation. This points to the need of a preference measure based on the expected success of a query. This is the subject of the next section.

6. Preferred models for scene interpretation

It has been shown at several points in the previous sections that stepwise interpretation needs guidance for selecting the most "plausible" or preferred partial interpretation among alternatives. In AI, various approaches have been developed to augment the knowledge base with preference rules of some sort [3]. In earlier work we have explored extensions of DLs using default rules [52]. The main drawback of rule-based approaches is the need to handcraft the rules, so it is worthwhile to look for preference measures which can be generated from general principles or can possibly be learnt. This has led us to investigate probabilistic approaches and ways to combine probabilistic information with a structured knowledge base.

There exist several approaches for merging DLs and probabilities, motivated partly by interest to fuzzify crisp concept membership [53], partly by the goal to replace Bayesian Network nodes by structured expressions [54–56]. Our interest is to embed aggregates and their parts in a Bayesian Network while maintaining the crisp logical framework, so the latter category of work is relevant for us.

Our basic idea is to compare alternative interpretation steps by the probabilities of the resulting (partial) interpretations given current evidence, and to choose the interpretation step which maximises this probability. Intuitively, the probability of a particular scene follows from statistics about scenes in a given domain, and it is not implausible to assume that such statistics can be obtained, at least qualitatively. For example, the statistics would tell that in a table-laying scene a saucer is more likely to be part of a cover than part of a candlestick. Similarly, typical locations of cutlery relative to a plate could be distinguished from less typical locations. Obviously, relating conceptual scene models to experiences also allows to investigate how these models could result from learning. This connection to a learning scenario is intended and first results on learning spatio-temporal aggregates are reported in [57].

Let us go one step further and assume that the cases giving rise to the statistics are available in a case-base. Then a partial interpretation can be viewed as a set of assertions

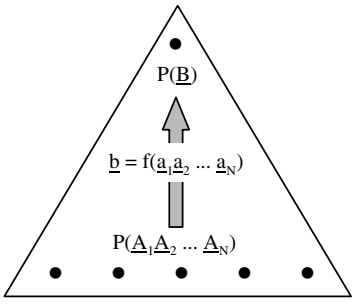

Fig. 10. Probabilistic structure of an aggregate. The upper node, described by a JPD P(B), represents export features. The lower nodes, described by a JPD P($A_1A_2 \ldots A_N$), represent import features of the parts. f is a deterministic mapping between import and export features.

which matches a subset of the cases in the case-base. Turned into a query of the RACER query language nRQL, the partial interpretation would retrieve this subset from the case-base. Hence, the probability of a partial interpretation can be viewed as the fraction of cases matching the interpretation. Furthermore, preferring an interpretation step which leads to a most probable interpretation means preferring the interpretation which is least restrictive regarding the number of remaining cases. Note that this is a strategy of least commitment.

Different from approaches which try to model the space of interpretations by a Bayesian Network [58,59] with aggregate nodes "causing" part nodes, we model a scene probabilistically at the level of primitive visual events provided by the GSD. Descriptions at higher abstraction levels are assigned probabilities according to the constituting primitive events. This motivates the following probabilistic structure for aggregates:

Each aggregate is described probabilistically in terms of a joint probability distribution (JPD) over part features ("import features") and a JPD over aggregate features ("export features") which are derived from the part features (Fig. 10). For example, the aggregate "cover" is described probabilistically by a JPD over part features such as location, size and colour, and a JPD over the export features of a cover, such as size and location of an enclosing rectangle.

The JPDs are fragments as each JPD only represents probabilities for the subspace of features for positive occurrences. So if the aggregate in Fig. 10 describes a cover, then P(B) is actually the fragment describing P(B, cover = yes). This is equivalent to specifying the prior P(cover = yes) and the conditional P(B| cover = yes). Similar structures have been proposed in [60] and [61].

There are no particular independency assumptions about part features within a single aggregate. However it is assumed that dependencies between different aggregates can be modelled exclusively with export features which then describe the aggregates as parts in a higher-level aggregate. For example, a "romantic-cover" could be defined as an aggregate consisting of a cover and a candlestick. Then it is assumed that the export features of cover suffice to model dependencies between the candlestick and all parts of the cover.

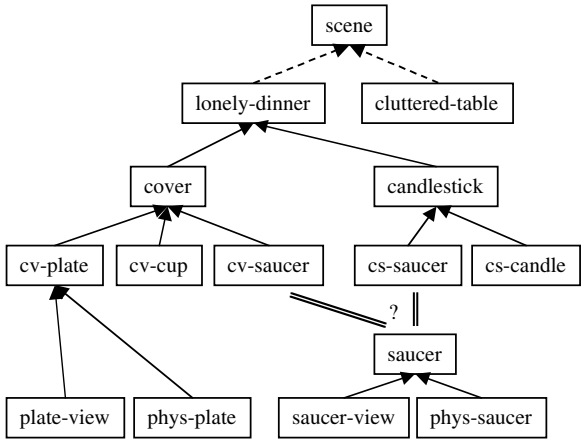

Fig. 11. Partial interpretation with two choices for assigning the saucer to an aggregate. Dotted lines denote specialisations, solid lines parts.

As a consequence, the probabilistic dependencies between aggregates remain tree-shaped when partial interpretations are constructed from several aggregates. Within an aggregate, however, the JPD may not always be representable by a tree-shaped Bayes Net, as typical dependencies in our table-setting scenario show. This increases the computational complexity, but it is limited by the number of parts and features combined in one aggregate.

To compute a measure of preference for an interpretation decision, for example of a part–whole-reasoning step, the probabilities of competing choices given evidence and context information are computed by a propagation algorithm similar to inferencing in a tree-shaped Bayesian Network [62] except of the structures within aggregates. It is beyond the scope of this contribution to present the inference procedure in detail. Instead, we will illustrate a typical preference computation by an example.

Consider a scene with a plate and a saucer as visual evidence, and context knowledge to the effect that a lonely-dinner table has been laid (Fig. 11). Let us assume that the current interpretation step is to assign the saucer either to the aggregate "cover" or the aggregate "candlestick". Hence the probabilities of the two alternatives must be compared:

$$P(\text{alt1}) = P(\text{cv-saucer} = \text{saucer} \mid \text{lonely-dinner} = \text{yes}, \text{plate-view}, \text{saucer-view})$$
$$P(\text{alt2}) = P(\text{cs-saucer} = \text{saucer} \mid \text{lonely-dinner} = \text{yes}, \text{plate-view}, \text{saucer-view})$$

Depending on the visual evidence, in particular on the locations of plate and saucer, and the JPD relating cover and candlestick in the aggregate cover, one alternative will be more likely than the other and determine the interpretation step.

Summarising this section, we have sketched a probabilistic inference scheme which provides preferences for choices left open by consistency-based interpretation. While probabilistic inferencing for scene interpretation has been proposed before, the new aspect in this research is the combination of probabilistic information with logic-based knowledge representation.

7. Conclusions and future research

We presented a conceptual framework for knowledge-based scene interpretation and examined how it could be realised with a DL system (Section 4). It has been shown that the the conceptual structure of multiple object occurrences, in particular temporal and spatial relations, can be represented using concept expressions and concrete domain constraints (Section 5.1). Interpretation results are encoded as satisfiable ABoxes (Section 5.2). ABoxes are generated by applying a set of generic construction operators using a cer-

tain control strategy (see Section 5.2). Queries for ABoxes are part of the control strategy for constructing hypotheses (Section 5.3). In addition we demonstrate that unwanted interpretations can be ruled out by applying a set of rules that, in addition to the TBox axioms, further restricts the set of satisfiable ABoxes (or possible interpretations).

Instead of extending the expressivity of a DL language for the sake of more expressive TBox definitions, we use a more expressive query language as a tool for scene interpretation. As the example of RACER's query language nRQL shows, feature chaining and other requirements for scene interpretation can be expressed intuitively and operationalised efficiently by a query or rule-based system which imposes additional constraints on ABox individuals. This opens up a way for accomplishing scene interpretation with the combined power of concept definitions and queries.

It has also been shown that the knowledge-based framework leaves several degrees of freedom regarding the selection of possible interpretations. A probabilistic approach has been sketched which provides guidance by preferring the most probable interpretation at choice points in the interpretation process. Further research on combining the probabilistic information with the conceptual units of the knowledge representation system is in progress.

The engineering-driven approach that we pursue in this article will ensure that practical experiments are possible with the RACER system in the very near future.

Acknowledgements

We gratefully acknowledge the collaboration with Volker Haarslev and Michael Wessel. This work was partly carried out in the EU project CogVis (IT-2000 29375) and the DFG project "Räumlich-terminologisches Schließen" (DFG Ne-279/8). Thanks also to the anonymous reviewers who gave thoughtful comments on this article.

References

[1] T. Kanade, Region segmentation: signal vs. semantics, in: Proc. Fourth Int. Joint Conf. on Pattern Recognition (IJCPR-78), 1978, pp. 95–105.
[2] M. Arens, H.-H. Nagel, Behavioural knowledge representation for the understanding and creation of video sequences, in: Proc. 26th German Conf. on Artificial Intelligence (KI-2003), LNCS, vol. 2821, Springer, 2003, pp. 149–163.
[3] S. Russell, P. Norvig, Artificial Intelligence – A Modern Approach, Prentice-Hall, 2003.
[4] N.J. Nilsson, Artificial Intelligence – A New Synthesis, Morgan Kaufman, San Francisco, 1998.
[5] M. Stefik, Introduction to Knowledge System, Morgan Kaufman, San Francisco, 1995.
[6] F. Baader, R. Küsters, Matching in description logics with existential restrictions, in: Proc. 7th Conf. on Principles of Knowledge Representation and Reasoning (KR-2000), 2000, pp. 261–272.
[7] L. Vila, A survey on temporal reasoning in artificial intelligence, AI Commun. 7 (1) (1994) 4–28.
[8] O. Stock (Ed.), Spatial and Temporal Reasoning, Kluwer, 1997.
[9] A. Cohn, S. Hazarika, Qualitative spatial representation and reasoning: an overview, Fundam. Inform. 1–2 (2001) 1–29.
[10] M. Haag, W. Theilmann, K. Schäfer, H.-H. Nagel, Integration of image sequence evaluation and fuzzy metric temporal logic programming, in: Advances in Artificial Intelligence (KI-97), LNCS, vol. 1303, Springer, 1997, pp. 301–312.
[11] H.-H. Nagel, From video to language – a detour via logic vs. jumping to conclusions, in: Proc. Speech and Image Understanding, IEEE Computer Society, 1999, pp. 79–99.
[12] A.G. Cohn, D. Magee, A. Galata, D. Hogg, S. Hazarika, Towards an architecture for cognitive vision using qualitative spatio-temporal representations and abduction, in: C. Freksa, W. Brauer, C. Habel, K.F. Wender (Eds.), Spatial Cognition III, Routes and Navigation, Human Memory and Learning, Spatial Representation and Spatial Learning, Springer, 2003, pp. 232–248.
[13] F. Baader, D. Calvanese, D. MacGuinness, D. Nardi, P. Patel-Schneider (Eds.), The Description Logic Handbook, Cambridge University Press, 2003.
[14] B. Neumann, Natural language description of time-varying scenes, in: D. Waltz (Ed.), Semantic Structures, Lawrence Erlbaum, 1989, pp. 167–206.
[15] C. Schröder, Bildinterpretation durch Modellkonstruktion: Eine Theorie zur rechnergestützten Analyse von Bildern, Dissertation, DISKI 196, infix, 1999.
[16] E. Di Sciascio, F.M. Donini, M. Mongiello, A description logic for image retrieval, in: E. Lamma, P. Mello (Eds.), Advances in Artificial Intelligence (AI*IA 99), LNCS, Springer, 1999.
[17] J. Barwise, J. Perry, Situations and Attitudes, Bradford, 1983.
[18] F. Baader, B. Hollunder, Embedding defaults into terminological representation systems, J. Autom. Reasoning 14 (1995) 149–180.
[19] R. Möller, B. Neumann, M. Wessel, Towards computer vision with description logics: some recent progress, in: IEEE Workshop Proceedings ICCV'99 International Workshop on Integrating Speech and Image Understanding, IEEE Computer Society, 1999, pp. 101–116.
[20] A. Blake, A. Yuille (Eds.), Active Vision, The MIT Press, Cambridge/MA and London/UK, 1992.
[21] R. Howarth, Interpreting a dynamic and uncertain world: high-level vision, in: Artificial Intelligence Review 9, 1995, pp. 37–63.
[22] B. Neumann, T. Weiss, Navigating through logic-based scene models for high-level scene interpretations, in: Proc. 3rd Int. Conf. on Computer Vision Systems (ICVS-2003), Springer, 2003, pp. 212–222.
[23] P.N. Johnson-Laird, Mental Models, Harvard University Press, 1983.
[24] R. Reiter, A. Mackworth, The Logic of Depiction, TR 87-23, Dept. Computer Science, Univ. of British Columbia, Vancouver, Canada, 1987.
[25] T. Matsuyama, V.S. Hwang, SIGMA – a knowledge-based aerial image understanding system, in: Advances in Computer Vision and Machine Intelligence, Plenum, 1990.
[26] C. Schröder, B. Neumann, On the logics of image interpretation: model-construction in a formal knowledge-representation framework, Proc. Int. Conf. on Image Processing (ICIP-96), vol. 2, IEEE Computer Society, 1996, pp. 785–788.
[27] W.A. Woods, What is in a link? Foundations for semantic networks, in: D.B. Bobrow, A. Collins (Eds.), Representation and Understanding: Studies in Cognitive Science, Academic Press, 1975, pp. 35–82.
[28] R.J. Brachman, On the epistemological status of semantic networks, in: N.V. Findler (Ed.), Associative Networks – Representation and Use of Knowledge by Computer, Academic Press, 1975, pp. 3–50.
[29] P.J. Hayes, The logic of frames, in: D. Metzing (Ed.), Frame Conception and Text Understanding, De Gruyter & Co., 1979, pp. 46–61.
[30] W.A. Woods, J.G. Schmolze, The KL-ONE family, in: F.W. Lehmann (Ed.), Semantic Networks in Artificial Intelligence, Pergamon, 1992, pp. 133–178.

[31] R. Brachman, D. McGuinness, P.F. Patel-Schneider, L.A. Resnick, A. Borgida, Living with CLASSIC: when and how to use a KL-ONE-like language, in: J. Sowa (Ed.), Principles of Semantic Networks, Springer, 1991, pp. 401–456.
[32] I. Horrocks, U. Sattler, S. Tobies, Reasoning with individuals for the description logic SHIQ, in: Proc. 17th Int. Conf. on Automated Deduction (CADE-17), LNCS, Springer, 2000.
[33] V. Haarslev, R. Möller, M. Wessel, The description logic ALCNHR+ extended with concrete domains: a practically motivated approach, in: Proc. Int. Joint Conf. on Automated Reasoning (IJCAR-2001), 2001, pp. 29–44.
[34] F. van Harmelen, J. Hendler, I. Horrocks, D.L. McGuinness, P.F. Patel-Schneider, L.A. Stein, OWL web ontology language reference http://www.w3.org/tr/owl-guide/, 2003.
[35] V. Haarslev, R. Möller, RACER system description, in: Proc. Int. Joint Conf. on Automated Reasoning (IJCAR 2001), LNAI, vol. 2083, Springer, 2001, pp. 701–705.
[36] T.A. Russ, R.M. MacGregor, B. Salemi, K. Price, R. Nevatia, Veil: combining semantic knowledge with image understanding, in: Proc. Image Understanding Workshop (IUW-96).
[37] F. Baader, P. Hanschke, Extensions of concept languages for a mechanical engineering application, in: Proc. 16th German Workshop on Artifcial Intelligence (GWAI-92), LNCS, vol. 671, Springer, 1992, pp. 132–143.
[38] M. Schmidt-Schauß, Subsumption in KL-ONE is undecidable, in: Proc. 1st Int. Conf. on the Principles of Knowledge Representation and Reasoning (KR-89), Morgan Kaufman, 1989, pp. 421–431.
[39] F. Baader, P. Hanschke, A schema for integrating concrete domains into concept languages, in: Proc. 12th Int. Joint Conf. on Artificial Intelligence (IJCAI-91), 1991, pp. 452–457.
[40] M. Schmidt-Schauß, G. Smolka, Attributive concept descriptions with unions and complements, Artif. Intell. 48 (1) (1991) 1–26.
[41] J.F. Allen, Maintaining knowledge about temporal intervals, in: Communications of the ACM 26(11), 1983, pp. 832–843.
[42] D.A. Randell, Z. Cui, A.G. Cohn, A spatial logic based on regions and connections, in: B. Nebel, C. Rich, W. Swartout (Eds.), Principles of Knowledge Representation and Reasoning, Morgan Kaufman, 1992, pp. 165–176.
[43] F.M. Donini, M. Lenzerini, D. Nardi, A. Schaerf, W. Nutt, An epistemic operator for description logics, Artif. Intell. 100 (1–2) (1998) 225–274.
[44] W.W. Cohen, A. Borgida, H. Hirsh, Computing least common subsumers in description logics, in: Proc. 10th Nat. Conf. on Artificial Intelligence (AAAI-92), AAAI Press/The MIT Press, 1992, pp. 754–760.
[45] A. Artale, E. Franconi, N. Guarino, L. Pazzi, Part–whole relations in object-centered systems: an overview, Data & Knowledge Engineering (DKE) Journal 20, Elsevier, North-Holland, 1996.
[46] P.-H. Speel, P.F. Patel-Schneider, A whole–part extension for description logics, in: Proc. ECAI Workshop on Parts and Wholes, Amsterdam, August 1994, pp. 111–121.
[47] P. Lambrix, Part–whole reasoning in an object-centered framework, Lecture Notes in Artificial Intelligence, 1771, Springer, 2000.
[48] U. Sattler, Description logics for the representation of aggregated objects, in: W. Horn (Ed.), Proc. 14th European Conference on Artificial Intelligence (ECAI-2000), IOS Press, Amsterdam, 2000.
[49] P. Gärdenfors, Conceptual Spaces – The Geometry of Thought, The MIT Press, 2000.
[50] A. Artale, E. Franconi, A survey of temporal extensions of description logics, in: Annals of Mathematics and Artificial Intelligence (AMAI), vol. 30(1–4), Kluwer Academic Press, 2001, pp. 171–210.
[51] M. Wessel, R. Möller, A high performance semantic web query answering engine, in: I. Horrocks, U. Sattler, F. Wolter (Eds.), Proc. International Workshop on Description Logics, 2005.
[52] R. Möller, B. Neumann, M. Wessel, Towards computer vision with description logics: some recent progress, in: Proc. Speech and Image Understanding, IEEE Computer Society, 1999, pp. 101–116.
[53] U. Straccia, Reasoning with fuzzy description logics, J. Artif. Intell. Res. 14 (2001) 137–166.
[54] J. Heinsohn, Probabilistic description logics, in: Proc. 10th Conference on Uncertainty in Artificial Intelligence, Morgan Kaufman, 1994, pp. 311–318.
[55] D. Koller, A. Levy, A. Pfeffer, P-CLASSIC: a tractable probabilistic description logic, in: Proc. AAAI/IAAI, 1997, pp. 390–397.
[56] P.M. Yelland, An alternative combination of Bayesian networks and description logics, in: Proc. Principles of Knowledge Representation and Reasoning (KR2000), Morgan Kaufman, 2000, pp. 225–234.
[57] S. Hongeng, Unsupervised learning of multi-object event classes, in: Proc. of the 15th British Machine Vision Conference (BMVC'04), London, UK, 2004.
[58] T.O. Binford, T.S. Levitt, W.B. Mann, Bayesian inference in model-based machine vision, in: T.S. Levitt, L.N. Kanal, J.F. Lemmer (Eds.), Uncertainty in AI(3), 1989.
[59] R.D. Rimey, Control of Selective Perception Using Bayes Nets and Decision Theory, Dissertation, Univ. of Rochester, TR 468, 1993.
[60] K. Laskey, S. Mahoney, E. Wright, Hypothesis management in situation-specific network construction, in: Proc. 17th Conf. on Uncertainty in Artificial Intelligence, Morgan Kaufman, 2001, pp. 301–309.
[61] E. Gyftodimos, P. Flach, Hierarchical Bayesian networks: a probabilistic reasoning model for structured domains, in: E. de Jong, T. Oates (Eds.), Proc. Workshop on Development of Representations (ICML-2002), 2002, pp. 23–30.
[62] J. Pearl, Probabilistic Reasoning in Intelligent Systems – Networks of Plausible Inference, Morgan Kaufman, 1988.