Incorporating VR Databases into AI Knowledge Representations: A Framework for Intelligent Graphics Applications

M. E. Latoschik
Faculty of Technology, AI & VR Lab
University of Bielefeld
Bielefeld, Germany
marcl@techfak.uni-bielefeld.de

M. Schilling
Faculty of Technology, AI & VR Lab
University of Bielefeld
Bielefeld, Germany
mschilli@techfak.uni-bielefeld.de

ABSTRACT
This article presents a framework for incorporating commonly used VR (Virtual Reality) databases for graphics and physics simulation into an AI (Artificial Intelligence) knowledge base using a unifying semantic net (SN) representation. Several examples in the area of multimodal interaction for AI based graphics applications are given to motivate this approach. An evaluation of the chosen SN knowledge representation (KR) regarding five roles suitable to analyze a given KR is followed by a discussion about resulting conceptual and technical requirements for the underlying DB/KBMS (database/knowledge base management system) which supports the chosen KR as well as mediating layers for external simulation relevant modules.

KEY WORDS
Virtual Reality, Artificial Intelligence Techniques, Human-Computer Interaction, Multimodal Interaction

1 Introduction

This article illuminates central aspects of work conducted at the AI & VR Lab at the University of Bielefeld where Artificial Intelligence (AI) techniques are explored for the establishment of intuitive communication links between humans and highly interactive 3D computer graphics (CG). One of the lab's first approaches in that area was the VIENA project (Virtual Environments and Agents) [4]. Its primary goal was the desktop based interaction with an interior design environment via typed –albeit it incorporated first experiments with a commercial speech recognizer– language input and simple 2D pointing gestures (mouse and primitive data glove). A user could utter requests like *"Make the chair red."*, *"Move it a little bit to the right."* or *"Put it* [pointing gesture] *there."* where the interpretation of spatial adverbs like *"left"*, *"right"* or *"closer"* adapted to user preferences regarding different frames of reference of a) the user (egocentric) b) the object (intrinsic) and c) Hamilton, an anthropomorphic agent which was located in the graphical scene.

The CODY project (COncept DYnamics) [2] handles and processes direct manipulation as well as natural-language instructions for mechanical object assemblies where the objects have certain connection constraints and can take changing roles during the assembly task: The same generic part will have equal connection capabilities but, e.g., varying lexical denominations or connection preferences when used in an airplane's propeller or landing gear. Like VIENA, CODY's workplace centered around desktop based interactions with a 3D visualization.

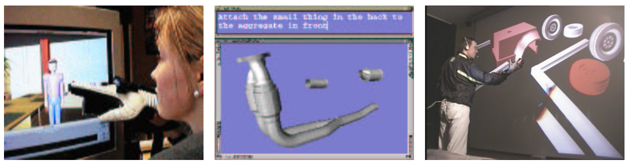
Attach the small thing in the back to the aggregate in front

Figure 1. Examples of AI based applications that offer uni- and multimodal interactions for 3D graphics environments. From left to right: VIENA, CODY and SGIM (see text).

In contrast to that, SGIM (Speech and Gesture Interfaces for Multimedia) [3] was our first approach to transfer multimodal interaction techniques into an immersive real-time environment. While using the CODY assembly knowledge base (KB), SGIM utilizes free multimodal input (gestures and speech) to interact with the environment by means of deictic, mimetic and iconic gestures occurring anytime during and after spoken input to specify object places, manipulations and shapes respectively. SGIM's multimodal integration module takes care of temporal and semantic relations between both modalities.

Figure 1 illustrates snapshots of interactions with the three systems. Most of our current projects now make use of one of the lab's immersive environments which, in addition, are inhabited by MAX, our Multimodal Assembly eXpert. He is a successor of the articulation wise limited Hamilton and has the ability to perceive his environment –the virtual one using virtual sensors located directly in the scenes graphical database as well as the real one using a camera. MAX –as a communication partner for the immersed user– is able to move around and to express himself using coverbal gestures.

Following the development of the lab's projects some principal problems arose. First of all, advances made once could not easily been utilized in a different or successive project and had to be re-implemented and sometimes re-engineered, a very error prone work. A good example for this is the existence of three different approaches for the reference analysis which all have (or *should* have) a significant functional overlap. But secondly, there is no straight forward way to transfer the desktop based AI related methods –which depend on quasi static scene descriptions with relatively unchanging positions regarding the user to the scene and the objects– to a highly interactive immersive environment where a user can change position freely even *during* the utterance of spatial adverbs and where objects might be animated and moving.

The resulting problems separate in three categories which are now revised, namely 1) **structures** and (possibly external) **formats**, 2) **interface types** and **access methods** and 3) potential **synchronizations** of the involved knowledge bases. The following paragraphs describe a knowledge and database management system which incorporates all necessary information for an AI augmented VR simulation system. Similar approaches (for different domains) found in [5] and [6] will be extended in the sense that a given representation will not be augmented with semantic knowledge but that representations relevant for a VR simulation will be incorporated into an AI knowledge base.

2 Related databases

This section takes a closer look at typically required databases and their related concepts in the context of projects illustrated, namely for the simulation of graphics, physics and AI. Terminology: The term *graphics database* (gfx DB) denotes data related to the visual simulation including associated structures (e.g., for sensor-based interaction modeling). The term *knowledge base* denotes all AI related databases which are organized using appropriate knowledge representation formalisms. Where specifically required, we will further distinguish databases regarding their domain, e.g., *physics database* (phys DB). The single term *database* (DB) covers all related domains. The terms DBMS (DB management system) and KBMS (KB management system) denote the underlying technical systems which support a given representation.

2.1 Knowledge domains and types

The domains of the relating data and knowledge types can be summarized as follows:

1. Graphical representation: The two dominating representation formats required for the shading process of the objects which make up the virtual scene in a real-time simulation are a) surface and normal centered or b) volume based.

2. Physical simulation: To allow the system to calculate dynamics (forces, momentums), collisions and friction, the objects need to be augmented with masses, mass distributions, shapes and surface attributes like friction coefficients.

3. Ontology: Objects that inhabit the virtual world and that we want to interact with can be thought to have an analogous representation by the interacting user. The ontology defines object concepts and specific attributes or features which make up the object:
(1) From the technical AI side, such an ontology describes the objects in relation to each other, e.g., typically using **is-a** hierarchies linked to instances by **inst-of** relations for representing taxonomies or **part-of** relations between aggregated objects.
(2) The conceptual representation is closely bound and connected to lexical data for a successful natural language parsing and interpretation. Entities in taxonomies or functional concepts will be associated with appropriate nouns and verbs respectively which denominate them in a given language.
(3) Attributes describe predicates of objects. A singular attribute can express a functional concept that does not need to be lexicalized, e.g., the fact that an object is ”collidable” might only be significant for the application but not be addressable by the user. An attribute can carry quantified information for a specific functionality, e.g., how much space is left in a predefined connectable spot located at an object.

2.2 Representation formalisms

1. The VR gfx DB representations broadly fall into two categories. The majority of applications uses scene-graphs (SG) –hierarchies of arranged CG-nodes– as the natural method for modeling relative six degree of freedom (6DOF) changes. In contrast to that, an object centered entity concept uses only shallow structures and is often found in game engines. The SG approach is frequently extended to incorporate object animations and user interactions by supplying specialized node concepts like interpolator, engines and sensors.

2. The phys DB is usually object or entity centered. An object’s 6DOF are required with respect to a global frame of reference. Object attributes are local features of an instance. Such objects are often stored in shallow DBs which can sort them due to aspects currently required by the physics engine, e.g. to quickly access all activated objects. The collision shapes can have a close association to the gfx DB.

3. The multitude of different AI representation formalisms imply certain views and reasoning schemes regarding the domain to be modeled where many of them being notational variants of each other. Technically the formalism to use has a large influence on the KB administration task. Some of the primary

formalisms are logic (e.g., predicate calculus), rules, frames, inheritance graphs, neural networks, semantic networks (SN) or KL-ONE based approaches.

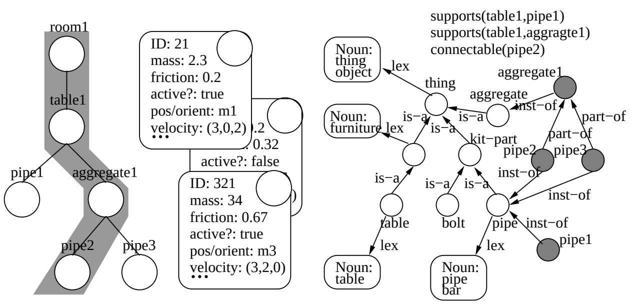

Figure 2. Typical parts of representations for the gfx DB, phys DB and KB respectively (from left to right).

An example snapshot of three regarded DBs during an assembly scene is shown in figure 2 where a single connectable object and one aggregate are placed on a table. The gfx DB is represented as a SG where the parts emphasized in dark grey show how some object attributes, here the 6DOF, are distributed and accessible only as unique paths in the graph. The phys DB is just a (sorted) accumulation of objects with attributes. The KB is separated in a logic part with predicates (figure 2 top right) and a semantic network which connects a taxonomy with lexical and instance information using respective relations.

2.3 Access methods and architectures

1. The typical access method for an SG graphics representation is the SG traversal where the graph is processed typically top-down. All information a requesting process needs is a starting point in the SG, e.g., the scenes root node.

2. Objects in the phys DB are usually accessed object centered. An application that requests read/write access to any of the associated data of an object in the DB needs a link to it. This is commonly provided by a low-level pointer or object ID interface which the requesting process must know and administer.

3. Access methods of KB formalisms are mainly tailored to the internal representation scheme, e.g., querying a logic system might be done by asserting a new axiom whereas accessing a DBMS representation of attribute-value pairs might favor SQL (structured query language). Nevertheless there exist approaches to access KBs using a unified access language. The Knowledge Interchange Format (KIF) and the Knowledge Query and Manipulation Language (KQML) as well as the Darpa Agent Markup Language (DAML) Query Language (DQL) being good examples for the latter.

3 The integration approach

3.1 Discussion of requirements

A closer examination of the design of two projects illustrated in section 1 shall guide the discussion to a set of specific requirements a more general approach for an architecture and DB/KBMS tailored for immersive environments must take into account. One way to incorporate different DBs is to favor a so-called **loose coupling** between the participating modules of a simulation application. Figure 3 illustrates the VIENA system architecture which was based on loose coupling. The design was primary driven by an *agent* based system modularization where each module or software agent had a specific function to solve.

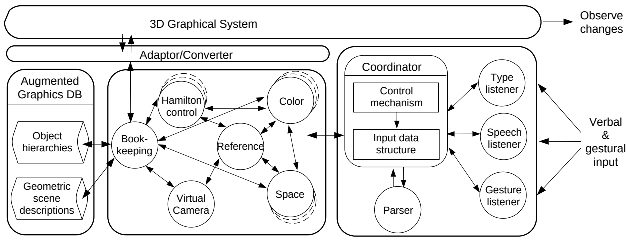

Figure 3. The VIENA architecture [4]: A loose coupling was favored between the different DBs.

Such an architecture imposes certain prerequisites: Redundant DB content might be accessed and maintained by different modules. This requires a clean locking semantics as well as a general synchronization scheme between modules with overlapping DB content. In applications that use a clearly serialized interact→compute→scene update circle this can be provided with some precaution, e.g., VIENA used one central controlling instance called bookkeeper (see figure 3) which handled all read/write accesses.

The SGIM project was faced with implications of an immersive real-time environment. A constantly changing relative scene representation, e.g., the user's or MAX's changing perspective and position/orientation with regard to a dynamic scene, required a **close coupling** between the gfx DB and all modules with spatial resolution or interaction functionality. In addition, the interact→compute→scene update scheme of discrete interactions now was supplemented by methods of continuous interactions, e.g., when a user describes a desired object manipulation using a mimetic gesture the SGIM system translates the unprecise trajectory into precise object transformations during the ongoing interaction. Following the SG type of the gfx DB, the required coupling was established by specialized node types which were integrated at specific positions in the graph and the simulation process. Figure 4 illustrates how **detectors** and **raters** serve as access entities for the SGIM related AI modules. They constantly provide the multimodal integration engine with data of certain aspects of ongoing user movements like gestures and changing view respectively. **Semantic entities** (SE) on the other hand allow a SG-type traversal to access KBs of specific SG-located objects. SE link to SG nodes (in figure 4 their siblings) and provide a standardized way to query object attributes via traversals. The remaining illustrated concepts like channels, actuators or motion-modificators and their connection types are mainly responsible for the continuous interaction modeling and shall be ignored here. Still, SGIM accesses parts of the application relevant KB which represents the assemblies of objects using loose coupling with explicit synchronization.

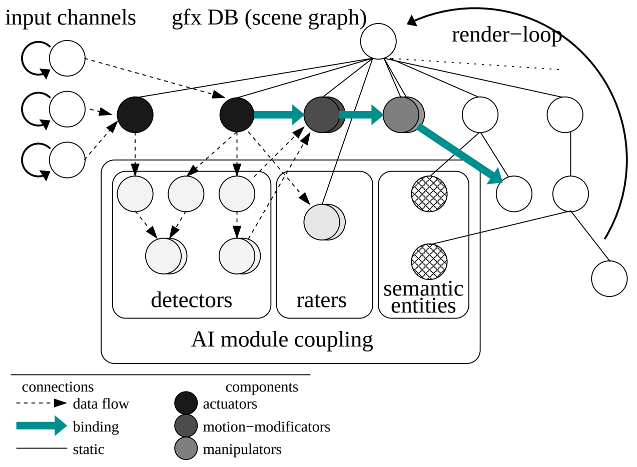

Figure 4. The SGIM architecture: A close coupling establishes direct access to spatial content represented for each frame by the gfx DB.

Following the SGIM related experiences and advances in incorporating AI and VR methods, it seems desirable to access all DBs using a close coupled scheme while allowing all participating modules to access data in their favored way. To summarize, such a unified DB/KBMS should support the following features: 1) A single and centralized representation with facilities for 2) access methods tailored to the accessing modules encouraging 3) close coupling between the representation and external processes having 4) unique entities in the representation while supporting 5) transparent synchronization in cases where the former is impossible due to external constraints.

3.2 Integration of representations using a Semantic Net

The choice of the representation formalism useful for the unified DB/KBMS is guided by work conducted in [1] which defines five roles of a KR as 1) surrogates, 2) a set of ontological commitments, 3) a fragmentary theory of intelligent reasoning, 4) a medium for efficient computing and 5) a medium of human expression. A closer inspection of these roles using a SN for a unified representation will lead the following survey, where we consider the technical DB/KBMS and the representation we will achieve with it.

Regarding the example illustrated in figure 2 interconnections between the three given representations expand the SN scheme of the shown KB fragment on the right of the figure. Two new relations serve this purpose: **has-phys** relations from the KB instances (and possibly from the taxonomy classes as a means of attribute inheritance) to the entities in the phys DB and **has-gfx** relations between the KB instances and the SG nodes which represent the gfx DB instances. This leads to a SN with all of the several representations included which are now accessible by following the appropriate relations from a given node in the net. Note that from an AI standpoint a SG type organization already is a SN connected by **part-of** relations here called **SG-child-of**. Figure 5 illustrates an instance- or object-centered view of a part of the resulting net. Some additional information is necessary to understand the consequences of the **SG-node** and **SG-path** relations or the additional **has-6DOF** concept inserted for pipe2. They will be discussed in detail in the upcoming sections about views and synchronization.

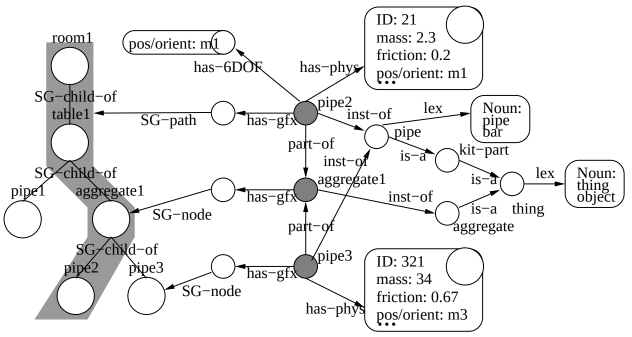

Figure 5. A part of a SN which incorporates gfx DB, phys DB and KB into one representation following the example from figure 2.

Regarding the five roles of a KR, the first role of the chosen representation as a surrogate for the entities in question is quite suitable since the objects do not even have to have a physical counterpart but might only exist in the VR. In such cases, the surrogate *defines* the object whereas the user's internal representation might (and most likely will) differ from the system's representation. Following [1], the surrogate is inherently faulty due to the fact that semantic alone determines the correspondence between object and surrogate, a fact that supports our approach where objects and their surrogate are regarded as equal. The problem of a suitable correspondence here locates between the user's assumption about object behavior and the behavior provided by the simulation which is based on the chosen representation. This leads to the question about ontological commitments we make regarding an actual representation. The latter is determined by the granularity of our incorporated object features and implemented functionality. The choice of abstraction is guided by the aspects of the world we want to simulate. If we provide, e.g., enough information about masses, friction and about object shapes etc., the physical simulation will be reasonable reliable. The next role as a theory of intelligent reasoning does not influence the DB/KBMS itself but the actual representation we choose to model with it. So far, our approach does not propose any specific theory of that kind and allows for experimental applications in this area but –for the examples given– already suggests a KB organization where concepts are parts of taxonomies which relate to lexical information. Efficient computing (role 4) is of major importance in a real-time environment. SNs alone do not guarantee efficiency by their internal structure but they have the potential to be very efficient by carefully choosing the relation semantics (by, e.g, avoiding circles and deep transitivity) and inferences based on these relations. The last role as a medium of human expression will be of significance for the KB maintenance task. For this, an external KB format has to be human readable and editable, an important aspect once the system will go into production state.

3.3 Access and Views

Access of the DB/KBMS is centered around instances or relations. In the first case we query attributes of an object, e.g., to ask if an instance is a kit-part or if it is connectable with respect to a possibly underlying assembly system. Note that in our approach this information can be represented in two distinct ways. In many cases it is sufficient to augment SN nodes with attribute-value matrices as containers for the requested information whereas an enhanced representation will insert a new relation which makes the queried attribute a full qualified object in that sense, that now the attribute itself can relate to other SN concepts in our representation, e.g., to incorporate lexical information for the attribute. The second access case requests informations by relation types like **SG-node** or **inst-of**. Setting up the graphics system on initialization time by the DB/KBMS is an example for the latter.

Both access methods define so-called **views** on the unified DB. Views represent dynamically generated sections of the DB which are of particular interest for a specific module. Views are a useful if not necessary simplification to make the potentially highly complex SN structure readable and comprehendible for humans, e.g, during a DB maintenance task. In fact, the separated structural representations in figure 2 can be interpreted as such views whereas in figure 5 the underlying structures are interconnected and merged resulting in a view to be just an online generated data extraction. Regarding the DB/KBMS, generating views is a common task and hence is a primary target for acceleration techniques like relation indexing.

3.4 Synchronization

One of the most crucial tasks for the DB/KBMS is to guarantee consistency. The prominent example for this is the gfx DB which determines the user's view. The system must keep up-to-date with what the user sees and how he sees it when reasoning about user utterances regarding the domain, otherwise the system's perception of (the mainly virtual) reality will differ from the user's perception. In that case, reasoning will be error prone by initial conditions and fail. Now, the concept of a central representation should not have any synchronization conditions but in the reality of a system engineering task such conditions almost certainly show up. When integrating third party modules, e.g., for the graphics system or for the simulation of collisions and/or dynamic behavior, these modules will most often technically or conceptually restrict the direct structure access (DSA) to their internal data representations which is necessary for a clean integration into the SN.

Since here we assume that both modules for graphics and physics a) have their own 6DOF representation and b) do not allow DSA, we have added a central unifying concept **has-6DOF** for pipe2 in figure 5 for illustration. Having relations to nodes with redundant attributes in conjunction, the system can automatically synchronize the representations only based on given relation semantics. For example, when –during the simulation loop– approaching a SN instance node which has a) the unifying **has-6DOF** relation and b) a relation defined to have its own redundant 6DOF representation like **has-gfx** and **has-phys**, the system can automatically register a synchronization step which propagates the potentially new 6DOF values. Note, that this does not impose any write semantics: The attribute changes can be competing due to the fact that, for the current simulation step, they occur simultaneously in the gfx DB and in the phys DB. Which write semantics to choose is application specific. A second example is the supports(table1,aggregate1) predicate illustrated in figure 2 which is only a notational variant of a **supports** relation that could be asserted between table1 and aggregate1. This relation can be synchronized with a SG based gfx DB in that sense, that **supports** in the KB is reflected as a **SG-child-of** relation in the SG allowing supported objects to move according to their supporting ones, e.g., when user interaction occurs by a gfx DB manipulation.

As a consequence, mediating layers between DSA restricted modules are required for the redundant attributes. Such a layer can in principal support two different operation types, namely polling and event based. The first one checks for every simulation step if attributes in the mediated modules have changed their values. If so, the layer proposes a synchronization step. This is computational inefficient because such a layer must check all attributes in all mediated modules even if the attributes have not changed at all. Even worth, an attribute check can result in a complex calculation, e.g., when traversing the SG to transform a distributed attribute representation to an object centered one as indicated in figure 5 following the **SG-path** relation. To overcome polling, the second operation type is favored which uses an event mechanism. Here, a changing attribute notifies the mediating layer of its change. The layer only collects the events and if at least one event arrives it proposes a synchronization step only for the changed attribute(s). The latter type of operation requires that an event interface must be available. Since regarding the DB/KBMS an attribute can be represented either as an attached attribute-value matrix or as a relation, both types have to support the event interface to notify a value change or the assertion or retraction of a relation.

4 Conclusion and future work

Regarding the requirements raised in section 3.1, the unified DB/KBMS with the chosen KR can be verified as being appropriate for establishing qualified correspondences between domain entities and their representational surrogates, taking the question of ontological commitment into account. The approach offers a flexible way to model different theories of intelligent reasoning and provides several technical concepts like indexing and events for efficient computing. The KR as a medium of human expression is partly supported by the views' concept whereas a semantically enriched external representation format is ongoing work. We are currently aiming at an XML based notation which incorporates physics and graphics content, e.g., by incorporating VRML or X3D representations.

Ongoing projects in the area of intelligent VR using multimodal interactions conducted at our lab constantly demand representation methods for the related KBs. Here, a unified and integrated system greatly enhances development time and transfer of results between projects by supporting sharing of representations for similar processes. Additional synergies might be possible, e.g., between the graphics and the collision/physics engine by using shared representations of different granularities of objects. Although some of the concepts presented in this paper are currently under ongoing development, many of the ideas usefulness could initially be verified in the context of the SGIM project and its successors. This includes, e.g., integration of query modules required by a KB (the detectors and raters, see figure 4) into the simulation loop using a close coupling or providing KB access via standardized traversal mechanisms of interface nodes (the SE) from the graphics system to allow bidirectional information flow to and from the KB. As of today, most of the SGIM based concepts regarding multimodal interaction in VR left prototype state and are now utilized in at least four different projects. A unified reference analysis engine based on a constraint satisfaction solver has been implemented which "sees" and accesses required information in a unified way (still using mediation layers) of the different DBs. This engine already incorporates most of the functionality of our former projects. So far most representations used were technically based on proprietary C code which did not clearly separate the KB from the KBMS or on a SCHEME (a LISP derivative) implementation. Regarding the requirements for a DB/KBMS to support implementation of acceleration techniques like indexing and events for changes in the KB, a closer evaluation of available systems did not succeed in finding an appropriate SN toolkit which could be extended to fulfill our requirements. Therefore current work resulted in an event-enabled low-level graph toolkit implemented in C++ as the technological base for the semantic net implementation as well as for the constraint satisfaction solver used for the reference analysis. Regarding the interconnection with different third party DBs (graphics and physics), a clean interface specification for mediation layers which attach to the event system for transparent synchronization, e.g., to allow to transfer from an object centered view to a SG representation where attributes are distributed over the graph is on the way.

Conceptually, the idea of a SN for the representation of graphics, physics and AI entities is not limited to the examples given here. First explorations to mirror actual SG structures using abstract **SG-type** SN nodes or to represent interactions including the a) actions and associated verbs, b) object capabilities as well as c) the interaction transformation into required SG interaction nodes seem promising.

References

[1] Randall Davis, Howard Shrobe, and Peter Szolovits. What is a knowledge representation. *AI Magazine*, 14(1):17–33, 1993.

[2] Bernhard Jung, Marc Erich Latoschik, and Ipke Wachsmuth. Knowledge-based assembly simulation for virtual prototype modeling. In *IECON'98: Proceedings of the 24th annual Conference of the IEEE Industrial Electronics Society*, pages 2152–2157, Aachen, September 1998. IEEE, IEEE Computer Society Press.

[3] Marc Erich Latoschik. A gesture processing framework for multimodal interaction in virtual reality. In A. Chalmers and V. Lalioti, editors, *AFRIGRAPH 2001, 1st International Conference on Computer Graphics, Virtual Reality and Visualisation in Africa, conference proceedings*, pages 95–100. ACM SIGGRAPH, 2001.

[4] Britta Lenzmann and Ipke Wachsmuth. A user-adaptive interface agency for interaction with a virtual environment. In G. Weiss and S. Sen, editors, *Adaption and Learning in Multi-Agent Systems*, number 1042 in LNAI, pages 140–151. Springer, Berlin, 1996.

[5] Stephen Peters and Howie Shrobe. Using semantic networks for knowledge representation in an intelligent environment. In *PerCom '03: 1st Annual IEEE International Conference on Pervasive Computing and Communications*, Ft. Worth, TX, USA, March 2003. IEEE.

[6] Claudio S. Pinhanez and Aaron F. Bobick. Approximate world models: Incorporating qualitative and linguistic information into vision systems. In *AAAI/IAAI, Vol. 2*, pages 1116–1123, 1996.