Creating Virtual Worlds with a Graspable User Interface

Hauke Ernst Kai Schäfer Willi Bruns

Research Center Work and Technology (artec)
University of Bremen
Enrique Schmidt Str. (SFG)
28334 Bremen, Germany
{ernst | schaefer | bruns}@artec.uni-bremen.de

ABSTRACT

We introduce a system to interact with computers using real objects as interface. The graspable model on the table has a congruent counterpart in the computer which is updated continuously. The use of concrete objects offers advantages for interaction and cooperation. The underlying virtual model provides capabilities like remote visualization, simulation and documentation which play an important role for processes in engineering and education. System implementation, interaction concepts, projects as well as the impact on distant cooperation are introduced in this article.

Keywords: real reality, graspable user interfaces, image recognition, data gloves, multimodal access, cooperation

1 INTRODUCTION

This document gives a description of selected aspects which become interesting when computer representations and its functionality are automatically allocated to physical models. This is of relevance to our research projects where we use physical models as an intuitive interface for computer based modeling. We introduce two of our applications that use this concept in two different scopes and explain afterwards the technologic basis of our systems including an overview over the system architecture. Moreover, we will compare real world interactions with language processing. Additionally, we will identify open questions for the application of the concept in distant cooperative environments and propose mechanisms for solutions.

By bridging the gap between the real and the virtual world we are in a position to overcome some of the difficulties which obstruct the computer based modeling. This is generally characterized by its interface technology like monitor, keyboard and mouse, in Virtual Reality applications Head Mounted Displays, Data Gloves and other n-dimensional input devices. These techniques in general obstruct communication, don't allow direct grasping with appropriate sensual feedback and deliverer a more or less incomplete representation of the model to the user. The application of concrete components gives the user the feeling and the experience of the real world. The use of 3D visualization and schematic representations creates links between the concrete and abstract concepts required for learning. The application of computer technology means that for instance visual, schematic and acoustic information can be made available to the user. Once we have generated a virtual world we can attach properties to the objects which allow us to simulate their functionality within the virtual model or to export them to off-the-shelf simulators. In order to achieve a bridge between these different media a new interface is required which is able to translate the construction of a concrete world into an abstract representation. This combines a multimodal access to virtual environments with a normal communication between people in the real world.

We use two different technologies to synchronize the position of the real objects with the positions of the virtual ones. The first one applies image recognition to detect and locate the real world objects using simple, fast and robust technologies. The second one uses magnetic trackers attached to the users hand. Monitoring the actions of the hands allows the synchronization of the position and also the movements of real and virtual objects. This approach allows the realization of modeling and demonstration processes for creating and programming virtual worlds. We summarize these concepts under the name *Real Reality*.

Published in: 15th Twente Workshop on Language Technology (TWTL 15): Interactions in Virtual Worlds. Nijholt, A.; Donk, O.; Dijk, B. (Eds.). University of Twente, NL, 1999

2 APPLICATIONS

2.1 APPLICATION IN VOCATIONAL TRAINING OF PNEUMATICS

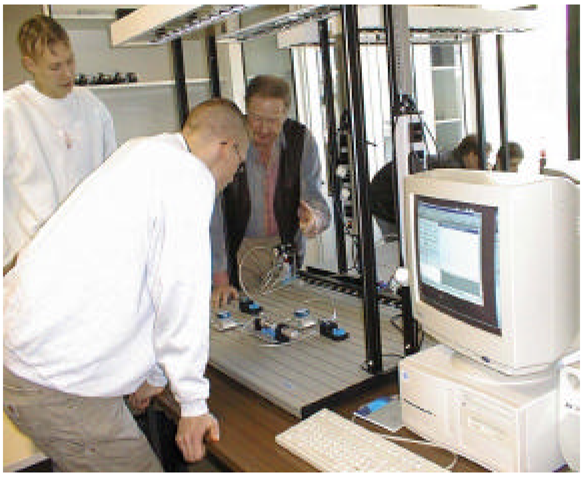

Fig. 1: Teaching Situation with the BREVIE ModellingDesk

After a couple of national projects which provide the theoretical background and the basic technology for our *Real Reality* concept we are now in the second year of the more product-oriented European (ESPRIT and Leonardo) project BREVIE: Bridging Reality and Virtuality with a Graspable User Interface.

The BREVIE project aims at designing, developing and evaluating a new kind of teaching and learning environment in the field of vocational training in pneumatics. The main characteristic of this environment is to provide transitions and links between two worlds which are currently separated:

the physical pneumatic circuit which can be grasped and touched with our hands, and the world of abstract models which can be viewed on paper or on a computer screen. The links are achieved by a "Graspable User Interface" which enables students to work and learn in both of these worlds (Fig. 1).

In the vocational training for pneumatics several kinds of learning material for teaching can be found. For the project consortium, the most compelling is the very popular pneumatic construction kit of our partner Festo Didactic which can be used for building functional pneumatic circuits. These circuits work with compressed air and allow a very close representation of real production environments.

The project makes use of these construction kits and develops technical links - through the Graspable User Interface - to computer-based learning media.

In the BREVIE project we use two low cost video cameras and image recognition algorithms to synchronize the physical model with a virtual model which is the basis for the processing on the computer. This automatic link between the real model and corresponding computer representations provides smooth transitions between multiple views on the same model which supports the development of rich mental models. In BREVIE we have integrated the following views, each with the possibility to interact with the model and to manipulate it:

- The physical model (Fig. 1)
- The virtual model in 3D (Fig. 3)
- A symbolic/logical view in a simulation software for pneumatics (Fig. 3)

The BREVIE Learning Environment also provides access to multimedial learning material which introduces the functional properties of the pneumatic parts (Fig. 4). The user can intuitively call up this information with the help of a pointing stick which enables him to select components within the physical model.

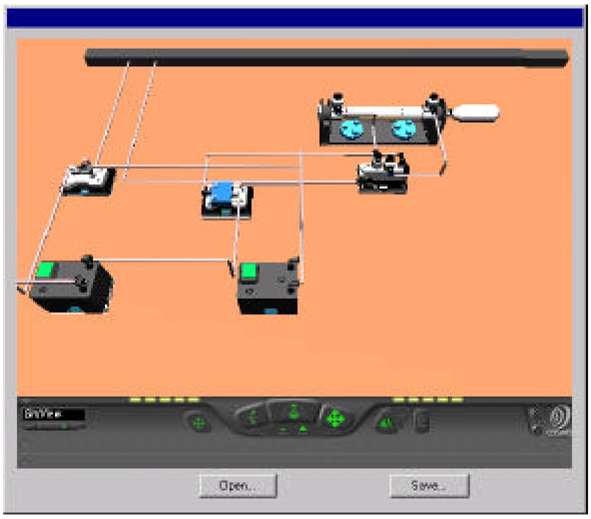

Fig. 3: 3D-View of the Virtual Model within the BREVIE Learning Environment

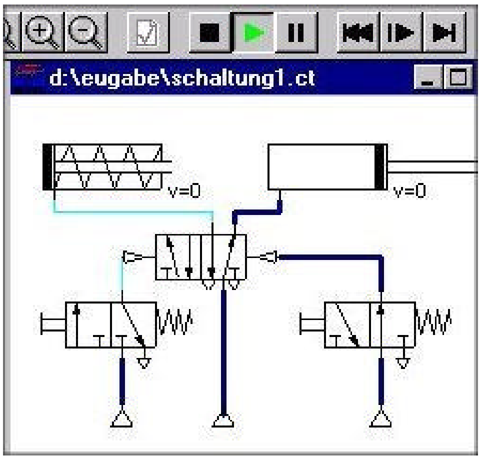

Fig. 3: Working with the Model in a Simulator for Pneumatics (FluidSim)

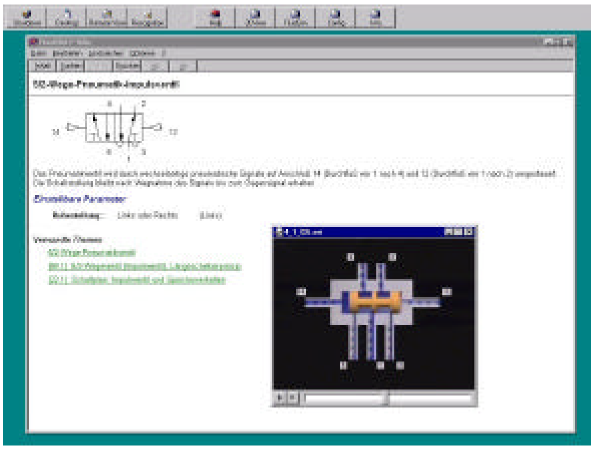

Fig. 4: Presentation of Learning Material in the BREVIE Learning Environment

2.2 Application in Factory Layout

In our latest completed project RUGAMS we applied the *Real Reality* Technology to the design of factory layouts and to the programming of plants like conveyor systems and robots (Fig. 5).

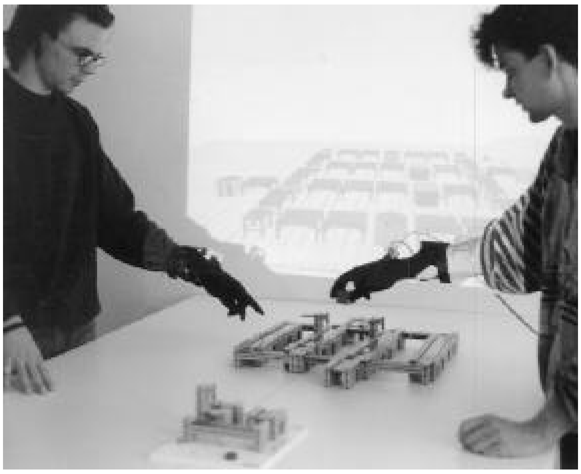

Fig. 5: Co-operative Modelling and Simulation of an Industrial Plant

In this context we configured the modeling environment for interdisciplinary teams to plan layout and material flow in complex factory scenarios. Therefore, we constructed an appropriate set of factory equipment like conveyors, plants, buffers and junctions presented in an object box (in the foreground). Corresponding virtual objects were been defined to represent geometry and behaviour of the elements. The group plans and discusses the layout in the modeling session. The computer in the background keeps track of the modeling process, by generating a virtual model of the plant. This enables a simulation applying the behaviour stored in the virtual components. A video beamer projects the activity of this digital factory model in the background to provide a feedback to the planning group. In later versions the simulated dynamics are projected directly onto the objects in the scene using Augmented Reality Technologies (Fig. 6). This helps to provide a better context to the model.

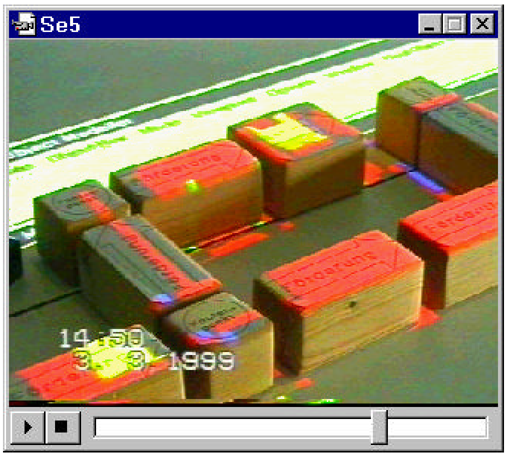

Fig. 6: Augmentation of a concrete Model to visualize the Dynamics of the Simulation

Later in this article we will describe in what way the predefined behaviour of the objects can be influenced when using programming by demonstration techniques. These demonstration techniques allow the specification of the material flow which optimizes the system performance. This environment supports a substantial part of the planning tasks within a group of decision-makers in just one session. This reduces the time to market significantly which is one of the industries' major goals.

3 Technology

3.1 Model Synchronization

3.1.1 Grasp Recognition

Bruns 1993 et al [1] laid the foundation for a new class of user interfaces in shop floor and handicraft working. The main characteristic of the "*Real Reality* User Interface", as they called it, is the application of the user's hand as a manipulator of physical objects in a real environment. Appropriate interface devices like data gloves and tracking systems capture the user's hand movements and finger flexions. Gesture recognition algorithms analyze the raw interface data and recognize gestures, grasps or user commands in real time [2]. Working with physical objects while being linked to a computer has a certain analogy to the well known Drag & Drop principle of GUIs. When the object is grasped all following data of the Drag-Phase is recorded. This phase terminates when the user puts the object to another location and releases it (Drop). Now, the physical object has a new position and due to this the virtual computer model of the physical environment has been updated. The system will trigger a grasp event, if a grasp gesture together with a collision between index fingertip and the boundary box of a virtual object is detected. Stopping a grasp gesture triggers a release event. Giving the user an acoustic feedback in the moment of grasping and releasing, the graphic output on a display becomes obsolete. So the user can work independently of the encumbering aura of the monitor, the keyboard and the mouse.

The term *Real Reality* emphasizes the difference to the term *Virtual Reality*. The user immerses in the Virtual Reality and is surrounded by the interface. *Real Reality* means to remain in the real world and to experience it. All human senses are stimulated and communication within groups is encouraged. The interface becomes a passive observer and is ideally not noticed by its users. We achieve this by linking physical objects to their virtual counterparts. As our physical objects always have a virtual counterpart they are called *„Twin Objects“*. In this way the description of actions effecting two model representations becomes much easier.

3.1.1.1 Preparing Real Reality Modeling Sessions

The *Twin Objects* are one of the basic elements of the *Real Reality* concept. For both kinds of object representations a number of instances must be available. This means to create a virtual representation consisting of the object's geometry and attributes describing the dynamic behavior. The geometric description contains the object's size (length, width, height) and its surface shape. On the other hand, the physical objects may be constructed by using technical construction kits, wood or other materials. The object representations may vary in shape, size and level of detail.

After putting on the data glove, its bending sensors must be calibrated by opening and closing the hand. Then the user places the hand on the tabletop which serves as the model ground, and the zero position is determined by the *Real Reality* system software. Now, the user's hand position is defined relative to the origin of the model's frame of reference.

In the initial state, the objects are located in an object box which has a predefined position on the tabletop (Fig. 7). Thus, for each object in the box the position can be computed.

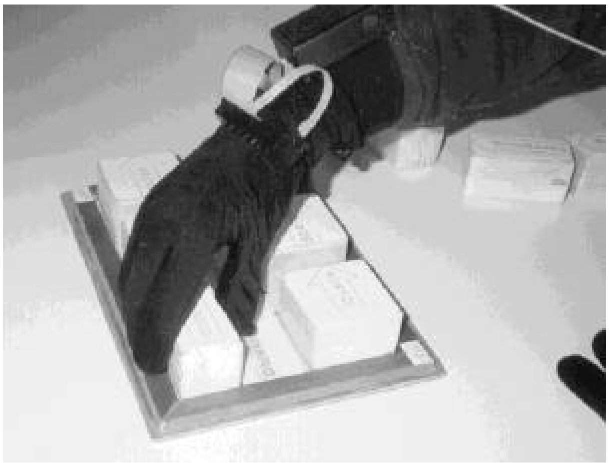

Fig. 7: Object Box

3.1.1.2 Using the Real Reality Modeling System

After the preparation of the modeling elements and the data glove, the *Real Reality* software is initialized and ready for use. A model is created gradually by taking *Twin Objects* out of the object box and putting them on the model ground. As two models are handled synchronously, the *Real Reality* system provides two views on the same model. With the help of 3D visualization software, the virtual representation is displayed and animated on a monitor screen, projected directly onto the model table or observed remotely via a network. Although the visual feedback is not necessary for persons who model with the physical objects, it is used for replaying actions recorded during a modeling session.

Fig. 8 shows the virtual representation of a hand reaching for a *Twin Object* contained in the object box, in this case a conveyor. It is taken out of the box and placed in a location near another conveyor which was inserted into the model during a preceding step (Fig. 9).

Some other forms of interaction are provided. By pointing at a specific object the user gets access to information about it, while creating a model. The information is shown on the display. For the future, voice output by the computer are planned.

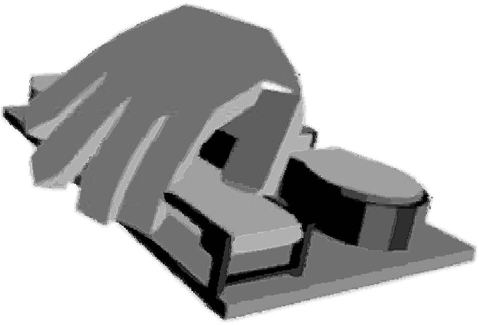

Fig. 8: The Virtual Hand grasping a Twin Object

The virtual model is represented on a scene graph in VRML97 Format. This is a dynamic representation of the real scene. Real-time analysis permits the filtering out of relevant events like collision, connnectings and separations, pile up elements and characteristic object motions. Saving the scene graph in a VRML97 File permits subsequently analysis, and serves as a documentation of the modeling process which can be recalled and graphically animated later.

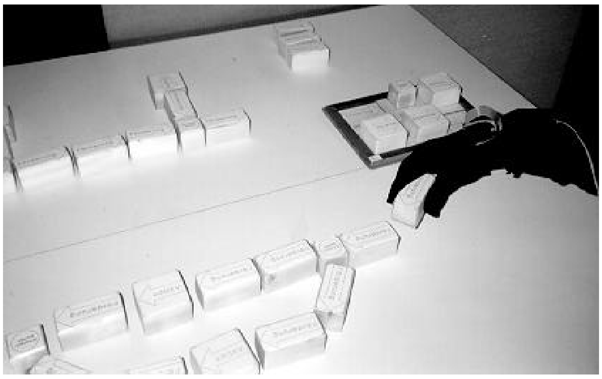

Fig. 9: Building a Model with Twin Objects

3.1.2 Image Recognition

In our latest research project (BREVIE) we decided to use image recognition methods to recognize actions in the real world because of financial restrictions. The system is designed for the application in schools which cannot afford data gloves equipped with tracking systems.

Therefore we developed a new software module which is capable to analyze images that are delivered by two low-cost CCD cameras (s. Fig. 10) and to recognize the components which are placed on the base plate by the user. The software is also able to recognize the pneumatic tubes which connect the components.

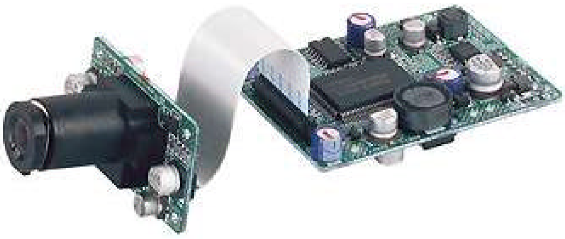

Fig. 10: CCD Camera

The output of the recognition process is also a VRML97 definition of the model which can be processed with 3D viewers and be converted to formats which off-the-shelf simulators can import.

For the first version we use color-barcodes which are mounted on the pneumatic components (s. Fig. 11) and the tubes. the barcodes turned out to be unfavorable for the commercialization of the system, we currently investigate in possibilities to exchange the unwieldy barcodes and look after techniques based on neural networks instead.

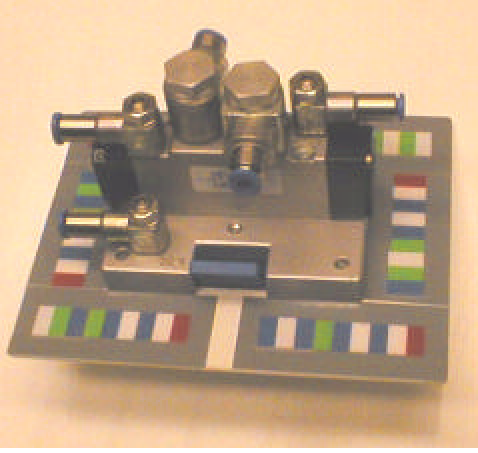

Fig. 11: Pneumatic Component tagged with Barcodes

3.2 ARCHITECTURE

In our *Real Reality* systems we integrate several hard- and software components into coherent applications. Since every application field requires a special combination of components we have introduced a highly modular system architecture which permits us to connect easily existing or new components. The design follows a client-server approach with a *Real Object Manager (ROMAN)* as the central server unit (s. 3.2.2). Other software units are connected to ROMAN via tcp/ip based Socket protocols. The following diagram outlines the spectrum of system components which are currently part of our

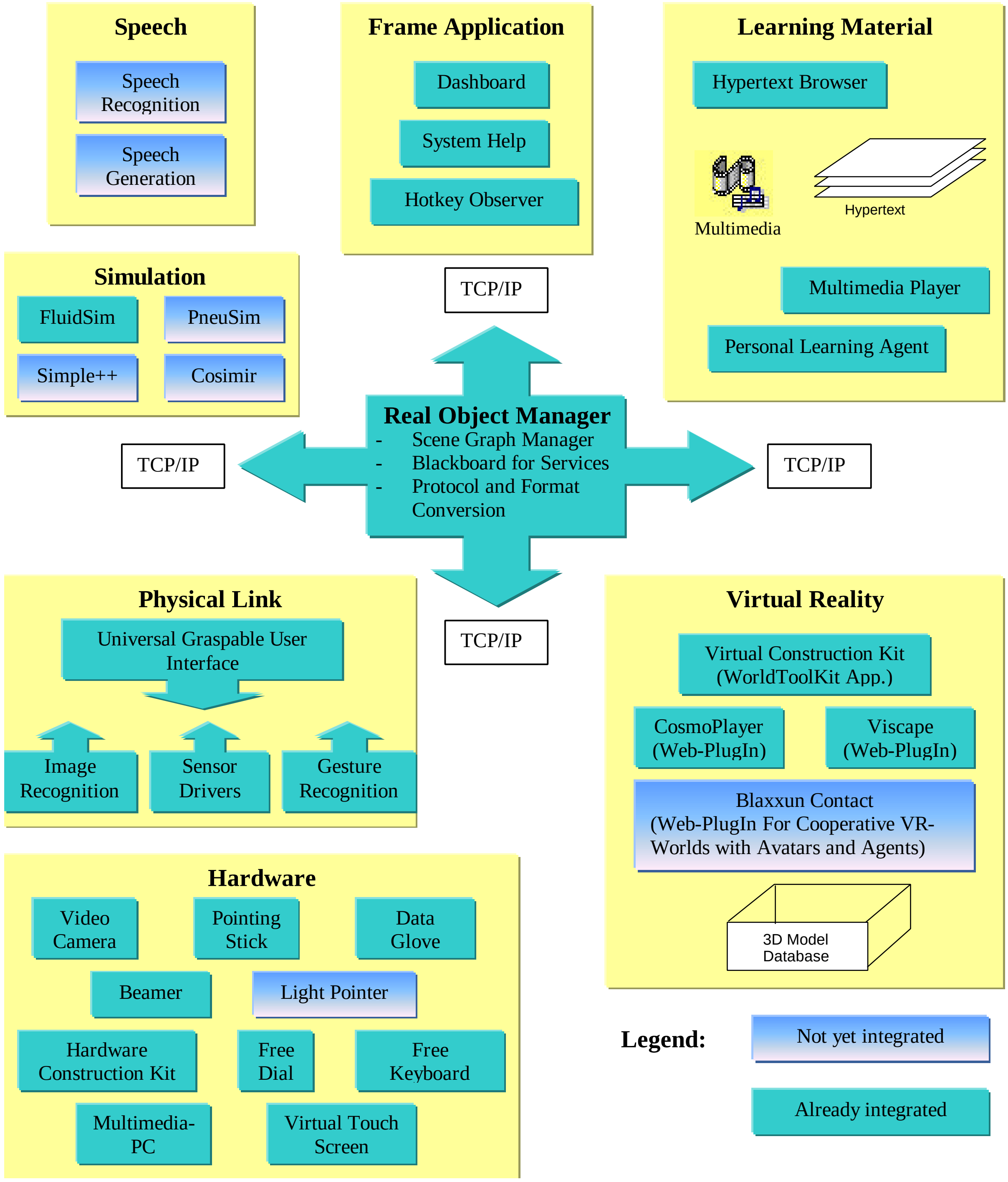

Fig. 12: Overview of the Real Reality Technology

applications or in our research focus including

- Interfaces to several simulators for pneumatics, electronic, robotics and mechatronics
- Interfaces to Virtual Reality systems
- Integrated learning material
- Interfaces to speech recognizing/generating packages
- A *Universal Graspable User Interface* as an abstraction layer with a plug-in interface for software plug-in modules which implement the link from the physical to the virtual world
- A couple of hardware devices for the interaction between the user and the physical model.

3.2.1 Special Hardware Devices

Some of the devices which are shown in the diagram-box “Hardware” may need additional explanations:

The *Grasp Tracker* is a new device which can replace the data glove. It is a ring worn at the index finger tip (Fig. 17). The tracking system is mounted on top of it, so that its relative position to the grasped object is fixed. The underside is covered with a pressure-sensitive foil for triggering grasp events. Grasp and pointing gestures can be distinguished by inclination. If more different gestures are desired, the *Grasp Tracker* can be worn together with a data glove (Fig. 17).

Free Keyboard and *Free Dial* are devices for data manipulation within the *Real Reality* environment. Although it is a pure fake device with no functionality and cable connection, it is possible to enter data with the sensorized hand. At the *Free Keyboard* keystrokes are detected through collisions with the virtual keys. The *Free Dial* works like the Shuttle Jog or Jog Dial known from video recorders and other technical devices. An actual value is attached to the actual dial position. Turning the dial permits a variation of the value between lower and upper limit. A sensorization of the dial is not necessary because all changes occur relative to the arbitrary starting position.

Virtual Touch Screen permits the interaction with a projected Windows Desktop without a sensorized surface. The mouse position is calculated from the relative position of the tracking sensor to the projection area. Click events are raised through the touch sensor at the *Grasp Tracker*.

3.2.2 Real Object Manager (ROMAN)

ROMAN as an Object Manager

ROMAN is the server unit in our client-server architecture. ROMAN coordinates the access to the virtual model data, thus ensuring data consistency. The other software components are connected to ROMAN as clients via a Socket link and do access or modify the model through this component by using protocols with a defined set of commands. The model is internally represented as a hierarchical scene graph which consists of the nodes and attributes.

The network connectivity is realized with tcp/ip stream Sockets. Concerning the network protocol ROMAN itself implements only a basic transport layer for text transmission. Higher protocols are added as plug-in modules.

ROMAN as a Service Coordinator

Based on the Client/Server techniques it is possible to build up a network of specialized software agents which offer services to each other (and in the end to the user). ROMAN treats the agents as clients and manages the services for each agent. In this context ROMAN is responsible for taking care of the message routing between the agents.

ROMAN manages services by means of a dynamic routing table for events. Clients can access this table via the network protocol. They can create events as well as subscribe or unsubscribe to them. On the other hand clients can raise events: ROMAN checks which clients have subscribed to the event and routes the event to these clients. Events can transport arbitrary textual information.

ROMAN as a pure manager of events and services takes a back seat from the users point of view. Since the ROMAN’s graphical user interface provides only monitoring information which is merely interesting for developers ROMAN will become invisible if installed as a release version.

Plug-ins

Aiming at a maximum generality ROMAN offers a plug-in interface for network protocols, format parsers and generators etc. This technique results in a strict modularity of the software and an optimal extensibility concerning further investigation and the integration of other products.

Network protocols as Plug-ins

One important application of the plug-in-interface is the implementation of network protocols as plug-ins. This mechanism enables ROMAN to coordinate clients with different protocols or different protocol versions. Existing client applications with existing protocols can, in this way, be integrated with minimal effort: it is only necessary to develop an interface plug-in which implements the protocol to fit the application to the ROMAN.

Parsers as Plug-ins

The support of different exchange formats for the definition of virtual models can be realized as ROMAN plug-ins, too. Such parser-plug-ins assist ROMAN in reading serialized model descriptions into the internal scene graph and vice versa.

As ROMAN contains a scene graph representation which is independent from concrete formats it can use the plug-ins to convert freely between the formats being provided by the plug-ins.

Other ROMAN plug-ins like the protocol-plug-ins are also able to make use of parser plug-ins to easily fulfil the clients’ format wishes.

In our projects we use mainly the VRML97 format.

3.2.3 Frame Application (Dashboard)

The Dashboard is the frame application of our *Real Reality* applications. It can be used in two different modes:

- The **configuration mode** allows a developer or a system administrator to configure what will happen in the execution mode. These configurations can be saved as so called start-scripts which may be installed as links on the Windows desktop or the Windows startmenu. The start-scripts comprise the definition of all programs involved as well as their start-parameters and dependencies. Furthermore function buttons and buttons to switch between different views can be specified. The configuration mode of the Dashboard provides a sound way to customize the system to each user's needs.
- In the **execution mode** all programs which where configured in a start-script are started and controlled. Moreover, it provides a common user interface to access the system's functionality. The Dashboard is visible as a button-bar on the upper edge of the screen which is always in the foreground.

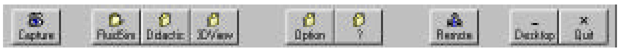

Fig. 13: Dashboard in Execution Mode

The Dashboard application can be connected to a ROMAN server and therefore sends events to the overall system if the user switches between different views or presses function buttons. Furthermore, the Dashboard accepts messages from ROMAN concerning the window management.

4 PROGRAMMING BY DEMONSTRATION

As described before one of the major aspects of simulation models is dynamic. One of the major advantages when using the grasp detection technology is the capability to recognize dynamics the users perform with model objects. Interpreting this demonstration as programming-language enables the user to program system behaviour for simulations and plant control. To make the computer redo respectively to derive programs from what the user has previously demonstrated, is an old issue in human-computer interaction. This research is focussed on the conventional WIMP (Windows Icons, Menus and Pointers) interface styles [3]. The 3D interface provided by the *Real Reality* concept offers new opportunities for specifying dynamic model behaviour by demonstration. The investigation of this potential is one of our main goals of research. In this section we discuss an approach to this issue.

4.1 PROGRAMMING MATERIAL FLOW IN CONVEYOR SYSTEMS BY DEMONSTRATION

Currently, we are working on the application of the *Real Reality* concept for event based simulation systems for material flow control and plant layout. A typical scenario in this area of application is a conveyor system supplying several machines in a factory with raw materials or partially finished parts for further processing. It is the task of a control system to ensure an optimal utilization of the production capacities. Especially the order of machine utilization is an important factor for the productivity of such systems. In practice, this logistical problem is hard to solve, even for small plants. Therefore, simulation technology, particularly event-based simulators are a widely used tool.

In order to support this kind of simulation task a construction kit consisting of conveyors, machines and workpieces has been built (Fig. 5 and Fig. 9). Rectangular solids represent conveyors, the quadric solids are the machines, where specific operations are performed. The arrow on each element indicates the direction of the material flow. If a single element has two exits, which means that there are two possible directions available to continue, this constellation will be called a branch. On the other hand, if a single element disposes of two inputs, the material may flow together again. Such elements may cause conflicts if material is delivered from more than one direction at the same time.

At a branch a decision is necessary to determine which direction has to be followed. In our examples, blank workpieces are delivered and put in a circuitry where they are buffered and move around, until a series of machine operations is performed with them. Afterwards the workpieces leave the circuitry via a branch. This simple system allows the investigation of important aspects regarding the flow of workpieces through the plant. The main issue discussed here is the question how to derive rules for branching decisions from the input recorded with the *Real Reality* Modeler. Furthermore, these rules must be represented in a formal notation for the utilization in subsequent simulation experiments as well as for the transformation into control programs.

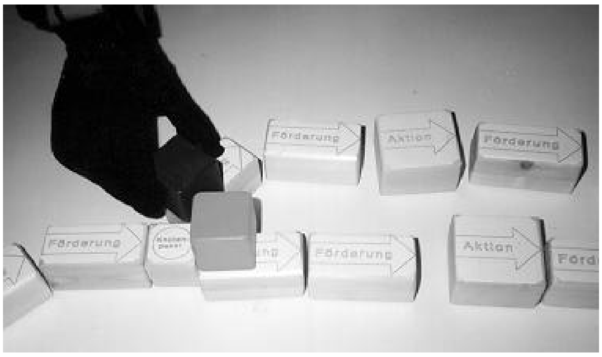

Fig. 14: Demonstrating a Branching Decision depending on Object Properties

Already our first program version presented at the Hannover-Messe ´96 was able to generate rules depending on workpiece attributes coded by color. Of course, a more sophisticated control mechanism is needed. Fig. 14 shows a situation in which the following specific rule has been demonstrated: *„move all light containers straight and branch dark ones out“*. This rule is extracted, transferred to the simulator and the participants can evaluate their program behaviour immediately in this system configuration.

In a different situation, the user may demand from the system to make a decision depending on the current state of the plant. Each resource (machine or conveyor) of the plant is either free or occupied. These two states determine whether a resource is available for processing or not. In the model this is indicated by placing workpieces, represented by colored cubes, on the resources. In a branching decision just a few resources are relevant. The context of a decision rule must be specified by placing tags on these resources (Fig. 15). This situation shows that the state of the two machines determines the decision of branching which is indicated by their tags (see the small discs). One of the machines is occupied whereas the other one is free for processing. The user moves the light-colored workpiece towards this machine. From this action the following rule can be derived: *„if machine A is occupied and machine B is free then route pallets to B“*. From now on, the simulator will apply this rule each time the specified condition occurs.

These activities of demonstration can be processed as a programming language. This allows the recognition of user intentions and the generation of formal specifications serving as programs for simulation or plant control. The major advantages compared to textual as well as to mainly graphical programming languages are:

- Easy to learn
- Context of the program and the plant model is kept permanently

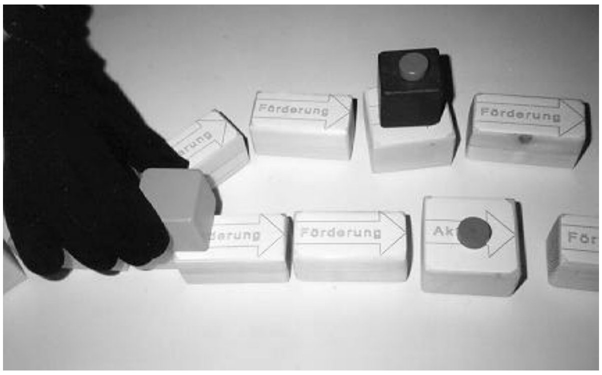

Fig. 15: A Branching Decision depending on the Plant's State

- Immediate feedback through simulation is possible
- Simultaneous and cooperative work of participants is being featured

Machine understanding of demonstrated rules is the topic of the following passage.

4.2 A STAGE MODEL OF DEMONSTRATION ANALYSIS

A model consisting of seven stages has been developed to model the process of plant layouts and material flow programming from the first conceptual meeting to the deliverable specification and control programs. From stage to stage an abstraction of the raw input data to formal programs takes place. Fig. 16 illustrates and names the stages. Above the square process stages the characteristic input information provided by the system is shown. This information partially depends on the application context and is therefore named context knowledge. Below the process stages the feedback provided to the users is represented. This feedback either supports the user interactions at the model directly or offers relevant information about the actual model. This computer generated information helps to refine the model and optimize the planning results.

4.3 COMPARING REAL WORLD INTERACTIONS WITH LANGUAGE PROCESSING

The tracker-based modeling generates data streams which can be processed analogous to natural speech. This gains to make use of advanced technologies for speech recognition.

The data streams of the glove and the tracking system are treated in seven successive processes (see Fig. 16) to define VRML97 scenarios for simulation and control programs. These steps are related to those in language processing. While glove and tracking system deliver vector streams directly, those

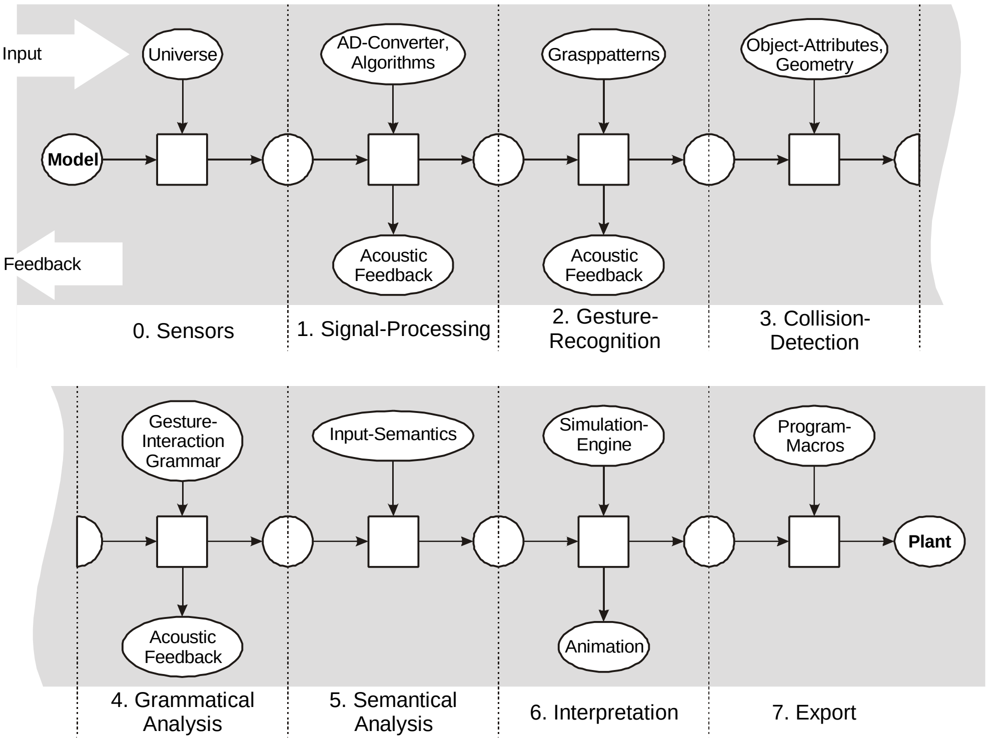

Fig. 16: Processing User Inputs on a concrete Model in 7 Stages

have to be extracted first from data streams recorded by microphones for language. It is difficult to describe voice patterns analytically. This is the reason why statistical methods which can be trained are more common [4]. Subsequent analysis of language makes use of knowledge which can be specified like lexical expressions kept in lists or the permitted grammar specified in some formal manner [5]. Depending on the application, extracting the semantical meaning out of the spoken language is a difficult task not solved satisfactorily yet. One of the main reasons for this problem is that knowledge about the relevant context is required and new information must be integrated into the database which means processing and learning in real-time. The system architecture may vary in different approaches to solve typical language recognition problems like

- individual characteristics of speakers and microphones,
- improper and fuzzy pronunciation and
- vague or indistinct meaning of the context and
- relevance of intonation.

Anyway, in the left column Tab. 1 tries to identify typical stages for language recognition.

We consider interpreting of taught behaviour as programming language for industrial plants to be a much easier task than this. Our reasons are as follows:

- Tracking system and data glove deliver feature vectors directly.
- We deal with a formal programming language defined by the target application.
- Relations between objects like collisions are easier to determine than between words and meanings.
- The recognition system doesn't have to learn.

Nevertheless, we may profit from developments made in language recognition. This idea led to the development of the seven stages in Fig. 16. The right column of Tab. 1 compares gesture interaction processes directly with language recognition. We see the different input data types converge with ongoing processing, and they are equivalent from step 4. on. We found this paradigm very helpful and also stable in proceeding development of our applications. Consequent idea is to enlarge the possibilities of interactions with real objects to a more expressive language for programming and communicating with computers and partners connected via the web.

	Language processing	**Gesture interaction**
0.	microphones	data glove and tracking sensor
1.	signal processing & feature extraction	signal processing
2.	phoneme detection	grasp recognition
3.	lexical analysis	collision detection
4.	grammatical analysis	interaction grammar
5.	semantical analysis	semantical analysis
6.	interpretation of information	interpretation of information
7.	control of applications	control of applications

Tab. 1: Process Steps of Language Processing compared with Gesture Interaction

5 DISTANCE COOPERATION ASPECTS

As already mentioned in chapter 2. (Applications) the modeling within a *Real Reality* environment is especially suited for working cooperatively with the physical model. Several users can stand around the modeling desk, and we only have to make sure that the sensors can recognize the actions of all participants and that everybody has a good view on what happens on the monitor.

Thinking one step further and imagining the use of *Real Reality* in applications for cooperative modeling with participants which are locally separated but connected via the web we will come to a new class of conceptual and technical questions – but also to a couple of new application fields. These fields will be discovered in coming research projects.

5.1 SYNCHRONIZATION OF PHYSICAL MODELS

Unlike the possibilities purely local *Real Reality* applications offer, the user of the modeling environment shares now his model with other, distant users now, i.e. remote participants cannot only view but actually manipulate it..

Remote users can either work with their own physical models or just with purely virtual models on normal computer workstations. For the other participants this should, however, make no difference, since modeling in the real and in the virtual world are, in principle, interchangeable forms of modeling.

A fundamental problem in the application of *Real Reality* and spatially separated, cooperative environments exists in so far as, that the system must show the user clearly, what has been changed at the model by other participants.

One possibility would be, that e.g. a robot automatically executes remote manipulations. The employment of a projection for the augmentation of the virtual model on the modeling desk through a beamer is certainly easier. A user of the physical model would then have to re-build the change of the remote user manually.

A further, completely different approach would be to employ multiple virtual models – one for each physical model. In the 3D view this could be shown as a corresponding number of virtual tables with working plates which are arranged next to each other. The user could navigate between the tables in the virtual world, look at what the others have build up and manipulate it within the virtual world. This technique would be very suitable for the support of the creation and discussion of alternative solutions.

We will further investigate in the question which of the two techniques fulfill the needs for cooperative *Real Reality* system best and think about dependencies on application fields.

5.2 VIRTUAL PRESENCE

In comparison with purely local *Real Reality* environments an important conceptual change occurs when modeling takes place cooperatively and spatially separate: the reference model moves from the physical world into the virtual world. The user of the physical model doesn’t define anymore the model through his operations alone, since the virtual model can become newer than his physical one through the manipulation of the remote participants.

Therefore, the user must pay more attention to what happens in the virtual now, he must observe the other users’ actions and make sure that his own actions will become visible to the others. Despite spatial separation he needs the feeling to be really located in a common, virtual modeling environment. Thus the user requires the perception of virtual presence, which procures him the sense of participation and involvement.

The feeling of presence appears as consequence of different sensory impressions. Important elements are, among others ([6]):

· Increase of the sensory information.
· Control over the relationship of sensors to their environment (through navigation, grasping, rotation of the virtual head for changing the viewpoint and the acoustic perception).
· Ability to manipulate the environment.

In the development of cooperative applications we can meet these points by extensively projecting the common virtual 3D-model onto the wall behind the modeling desk (s Fig. 5). A network connection to each remote participant is taking care of the immediate indication in the virtual model of all actions recognized by the sensors. With the help of an appropriate input device the user can navigate in the virtual model.

The own virtual presence and the one of the other participants should also contain a visual manifestation within the virtual world suitable for giving the users a good impression of the others’ actions in realtime. Such virtual representatives are also called avatars. Usually, avatars are presented as human-like figures. When selecting the combination of sensors for cooperative *Real Reality* applications we must pay attention to the possibility of extrapolating meaningful avatar movements from the sensor's input.

At the edge of this passage we want to mention, that such movable 3D figures are also very likely to provide functionality within virtual worlds. In this case the figures wouldn't be the representatives for human participants, but serve as metaphors for functions. They would be controlled by software modules and could play certain roles within the virtual world, e.g. offer access to help information. A concept for the behaviour definition of such automated characters is proposed in [7].

5.3 CONFERENCING

The users must be able to call the others' attention to their changes, must be able to explain, what they intended to do and must, on the other hand, have the possibility to ask the others, what they thought concerning their latest actions or what they plan to do next. Therefore, the system must offer mechanisms for getting into contact with each other. At this point the integration of conference- and chat-functionality is obvious (s. [8]).

Interacting with the participant's avatar would be an appropriate metaphor for using conferencing in virtual worlds. This could include selecting the avatar as an intuitive way to initiate a one-to-one talk and the mapping of the live video stream onto the avatar's head.

5.4 REMOTE POINTING AND SELECTING

Conversations regarding the model can also be supported with an appropriate visualization of actions like pointing and selecting by using the avatar figure. Pointing, for instance, could be indicated as movements of the corresponding avatars' arm and fingers; a selected object could be emphasized with a colored frame (s. [8]). In the physical model the marking of components could also be realized by employing a projector or a beam.

6 CONCLUSION

We agree with attempts to make engineering more visual in order to allow people to understand the process of technical planing. A rapidly growing number of projects with related topics show the evidence of that [9].

Using graspable models goes beyond that. People are in a position to easily interact in the model world and to participate in the planing process [10],[11],[12]. Bringing in their ideas leads to more planing quality and acceptance in the results [13].

By the way, people are learning very well in operating with objects because haptics are a major episode for learning to understand the environment [14]. The major drawback for making use of Virtual Reality lays in the modeling process. Designing virtual models is an expert domain and very cost intensive. This is currently leading to a development of giving away this job to polish or indian programmers occasionally (Panel "VR-The Future Engineering Workplace: A European Perspective" on Virtual Environments ´98 Stuttgart). Also for simulation studies, having an evident positive effect on planing processes, a major problem area is seen in a too pure support of the modeling process [15]. While the separate profit of concrete or virtual models is quite well proved we can assume an even better benefit from coupled models.

To go beyond theory and expectations we needed a prototype to experiment with coupled real and virtual worlds. This has been realized with the prototype of the *Real Reality* environment described in this article. During the design process it turned out that gesture based interaction requires complex software structures. We solved this problem by structuring this task into the 7 stages described above. Another major problem area is the required hardware. Data gloves cost either as much as a small car (Cyber Glove) or are not comfortable to wear and deliver inaccurate values (Nintendo Power Glove, 5th Dimension). In consequence, we are working on a pressure sensor based interaction device (see Fig. 17) [16].

The principle of magnetic tracking (Polhemus) will do a satisfying job if we accept modeling with non metallic objects. A new Tracking System based on opto-acoustic/inertial principles (Intersense) looks promising to solve this problem.
In practical use we have experienced a good handiness and acceptance of the system especially with non-expert groups modeling technical situations. We intend to certify this observation in an evaluation in vocational schools which is part of the project BREVIE.
An augmentation of the model provided by a video beamer projecting computer generated information directly onto the model, couples information from real and virtual model in an ideal way.
Motivated by the results of the past, we will concentrate our future work at artec on improving the *Real*

Reality interaction environment and keep you informed.

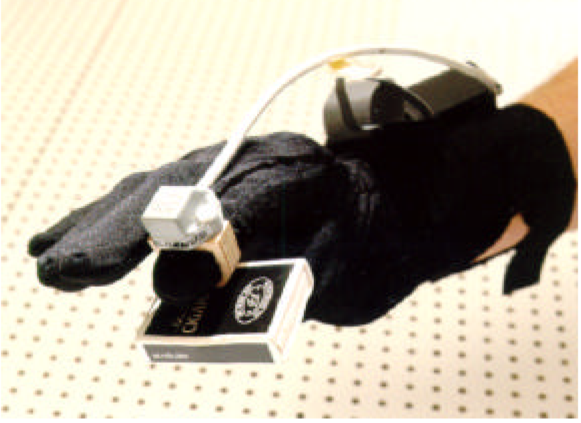

Fig. 17: Grasp Tracker

7 References

[1] F. W. Bruns. Zur Rückgewinnung von Sinnlichkeit - Eine neue Form des Umgangs mit Rechnern. In Technische Rundschau Heft 29/30, pp. 14-18, 1993.

[2] V. Brauer. Feature-basierte Erkennung dynamischer Gesten mit einem Datenhandschuh. Diplomarbeit, Universität Bremen, 1994.

[3] E. Cypher (Ed.). Watch What I Do - Programming by Demonstration. MIT Press, Cambridge, Massachusetts, 1994.

[4] R. Rabiner, A. Lawrence. Tutorial on Hidden Markov Models and Selected Applications in Speech Recognition. In: Proceedings of the IEEE, Vol. 77, No. 2, February 1989.

[5] H. Niemann, G. Sagerer, F. Kummert, M. Mast. Repräsentation und Nutzung von Wissen in einem Spracherkennungs- und Dialogsystem. In: Künstliche Intelligenz: Themenheft Mustererkennung, Band 3/95, pp. 35-41, 1995.

[6] W. Barfield, D. Zeltzer, T. Sheridan, A. M. Slater. Presence and Performance Within Virtual Worlds. In: W. Barfield, T. A. Furness. Virtual Environments and Advanced Interface Design. Oxford University Press, New York/Oxford, 1995, pp. 473-513.

[7] H. Ernst. Automatisierte Charaktere in Virtuellen Welten. Diplomarbeit, Universität Bremen, 1997.

[8] K. Schmudlach. Computerunterstütztes Konferieren in einer gegenständlichen Modellierumgebung. Diplomarbeit, Universität Bremen, 1998.

[9] Industrie Management – Visual Engineering 1/97, GITO-Verlag, 1997.

[10] H. Ishii, B. Ullmer. Tangible Bits: Towards Seamless Interfaces between People, Bits and Atoms. In: Proceedings of CHI 1997.

[11] M. Rauterberg, T.Mauch, R. Stebler. What is a Promising Candidate for the Next Generation Interface Technology? In: Proc. of the 5th Conf. on Interfaces To Real & Virtual Worlds, Montpellier, 1996.

[12] E. Arias, H. Eden, G. Fischer. Enhancing Communication, Facilitating Shared Understanding, and Creating Better Artifacts by Integrating Physical and Computational Media for Design. Proceedings of the DIS (Designing Interactive Systems), Amsterdam, 1997, pp. 1 ff.

[13] Scheel, Hacker, Henning. Fabrikorganisation neu beGreifen. TÜV Rheinland, Köln, 1994, pp. 155 ff.

[14] D. Müller. Simulation und Erfahrung. Ein Beitrag zur Konzeption und Gestaltung rechnergestützter Simulatoren für die technische Bildung. Dissertation, Universität Bremen, 1998, pp. 97 ff.

[15] G. Reinhart, K. Feldmann. Simulation – Schlüsseltechnololgie der Zukunft? Stand und Perspektiven. Herbert Utz Verlag Wissenschaft, München, 1997, pp 35-37.

[16] K. Schäfer, W. Bruns. artec-paper 67. Rechnergestützte Übergänge zwischen gegenständlichen und abstrakten Modellen produktionstechnischer Systeme. Dritter Zwischenbericht zum DFG Forschungsprojekt RUGAMS. artec, Universität Bremen, 1999.

More Information under:
http://www.artec.uni-bremen.de/field1/rugams/
http://www.brevie.uni-bremen.de/

Contact:
Hauke Ernst | Kai Schäfer
{ernst | schaefer} @artec.uni-bremen.de
Phone: ++49 / 421 / 218- {9089 | 4833}