A Gesture Processing Framework for Multimodal Interaction in Virtual Reality

Marc Erich Latoschik (marcl@techfak.uni-bielefeld.de)
AI & VR Lab [20], Faculty of Technology
University of Bielefeld, Germany

Abstract

This article presents a gesture detection and analysis framework for modelling multimodal interactions. It is particulary designed for its use in Virtual Reality (VR) applications and contains an abstraction layer for different sensor hardware. Using the framework, gestures are described by their characteristic spatio-temporal features which are on the lowest level calculated by simple predefined detector modules or *nodes*. These nodes can be connected by a data routing mechanism to perform more elaborate evaluation functions, therewith establishing complex detector *nets*. Typical problems that arise from the time-dependent invalidation of multimodal utterances under immersive conditions lead to the development of pre-evaluation concepts that as well support their integration into scene graph based systems to support traversal-type access. Examples of realized interactions illustrate applications which make use of the described concepts.

CR Categories: D.2.2 [**Software Engineering**]: Design Tools and Techniques - *User interfaces*; H.5.2 [**Information Interfaces and Presentation (e.g., HCI)**]: User Interfaces - *Interaction styles, Natural Language*; I.2.7 [**Artificial Intelligence**]: Natural Language Processing - *Language parsing and understanding*; I.3.6 [**Computer Graphics**]: Methodology and Techniques - *Interaction techniques, Languages, Standards*; I.3.7 [**Computer Graphics**]: Three-Dimensional Graphics and Realism - *Virtual reality*

Keywords: 3D HCI, gestures, multimodal, gesture processing, multimodal interface framework, gesture and speech input, interaction in virtual reality, immersive conditions.

1 Introduction

Human-computer-interaction (HCI) plays a significant role in the perception and the acceptance of computer-based systems. The interaction-metaphors went through changes that directly matched the technical input/output capabilities. It all started with batch mode operation. Input was specified by the adjustment of hardware registers or the use of punchcards. The output was delivered through lamps or typewriters. The invention of text-terminals and graphics-displays together with new input devices (e.g., the mouse) allowed the development of interaction techniques that finally lead to the so-called WIMP[1]-style interaction nowadays found in almost every computer system.

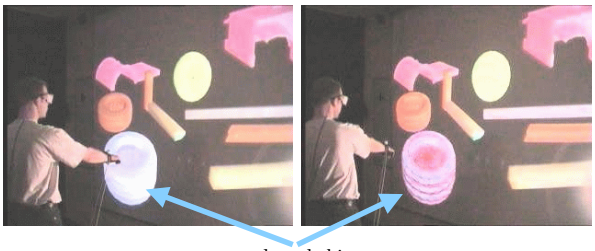

Figure 1: Multimodal interaction in front of a wall. A user selects an object (a wheel) using combined speech and a pointing gesture.

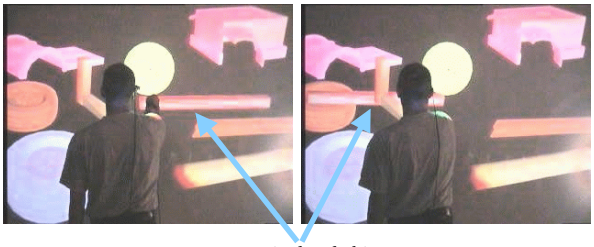

Figure 2: A previously selected object (a bar) is dragged by a distant grasping gesture parallel to a translational hand movement.

In this context, Virtual Reality makes new demands on how to interact *with* and *in* virtual worlds. Many VR-systems utilize — WIMP-like — point-and-click methods for selecting objects in space or commands from (in VR) *floating* menus. But an immersive 3D-display[2] extends the user's degrees of freedom (DOF) and often simulates a realistic surrounding. In such an environment the described interaction metaphor seems somehow clumsy and unnatural. To overcome the latter, there had been attempts to utilize natural multimodal communication in graphics applications and later in VR-systems. Figures 1 and 2 show two examples of multimodal and gestural interaction in a virtual construction scenario which are implemented with techniques described in this paper. A user selects an object by saying *"Take this wheel and..."* with a combination of a pointing gesture. After the selection process an object can be manipulated, e.g., to drag ist around (figure 2), more elaborated interactions are described at the end of this article.

[1] WIMP: Windows, Icons, Menu, Pointing

[2] A large screen display like a *wall*, a *cave* or a *responsive workbench*.

2 Related work

In the beginning of 1980, Bolt [3] developed the Put-That-There system which allowed to place (2D) graphical objects using a strict dialog driven interaction style. The pointing direction was measured with a 6DOF sensor and the actual dialog situation determined the evaluation time. Most of the following work, e.g., by Hauptmann and McAvinney [6], Koons et al. [7], Maybury [12] or Lenzman [10] concentrated on the multimodal integration of deictic utterances and there exploitation in 2D applications.

Böhm et al. [1][2] used VR-systems as the interaction testbed but considered only *symbolic*[3] gestural input to trigger actions. A characteristic property of such symbolic gestures is their *learned* social and culturally dependent meaning, which makes them unambiguous. One — from the gestural standpoint — contrasting idea was followed by Weimer und Ganapathy [19]. They analyzed translatorical aspects of arm movements to create and to modify curves in space. Another approach to specify objects or locations using gestures is found in the ICONIC system by Sparrell and Koons [7][16], which is remarkable for the utilization of *iconic gestures* (gestures that describe shape) to specify objects. Cavazza et al. [5] explicitly concentrate on multimodal input for VR-setups. They define the term *extended pointing* to describe the selection process of one or more objects with pointing gestures and criticize WIMP-style interaction in Virtual Environments (VE). Lucente [11] renounces exoskeletal sensors. He uses camera input and allows multimodal object selection (by pointing), dragging and scaling (iconic gestures).

The usefulness of multimodal interaction for graphics- or VR-applications could be approved by many authors regarding the previously cited work. But one important aspect did not find any attention, namely how to incorporate speech and gesture driven interaction under *general* VR-conditions. Considering the evolution of VR-toolkits or 3D graphics formats, it is quite obvious that there is a trend towards a generalisation in terms of output and dataflow specification. Standards like Java 3D, VRML97 [4] or X3D [18] allow a sophisticated view-model that can, e.g., handle arbitrary display devices. Scene graph structures with specialized node-types (e.g., input and sensor nodes) together with a general field concept provide the possibility to specify dataflow in a declarative and portable manner. Similar approaches can be found as well in research toolkits like AVANGO [17] etc.

To allow multimodal interaction the following prerequisites have to be satisfied: We need a method to recognize speech as well as to detect and analyze gestures. In addition, we need an integration scheme for both modalities that enables the modelling of current research results on multimodal correlations. Furthermore, we have to enrich the virtual scene with linguistic and functional knowledge about the objects to allow the interpretation of multiomodal utterances. And last but not least should the above functionalities be embedded with respect to general state-of-the-art VR-principles. In this article we will focus on a modular gesture detection and evaluation system applicable in the context of VR-environments. This system — and its undelying PrOSA[8] (Patterns on Sequences Of Attributes) concepts — is a basic building block for ongoing research on multimodal VR-based interaction in our lab.

3 Gesture detection

3.1 Actuators and attribute sequences

One of the basic problems dealing with gesture detection in a VR-setup arises from the amount of possible different sensor hardware devices each with its own sampling rate. In addition, for exoskeletal sensors there is no common agreement about the sensors fixation position. In contrast — considering the gesture detection task — there seems to be more and less significant body points for specific gesture movements (e.g., the fingertips [14]). To overcome these limitations and to establish general input data for gesture detection, an abstraction layer is needed. So-called **actuators** [9][8] perform the following actions:

1. Synchronization
2. Representation to a common base
3. Preprocessing: Transformation etc.
4. Qualitative annotation

Synchronization is needed to achieve simultaneous data samples. Their sample times will constitute the evaluation times of the whole gesture detection. To latch this process into a running render loop, the actual data is provided in **attribute sequences**, containers that hold multiple annotated data samples and provide them for higher-level processing units (see next section). An actuator can depend on one or more asynchronously accessed sensors (or **channels**). Typical actuator output can be a specific bodypoint, a joint angle, an alignment value or a direction (e.g., to calculate a pointing direction with respect to the arm elongation, the hand posture and the head-arm distance) depending on the considered gesture to detect.

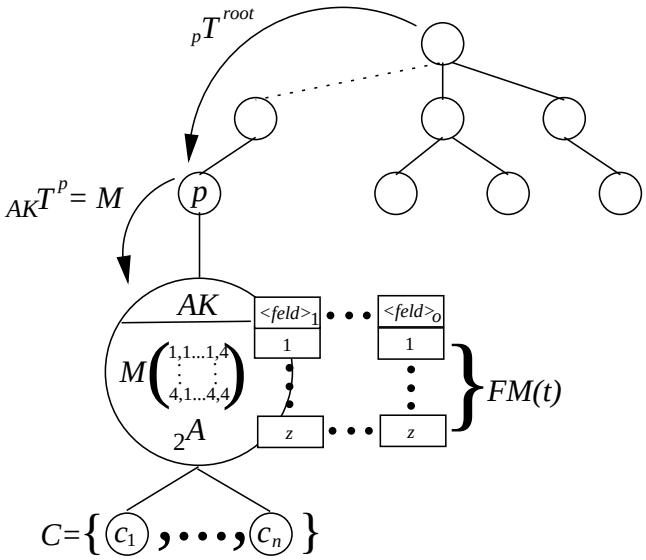

Figure 3: A scene graph node instance of an actuator as defined in [8]. $FM(t)$ represents an atribute sequence for a specific time t, which in general VR applications will be associated with the application stage.

Different actuators share information about their respective channel(s) and sample rate(s) to allow for the determination of a mutual evaluation time, usually by interpolating the data sets at the specific time spots. Actuators can — but do not have to — be instantiated as scene graph nodes (see figure 3). This allows easy adjustment to different position/orientation sensor coordinate systems in a specific application and hardware setup as well as to add required pre- and post-transformations, e.g., to transform from sensor fixation position to fingertips using additional data-glove information.

[3] Like signs of the diver language.

3.2 Detectors and detector nets

Gesture processing takes place for each actuator sample time. The detection and movement analysis is defined by logical combinations of symbolic descriptions of typical gesture features where the latter can be expressed on the lowest level as raw numerical calculations and approximations.

$$
\begin{aligned}
\text{HOLDS?}\,(\texttt{Orbitting}, i) = \\
&\text{HOLDS?}\,((\texttt{AvgSpeed} > 0.15), i) \\
\text{and}\ &\text{HOLDS?}\,((\texttt{SegAng} > 4), i) \\
\text{and}\ &\text{HOLDS?}\,((\texttt{SegAng} < 50), i) \\
\text{and}\ &\text{HOLDS?}\,((\texttt{SegAngSpeed} < 0.1), i) \\
\text{and}\ &\text{HOLDS?}\,((\texttt{NormAngSpeed} < 1), i) \\
\text{and}\ &\text{HOLDS?}\,((\texttt{NormAngAccel} < 0.1), i)
\end{aligned}
\tag{1}
$$

Equation 1 is an example of a simple definition of a hand rotation movement. The term HOLDS? is a predicate stating that `Orbitting` is `true` during the intervall i. The different `AvgSpeed, SegAng,...` are data samples from actuators or other *underlying* detectors. Rules of the above form are translated into networks consisting of detector nodes with simple processing capabilities. Groups of nodes — detector nets — establish new functional units and can be seen as large detectors (with hidden processing rules). The interchange between these detector modules — the *routing* — is again done using attribute sequences. The current implementation[4] utilizes a field concept for the interconnection of detector nodes and nets.

Two detailed examples will explain how the rules for detecting a pointing gesture and the extraction of linear hand movements (which may be used during iconic gestures to express *shape*) can be translated into detector nets. The following legend in figure 4 shall clarify the upcoming diagrams where the node icons will be supplemented with simple descriptions about their internal functions:

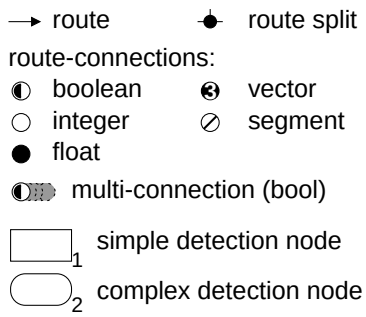

Figure 4: Legend for the following detector figures.

Figure 5 shows how the handshape of a typical (western) pointing gesture can be detected. On the input (top) side data about the finger stretching as well as a parameter threshVal for tuning the detector sensitivity is received by route connections from a handform-actuator or defined in advance respectively.

Node 1 calculates the minimum value regarding the extension of the middle finger, the ring finger and the pinkie. The index finger value is then subtracted from the result with node 2 and compared to threshVal in node 3. In consequence, isPointing? is a trigger that switches true or false depending of the performed calculations.

Figure 6 is a more sophisticated example consisting of three subnets (two of them will be explained below). The prominent inputs

[4]The current framework is implented using the AVANGO [17] toolkit developed by the GMD in germany.

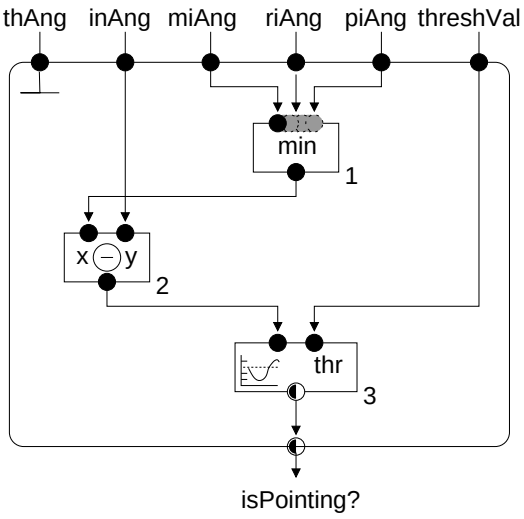

Figure 5: A detector for a pointing posture.

receive a position (e.g., in the middle of the palm) and an associated time. Node 1 smoothes the input. Nodes 2, 5 and 7 are detector nets themselves. Nodes 3 and 4 classify significant movements according to the velocity of the specific point. The overall output generates a segment from the beginning to the end of a stroke in case of a linear movement together with a trigger that reflects the actual detection result. Because human movements are seldomly executed — in a strict geometric sense — perfectly, several tuning parameters allow the modification of the detectors.

Figure 7 unfurls (net)-node 2 from figure 6. The actual movement direction vector is calculated, normalized and smoothed. This direction is compared to directly preceeding directions. In case of an alignment, the isLinear? output will be true.

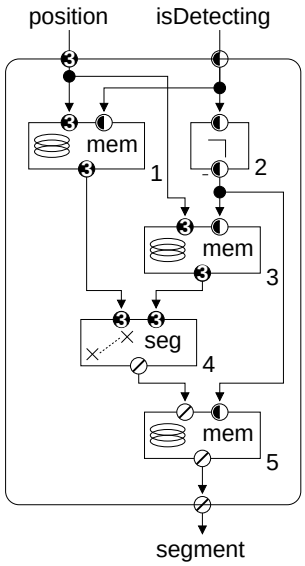

Figure 8: Building a stroke segment.

Another net-node is depicted in figure 8. What has to be done at this stage in the detector processing flow is to register all points and combine them with the results from the detectors for constant simple linear movement and significant movement velocity. This

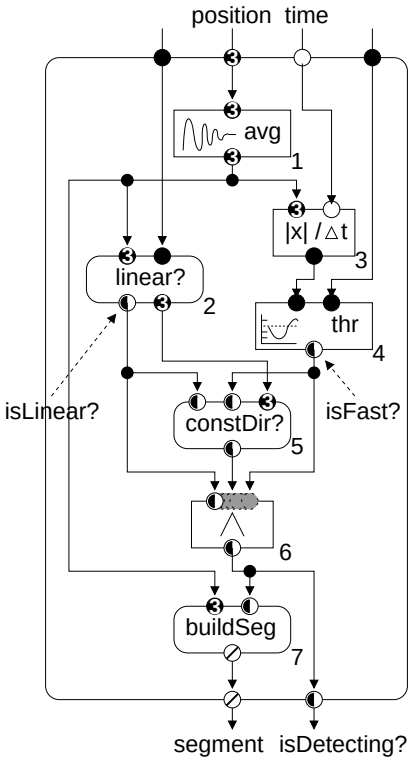

Figure 6: Detecting linear hand movements.

information is generated by the conjunction node 6 from figure 6 which triggers when to extract the stroke start (node 1, figure 8) and end (node 2 and 3, figure 8) points. Whenever the detection triggers false — when the linear movement stops — the actual stroke is registered in the stroke memory by the connection from node 2 to node 3 in figure 8.

3.3 Pre-evaluation using raters

Actuators and detectors establish the basis for the analysis and classification of movement in terms of gesture descriptions. But in VR, a user performs movements and gestures not in a vacuum but more or less immersed in a virtual scene. The user is surrounded by a (partly) artificial world with objects that establishes a virtual reference system. The semantics of multimodal utterances must be seen and interpreted with respect to this world. E.g., if the user looks around, verbal deictic expressions like "*...the left blue pipe...*" and all sorts of gestures which relate to the objects and space around him/her can only be understood in relation to the actual (at the time the utterances took place) environment configuration. This is extremly crucial when we take moving objects — a continuously changing scene — into account. Against this fact, an interpretation of multimodal input will most often have a latency that arises from speech input or grammatical analysis or similar processes. By the time we have a semantic input representation, the indexical integrity of deictic utterances can not in general be guaranteed.

To handle this problem, new node type concepts are introduced into the scene graph: **raters**. They are directly connected to specific reference data for a pre-evaluation and preprocess the surrounding scene. As an example the users view direction should be considdered as a ray based directly between both eyes. Regarding verbal deictic references, this view direction is evaluated for each simulation frame in advance. The objects are sorted according to their distance to the ray basepoint and ray direction. The resulting ordering is stored in a so-called **spacemap** (see figure 9) that buffers spatial scene configurations over the time[5]. The same is done for pointing directions and other important body reference rays (e.g., palm normals). This informations can now be collected by special scene graph traversers that disambiguate deictic references.

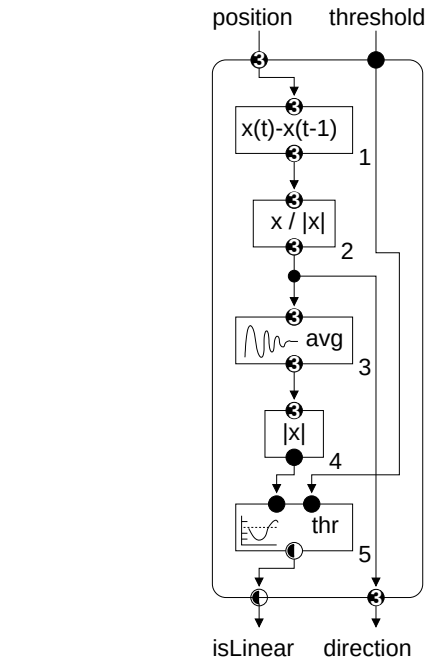

Figure 7: Detecting linearity.

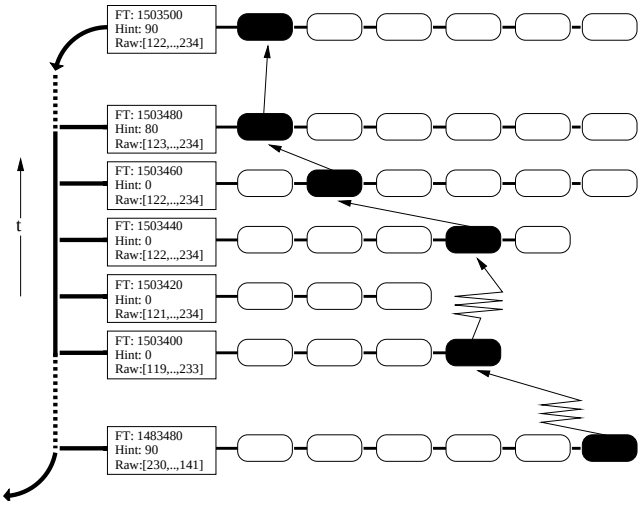

Figure 9: Sorting objects with a spacemap for later evaluation.

4 Summary

The given examples point out how movement patterns can be seen as a logical conjunction of different spatio-temporal gesture features. On the lowest level, these features are constituted by raw mathematical calculations regarding specific body configuration

[5] In the current system the buffer size is around 10 seconds.

data. During the presentation of the gesture processing framework, the term *node* was choosen with respect to a) the integration of the nodes into node-networks and b) the possibility to attach actuators and gesture processing nodes to current available scene graph toolkits and their respective node entities. Combined with the field-based implementation of connection routes, the PrOSA concepts offer a general way to integrate gesture detection into existing VR-setups. The latter was particular important for the presented work on the gesture processing framework and is an underlying principle for related work (e.g., on the multimodal integration and understanding modules) to allow a flexible application of the developed concepts.

One overall goal is to find out[6] what kind of gesture types are used particulary often in conjunction with speech. Work by Nespoulous and Lecour [13] on gesture classification distinguishes *deictic*, *spatiographic*, *kinemimic* and *pictomimic* as subtypes of *coverbal illustrative* gestures. Their functional gesture definition offers a good grounding for the study of corellation between linguistic and gestural counterparts regarding expressions with spatial content. These examinations lead to an idea about possible standard interactions that are expressable with multimodal communication. Prominent examples of currently implemented interactions can be seen in figures 1, 2, 10 and 11.

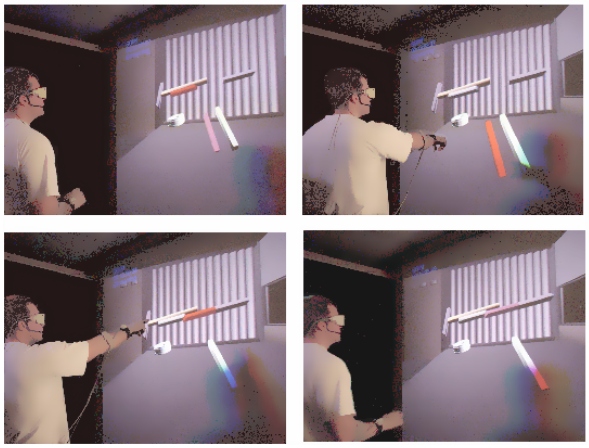

Figure 10: Connecting two pieces with gestures and speech during a construction task with a discrete (one step) interaction. In this example this is done using the verbal command "*Connect this thing with the gray bar over there...*" together with accompanying pointing gestures.

Figure 11: Rotating an object using a *mimetic*[8] gesture. The user says "*...and turn it this way...*" and describes the desired manipulation in parallel. The system continously converts the unprecise arm movement into a precise object manipulation.

[6] E.g., by experiments [15] done in our lab

5 Current and future work

The processing and evaluation of multimodal utterances requires to incorporate ontological prerequisits. Research on natural language processing (NLP) points out that important linguistic as well as semantic or conceptual knowledge is necessary for a successful interpretation of language input. Thus this is as well true in the multimodal — speech and gesture — case. There must be a way to express knowledge about the virtual environment and the objects in the scene. Important information should be representable at least about the objecttypes (maybe by a hierarchy), their attributes, relations between objects and about possible functions (in some cases roles) objects can have during the interaction with the system. And it would be convenient to handle linguistic knowledge using the same methods, the latter to simplify modifications of the knowledge base. Once a common representation structure is established, scene objects can be augmented with additional semantic information.

The currently implemented system represents object data in two different ways. Some object attribute knowledge is stored as frame-like attachements directly at the nodes themselves. More application specific knowledge for the virtual construction scenario about possible connection *ports* between objects and object roles during the construction process as well as some linguistic information is handled by an external knowledge base. One important task right now is to unify all kinds of semantic scene descriptions in a portable manner. Current work considers a special type of a *semantic net* as a candidate for this representation task. Together with a declaration of needed node-based access functions, more general evaluation traversers are currently developed which — as a final goal — enable portability of a multimodal interface between different VR-applications by enhancing scene graph structures with components for gesture/speech processing and for the evaluation of multimodal utterances.

References

[1] K. Böhm, W. Broll, and M. Sokolewicz. Dynamic gesture recognition using neural networks; a fundament for advanced interaction construction. In S. Fisher, J. Merrit, and M. Bolan, editors, *Stereoscopic Displays and Virtual Reality Systems, SPIE Conference Electronic Imaging Science & Technology*, volume 2177, San Jose, USA, 1994.

[2] K. Böhm, W. Hübner, and K. Väänänen. Given: Gesture driven interactions in virtual environments; a toolkit approach to 3D interactions. In *Interfaces to Real and Virtual Worlds*, 1992.

[3] R. A. Bolt. Put-That-There: Voice and gesture at the graphics interface. In *ACM SIGGRAPH—Computer Graphics*, New York, 1980. ACM Press.

[4] Rikk Carey, Gavin Bell, and Chris Marrin. Iso/iec 14772-1:1997 virtual reality modeling language (VRML). Technical report, The VRML Consortium Incorporated, 1997.

[5] Marc Cavazza, Xavier Pouteau, and Didier Pernel. Multimodal communication in virtual environments. In *Symbiosis of Human and Artifact*, pages 597–604. Elsevier Science B. V., 1995.

[6] A.G. Hauptmann and P. McAvinney. Gestures with speech for graphic manipulation. *International Journal of Man-Machine Studies*, 38:231–249, 1993.

[7] D.B. Koons, C.J. Sparrel, and K.R. Thorisson. Intergrating simultaneous input from speech, gaze and hand gestures. In *Intelligent Multimedia Interfaces*. AAAI Press, 1993.

[8] Marc Erich Latoschik. *Multimodale Interaktion in Virtueller Realität am Beispiel der virtuellen Konstruktion*. PhD thesis, Technische Fakultät, Universität Bielefeld, 2001.

[9] Marc Erich Latoschik, Bernhard Jung, and Ipke Wachsmuth. Multimodale Interaktion mit einem System zur Virtuellen Konstruktion. In Klaus Beiersdörfer, Gregor Engels, and Wilhelm Schäfer, editors, *Proc, der 29. Jahrestagung der Gesellschaft für Informatik - Informatik'99, Informatik überwindet Grenzen*, pages 88–97, Berlin Heidelberg New York, 1999. Springer-Verlag.

[10] Britta Lenzmann. *Benutzeradaptive und multimodale Interface-Agenten*. PhD thesis, Technische Fakultät, Universität Bielefeld, 1998.

[11] Mark Lucente, Gert-Jan Zwart, and Andrew D. George. Visualization space: A testbed for deviceless multimodal user interface. In *Intelligent Environments Symposium*, American Assoc. for Artificial Intelligence Spring Symposium Series, March 1998.

[12] Mark T. Maybury. Research in multimedia an multimodal parsing and generation. In P. McKevitt, editor, *Journal of Artificial Intelligence Review: Special Issue on the Integration of Natural Language and Vision Processing*, volume 9, pages 2–27. Kluwer Academic Publishers Group, 1993.

[13] J.-L. Nespoulous and A.R. Lecours. Gestures: Nature and function. In J.-L. Nespoulous, P. Rerron, and A.R. Lecours, editors, *The Biological Foundations of Gestures: Motor and Semiotic Aspects*. Lawrence Erlbaum Associates, Hillsday N.J., 1986.

[14] Claudia Nölker and Helge Ritter. Detection of fingertips in human hand movement sequences. In Ipke Wachsmuth and Martin Fröhlich, editors, *Gesture and Sign-Language in Human-Computer Interaction: Proceedings of Bielefeld Gesture Workshop 1997*, number 1371 in Lecture Notes in Artificial Intelligence, pages 209–218, Berlin Heidelberg New York, 1998. Springer-Verlag.

[15] Timo Sowa and Ipke Wachsmuth. Coverbal iconic gestures for object descriptions in virtual environments: An empirical study. Technical report, Collaborative Research Centre "Situated Artificial Communicators" (SFB 360), University of Bielefeld, 2001.

[16] Carlton J. Sparrell and David B. Koons. Interpretation of coverbal depictive gestures. In *AAAI Spring Symposium Series*, pages 8–12. Stanford University, March 1994.

[17] Henrik Tramberend. A distributed virtual reality framework. In *Virtual Reality*, 1999.

[18] http://www.web3d.org. WWW.

[19] D. Weimer and S.K. Ganapathy. Interaction techniques using hand tracking and speech recognition. In M.M. Blattner and R.B. Dannenberg, editors, *Multimedia Interface Design*, pages 109–126. ACM Press, 1992.

[20] http://www.techfak.uni-bielefeld.de/techfak/ags/wbski/wbski_engl.html. WWW.