Temporal Symbolic Integration Applied to a Multimodal System Using Gestures and Speech

Timo Sowa, Martin Fröhlich*, and Marc Erich Latoschik**

AG Wissensbasierte Systeme
Technische Fakultät, Universität Bielefeld
Postfach 100 131, D-33501 Bielefeld, Germany
e-mail: {tsowa, martinf, marcl}@TechFak.Uni-Bielefeld.DE

Abstract. This paper presents a technical approach for temporal symbol integration aimed to be generally applicable in unimodal and multimodal user interfaces. It draws its strength from symbolic data representation and an underlying rule-based system, and is embedded in a multi-agent system. The core method for temporal integration is motivated by findings from cognitive science research. We discuss its application for a gesture recognition task and speech-gesture integration in a Virtual Construction scenario. Finally an outlook of an empirical evaluation is given.

1 Introduction

Today's computer system users demand for interfaces which are easy to use and easy to learn. To cope with that demand, research in intelligent human-machine interfaces has become more important in the last few years. Therefore our group works to bridge the gap between the user and the machine through a mediating system which translates the user's input into commands for the machine and vice versa.

We focus on problems where command languages or WIMP (Windows, Icons, Mouse, and Pointing device) interfaces show their limitations most drastically, i.e. in interactive 3D computer graphics or Virtual Reality. It is desirable here to address a system in a natural manner, for example by allowing natural language and gestural utterances as input, because command languages are far too difficult to learn and to use. WIMP interfaces tend to overload the user with thousands of functions hidden in hierarchical menu structures. In contrast, it is a comparatively simple task for human beings to describe a 3D scene and to reference objects within that scene using gestures and speech.

While human utterances naturally consist of different modalities which are gathered by various sensor systems, we have to integrate the information from those channels into one single utterance to interpret what the user does and what he says. To fulfill

* Scholarship granted by "Graduiertenkolleg Aufgabenorientierte Kommunikation" of the Deutsche Forschungsgemeinschaft (DFG)

** Supported by the Ministry of Science and Research (MWF) of the Federal State North Rhine-Westfalia in the framework of the collaborative effort "Virtual Knowledge Factory"

the integration task we propose a generic, easily adaptable approach, based on a hierarchical, symbolic data representation as introduced in [4]. In the processing stages from descriptive to semantic representation, background and context knowledge has to be used. In addition to that, based on cognitive findings (e.g. Pöppel [14], Ballard [1]), we propose a data model which is visualised as an extruded triangle as shown in Fig. 1.

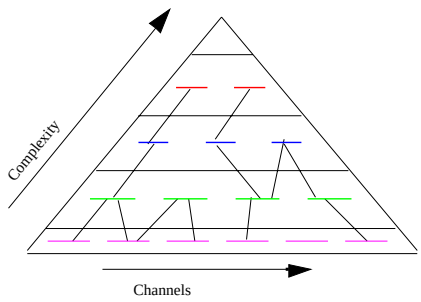

Fig. 1. Integration Hierarchy (further explanation see text), taken from [4]

Input data and model internal data can reach much vaster dimensions than in traditional interface techniques, hence it is a computational time consuming task. Therefore we use a distributed multi-agent system, in which agents are assigned to individual data sources to immediately analyze the incoming information.

This paper concentrates on the integration framework and its application. In Sec. 2, we describe the theoretical foundations of the integration framework and its implementation. In Sec. 3, we present an example implementation of a gesture recognizing multi-agent system based on the framework. In Sec. 4, we describe its general application in a complex system for Virtual Construction and assembly. Finally an outlook of an empirical evaluation is given in Sec. 5.

2 Integration Framework

2.1 Existing Approaches

Although there is no standardized mechanism for multimodal integration, most existing approaches are built on temporal coincidence. Following this basic principle, all fragments of utterance from different modalities are time-stamped and considered for integration if they are close enough in time. Nigay and Coutaz [13], for example, implemented a generic framework for multimodal temporal integration. They distinguish between three fusion steps in a data structure called *melting pot*, where the first two levels are based on temporal neighbourhood. Mircotemporal fusion combines data with nearly the ”same” timestamp, macrotemporal fusion is used in case of temporal proximity, whereas contextual fusion is time-independent and merges data according to semantic constraints. The melting pot itself is a data structure which consists of entries that make up the instruction for the application system. Johnston et al. [7] evaluate temporal

proximity with a time window of 3 to 4 seconds. Their system combines pen-gestures and speech input for a map-based application. Besides the introduction of a time window they also consider the precedence of gesture over speech, which was ascertained in an empirical study. They use this knowledge to refine the integration mechanism.

A system for the recognition and integration of coverbal depictive gestures with speech (ICONIC) is presented by Sparrell et al. [16]. In their approach the gesture representation is based on features, computed from the raw data of the cyber-glove and position tracker operating in 3D-space. In this system integration is speech-driven: If a word or phrase can be augmented by a gesture, the interpreter searches for suitable segments in the gesture stream. A segment matches if its stroke phase (the expressive part of a gesture) is temporally close to the word or phrase under consideration.

All proposals and systems have in common that integration of multimodal data is performed on the top-level within a fixed temporal window [2]. Since integration steps can be found on many levels of the integration triangle (Fig. 1), we developed a method that is applicable to many tasks. We use a common representation scheme for the different types of data and a rule-based integration mechanism. Our method is implemented in a program that we call *Integrator Agent*.

2.2 Symbolic Representation and Symbol Hierarchies

In our approach a symbolic data representation is used to apply common processes of integration in all levels of the hierarchy.

The symbols are organized in a conceptual hierarchy according to the superordinate relations between them. This allows an efficient and short notation, because we can use superconcepts for the expression of generally applicable knowledge. Since we deal with time-critical and often uncertain input, each symbol is augmented by a time interval, which represents the lapse of time of the symbol's validity, and a confidence value, which can be used to model vague concepts and uncertainty. Any kind of symbol shares these properties, so we created a common superconcept, called *Hypothesis*. Specialized subconcepts of *Hypothesis* may be, for example, hypotheses about the speech input, hypotheses about gestures and their features, or hypotheses about fixations of an object recorded with an eye-tracker.

2.3 Rule-Based Integration

Background knowledge about the relationship of different symbols is used in the step of integration which is henceforth called *integration knowledge*. To provide a flexible framework for integration tasks, we have chosen a rule-based approach to express integration knowledge. Following we give a simple natural language example for such a rule:

> *If the index finger is stretched, and if all other fingers are rolled (pointing hand-shape) and if the hand simultaneously is far away from the body, then we have a pointing gesture.*

Production systems are an appropriate means to cope with such kinds of rules. Their core component consists of an inference engine, that matches preconditions of a rule-set against the knowledge that is currently present in memory and executes the consequences (the rule ”fires”). The rule in our example will fire if two symbols for “pointing handshape” and “far away from body” are present in memory and their temporal relation (i.e. cooccurrence) is fulfilled. After execution, a new symbol for “pointing gesture” is present in the memory. It may be used as a command for an application or as a basis for further integration steps, if “pointing gesture” is one of the preconditions for another rule.

Using a rule-based system supports modularity since every rule is an encapsulated piece of knowledge and it shortens develop-and-test cycles. This enables the system designer to experience with different rules and to think about the system design on the task-level rather than on the implementation level. A drawback is the complexity in the execution stage. Since many of the symbols satisfy rule preconditions, the number of rule executions increases with the number of symbols. This leads to an exponential complexity for the execution cycle, usually seen as a drawback per se. What we have done to alleviate this effect will be discussed in the next part.

2.4 Alleviating Complexity by Using Time Windows

Experimental results from cognitive psychology and linguistics suggest that temporal integration in the human brain obeys limitations as a matter of principle. Pöppel [14] emphasizes, for example, that a fusion of successive states of consciousness is possible up to a threshold of about three seconds. This period of time characterizes the subjective presence. McNeill [12] proposes the concept of “growth points” that represent the semantic content of an utterance from which gestures and speech develop in close relation. He suggests a temporal displacement of approximately one or two seconds between two successive semantical units. Similarly, Ballard [1] presents an organization of human computation into temporal bands of 10 seconds for complex tasks, 2 seconds for simple tasks, 300 ms for physical acts, etc. Different tasks and acts – like moving the eyes or saying a sentence – show a tightly constrained execution time.

Based on these results, we can conclude that there is no need to keep each symbol forever. Obviously the importance of a symbol decreases with time and the system can remove the symbol if the timespan from assertion time until “now” exceeds a certain threshold. For the analysis of the current input, it makes no difference if the user has stretched his index finger five seconds ago. The timespan of memorizing symbols depends on their degree of *semantic content*. More complex symbols have a larger temporal scope. Therefore the system enables the designer to adjust the size of the integration window and to build a hierarchy of Integrator Agents with different temporal windows. Fig. 2 shows the general structure of an Integrator Agent. The introduction of a limited temporal integration helps to alleviate the effects of the exponential complexity during the execution cycle. Since nonrelevant symbols are removed from memory, they are not considered for pattern-matching anymore. Additionally, the symbols with high semantic content tend to be sparse compared with low-level symbols. This compensates the larger temporal window of a high-level Integrator.

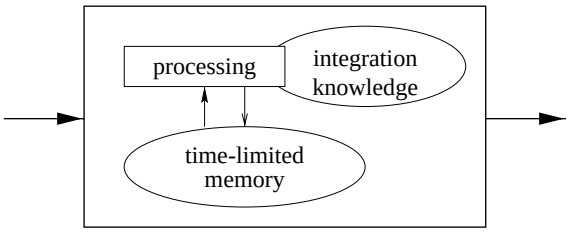

Fig. 2. Integrator Agent: general structure

2.5 Implementation

The implementation of the Integrator Agency is based on *CLIPS (C Language Integrated Production System)*, which is an expert system tool developed by the Technology Branch of the NASA/Lyndon B. Johnson Space Center since 1986. The core component of CLIPS is a production system that directly supports knowledge representation with independent rules, and pattern-matching of the rule preconditions with the RETE algorithm. In addition, conceptual hierarchies can be modeled within the CLIPS language, since it contains an object-oriented part, called *COOL (CLIPS Object Oriented Language)*. COOL-classes are used to represent the concepts, whereas the concrete symbols are represented with instances of COOL-classes. The representation of the user, for example, stretching his index finger may be an instance of the class `IndexStretched`. Its CLIPS-notation may look like this:

```
(ins42 of IndexStretched (begin-timestamp 764538752 909821234)
                         (end-timestamp 764538753 870426876)
                         (confidence 0.7))
```

In this case, `IndexStreched` is a subclass of `Hypothesis`, and `ins42` is one concrete instance of this class. The components `begin-timestamp`, `end-timestamp` and `confidence` are inherited from `Hypothesis`. Of course a class may be augmented with additional components.

We embedded the CLIPS functionality in a C++ class `Integrator` and added the temporal limitation of the integration window. Furthermore we provided methods to cope with rhythmical information like beats, that originate from independent sources. These can be used as a basis for future developements. As a prototypical example we implemented a gesture segmentation cue based on hand tension [5] to get a first impression of the beat/rhythm guided integration.

The Integrator itself was embedded in a further C++ class `IntegratorAgent` which enables the program to communicate with other distributed agents. These agents may be specialized recognizers that process input data from physical devices, they may be other Integrator Agents with another time window, or, finally, the application itself.

3 Gesture Recognition Using HamNoSys

The framework described above can be used for integrating atomic-gesture symbols to complex-gesture symbols (unimodal integration), or for integrating symbols from

different data sources, i.e. complex-gesture symbols and symbols delivered by a speech sub-system (multimodal integration). In this section we present an example of unimodal integration for gesture recognition.

To recognize gestures of the upper limbs we use a system [4] that encodes sensor features taken from 6 degree of freedom position sensors and data gloves into symbols. The set of symbols is a formalized subset (HNS') of the Hamburg Notation System (HamNoSys) [15] which encodes the *form*, not the meaning of a gesture. The features used in this HNS' notation are the same features used in the recognition process.

In this section we will not deal with the interpretation of a gesture, that is left to higher levels of the integration hierarchy.

Both HamNoSys and HNS' represent postures and motions of the upper limbs by a special set of characters. For example the diver's "all clear" gesture (also interpretable as a very rude one in different context) is encoded "∂ᵣ⁻ᵤ⁻" and a "push away" or "stop here" gesture is encoded "' ⅲ⁻ᵤ⁻", as shown in Fig. 3.

Fig. 3. Gesture and its HamNoSys encoding: ' ⅲ⁻ᵤ⁻

Each noteworthy feature of the represented gesture is encoded in a subset of semantically and therefore syntactically disjunct symbols. Groups – which can be seen as words – of such symbols, each taken from a different set, then encode a whole gesture.

If our recognizing agent [4] detects a feature set, it submits this HNS' represented hypothesis to the distributed Integrator Agency and an Integrator Agent will receive that message. After that, the integration process takes place as introduced above, with the additional knowledge of HNS"s syntax and semantic. The semantic of HNS' is defined by its semantic function, mapping HNS' words to their meaning, which is a subclass of class `Hypothesis` of the Integrator.

Besides a number of symmetry and grouping operators the feature focused here is the transition from one state (posture) to another one: $z_{pre}, z_{post} \in L : z_{pre} \rightarrowtail z_{post}$. This introduces a notation of concurrence (concatenation of symbols) and posteriority ($\rightarrowtail$ operator). These temporal constrains are enforced by the temporal integration functions SEQUENCE and PARALLEL which operate inside the integration window only.

3.1 From Atomic-Gesture Symbols to Complex-Gesture Symbols

Every hypothesis used by an Integrator Agent is an instance of the `Hypothesis` class. The pattern-matching for the preconditions of the CLIPS rules checks if instances of certain classes are currently present in memory. The definition of the classes from a specific domain belongs to the integration knowledge, which is stored in a domain specific knowledge base.

For HNS' we derived the class `HNSHypothesis` from `Hypothesis` as the new base class of all subsequent classes. `HNSHypothesis` is augmented by a slot called `hand`, which indicates whether the dominant (usually right) or the subordinate hand is meant. The higher level symbols (hypotheses) are defined by a rule constructor using implementations of the PARALLEL and SEQUENCE operators (here using CLIPS Object Oriented Language, COOL), such as:

```
expression = method superclass {"(" expression ")"} | class
method     = <COOL method; yields instance of type "superclass">
superclass = <COOL class>
class      = <COOL class>

i.e.: Wave = PARALLEL Hypothesis LocStretched
            (SEQUENCE Hypothesis LocLeft LocRight)
```

Analyzing this structure, it is easy to see that this scheme can be cascaded to construct symbols of increasing levels of complexity. To limit the memory size and the message flow, we use a runtime task oriented top down symbol definition structure. That is, every agent located higher in the hierarchy informs its inferiors which symbols it can handle, and only those symbols are taken into account by the inferior agent and are reported back to it's superior. The evaluation of the gesture recognition system is currently in progress. We plan to use data from a series of experiments (see Sec. 5) to test the performance and recognition rates on a set of manually defined gestures.

4 Applications of the Integration Framework

In the previous sections we introduced a formal method to merge symbolic information, i.e. HNS' hypotheses about gesture events, resulting in combined and hence more complex gesture symbols and therefore gaining a higher level of abstraction.

As we pointed out, specific rules could be defined to implement relation tests between temporal properties of hypotheses. The described framework itself is not limited to only these tests, fundamentally any kind of integrative work could be done after defining the required relation tests and the resulting event(s). To use the already defined rules, it is favourable to adopt the system to areas that comprise a hierarchy of symbols and sub-symbols with a basic structure ordered according to the symbols' temporal appearance.

4.1 Exploring New Interaction Techniques

One primary goal of our work was to establish a system for a gesture detection task and to use it for the integration of gestures and speech. Considering a stand-alone gesture

detection that could be used for an automatic sign-language recognizer, the usefulness seems obvious, for instance: to support disabled people or to operate systems in noisy industry environments. In order to further emphasize the importance of these new interface techniques we also have to take a look at areas where such interaction seems advantageous [19]. Dealing with this manner, one of our specific goals is the exploration of advanced human-computer interfaces in real operable systems.

Fig. 4. A user – located in front of a large screen display – performs a selection and a rotation task of an object while interacting with the Virtual Construction application.

In the SGIM project (Speech an Gesture Interfaces for Multimedia) we investigate the benefits of multimodal input during the process of Virtual Construction [8]. Once a user is not bound to Cathode Ray Tube -centered workplaces, either using large screen displays – like we utilize in our specific setup – using Head Mounted Displays or when operating systems that lack any visual feedback, for example embedded systems in the household, interacting with the system becomes a burden when it is still based on common user interface devices: keyboard and mouse and their 3-D equivalents.

4.2 Gesture Detection in Virtual Construction

When the user is immersed in the virtual scene and surrounded by visualized objects, there is in fact a limited and well defined set of interaction elements that are required during this type of instructor/constructor setup [9]. To communicate changes of the actual scene, a primary step is to identify objects and locations in the virtual space. Deictic gestures, by means of pointing to objects [10], is a natural way how humans can refer to spatially arranged items around them.

The evaluation of a pointing gesture is separated into two single tasks. First of all, the qualitative analysis triggers the gesture event, meaning its temporal occurrence. In case the major concern is just the detection of a gesture, in other words to determine just the time of occurrence and the type of a gesture, nothing else is to be done. Contrary to that, our goal is the evaluation of deictic and mimetic gestures to manipulate a virtual scene, hence utilizing the quantitative aspects of gestures.

This is achieved in a following processing step. In the case of a pointing gesture detection, the system follows an imaginary beam rooted at the users' limb pointing. If an object lies in the beam path, an object-reference hypothesis containing this object and the describing information, like its color, is generated. Selecting objects is just the first step in interacting with the system. Further manipulations of objects, in terms of rotating or moving them, must be possible. Current work enhances the gesture detection modules in SGIM with quantitative methods for identifying these geometric transformations.

4.3 Multimodality: Combining Gestures and Speech

In SGIM the gesture interpretation is combined with a speech input and recognition system. Deictic gestures, for instance, are supported using verbal object descriptions. For the basic groundwork of this task we use a commercial (Dragon Dictate) as well as a non-commercial tool that is developed by the "Applied Computer Science" group of our department. Whereas the first one detects only single words, forcing the user to concentrate on a proper pronunciation and slow speech generation, the second one is capable of continuous, user-independent speech recognition [3], a vital requisite for an intuitive interface. Both tools deliver just plain text as output, which now is processed and further analyzed.

To achieve a satisfying and fast respond of our system, and in contrast to a full semantic language understanding approach, every detected word is classified to its affiliation as soon as it is recognized. In addition to name objects or to describe the types they belong to, objects attributes like color and position serve as a major source of referential information. Examples of typical speech fragments during this *selection task* encompass phrases like: *"take the upper red wheel"*, *"connect this thing with"* or *"put this front cover over there"*. The word-spotter module performs a word-match test with its internal database. In this pool all the different object- and typenames and their relations, as well as attributes and words for location references are stored. If a word matches e.g. the color *red*, we carry out two actions:

1. Search if the last word generated is an object-reference hypothesis; if so, further specialize it, and enrich it with the new content (color *red*).
2. If there is no pending object-reference hypothesis then generate a new one.

4.4 Using the Basic Framework During Temporal Integration

The resulting hypotheses generated from both, gesture and speech input streams, have the same level of abstraction and hence the same format. At this integration level, the source of the information only plays a minor role, namely to check if both modalities have produced at least one hypothesis to form a system command. Now both streams are equally taken into account and support their semantic interpretation with new potential: on the one hand, precarious information from one source, e.g. missing or wrong words during the speech recognition process, can be compensated for by the other source and vice versa. On the other hand, redundant information as well can be used to amplify the

probability of a specific hypothesis. To achieve this, we are working on the application of our standard framework for this task.

The described method is obviously underspecified for a complete automatic integration pass. Therefore our current research focuses on the estimation and designation of adequate time intervals we have to take into account during the integration. Where are the start- and endpoints, and how long does it take the user for a complete coherent interaction? As described in section 2.3 and in [11], first attempts used fixed temporal frames with a certain length based on cognitive findings. Furthermore there are two other interesting temporal aspects of multimodal utterance.

The first one is to exploit segmentation cues like the measured hand-tension [5] during gesticulation; this parameter changes significantly between different gestures and, therefore, could be used as a hint to determine the beginning and end of an utterance. The evaluation of the hand-tension cue in a corpus of experimental data (see Sec. 5) shows first promising results. The second aspect does not assume a fixed temporal interval predetermined by the system designer but is based on a different pattern. It is noteworthy that in particular, human gesture and speech production seems to be linked closely together in a rhythmic fashion. Gesture strokes and speech timing as well as accentuation are closely correlated. Thus it seems to be another promising way to exploit rhythmic coherence for its usefulness in gesture and speech integration [18]. If we succeed in extracting the basic rhythmic pattern from user input, we are going to add adequate rules to the integration system. The basic framework developed so far is already capable of handling this type of information.

5 Experimental Evaluation

To evaluate our approach of solving the correspondence problem of multimodal integration (i.e. which gesture or feature belongs to which word or phrase) and to test our implementation, experimental data is needed. Although some insights from psychology and linguistics can be applied, experimental results in these fields mostly refer to narrative discourse. Additionally, experiments yielding quantitative results, for example about the timing of pointing gestures and corresponding words, are not appropriate for our 3D VR scenario. Hence, we collected empirical data about speech and gestures using a similar setting as used in our virtual construction application. In a first stage 37 subjects (26 male, 11 female, age 21-49 years) were told to name simple virtual objects on a wall-size display and to point at them. In a second stage the subjects had to describe more complex virtual objects. Handshape and hand/body positions were registered using data-gloves and 6DOF-trackers, speech was digitally recorded and the subjects were videotaped.

The evaluation of the data is currently in progress. Results concerning timing issues of speech and gesture will be used to refine the rules for integration. Results about the different shapes of gestures used will be utilized to improve the gesture recognition.

6 Conclusion

In this paper we presented a framework for unimodal and multimodal integration of time based symbolic information and gave some supplying examples. We showed how insights taken from cognitive science led us to a symbolic data representation and a rule-based system which we embedded in a multi-agent system. Furthermore we described how we applied our integration framework on the wider context of the SGIM project to illustrate its usability. In the future we will experiment with different time windows, add various segmentation cues, and try to exploit rhythmic coherence for the benefit of the integration task.

References

1. Dana H. Ballard. *An Introduction to Natural Computation*. MIT Press, Cambridge, MA, USA, 1997.
2. C. Benoit, J. C. Martin, C. Pelachaud, L. Schomaker and B. Suhm. Audio-Visual and Multimodal Speech Systems. In D. Gibbon (Ed.) *Handbook of Standards and Resources for Spoken Language Systems - Supplement Volume*, to appear.
3. G.A. Fink, C. Schillo, F. Kummert, and G. Sagerer. Incremental Speech Recognition for Multimodal Interfaces. In *IECON'98 - Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society* [6], pages 2012–2017.
4. Martin Fröhlich and Ipke Wachsmuth. Gesture recognition of the upper limbs: From signal to symbol. In Wachsmuth and Fröhlich [17], pages 173–184.
5. Philip A. Harling and Alistair D. N. Edwards. Hand tension as a gesture segmentation cue. In Philip A. Harling and Alistair D. N. Edwards, editors, *Progress in Gestural Interaction: Proceedings of Gesture Workshop '96*, pages 75–87, Berlin Heidelberg New York, 1997. Dep. of Computer Science, University of York, Springer-Verlag.
6. IEEE. *IECON'98 - Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society*, volume 4, Aachen, September 1998.
7. M. Johnston, P. R. Cohen, D. McGee, S. L. Oviatt, J. A. Pittman and I. Smith. Unification-based Multimodal Integration. *35th Annual Meeting of the Association for Computational Linguistics, Conference Proceedings*, pages 281–288, Madrid, 1997.
8. Bernhard Jung, Marc Erich Latoschik, and Ipke Wachsmuth. Knowledge based assembly simulation for virtual prototype modelling. In *IECON'98 - Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society* [6], pages 2152–2157.
9. Marc Erich Latoschik, Martin Fröhlich, Bernhard Jung, and Ipke Wachsmuth. Utilize speech and gestures to realize natural interaction in a virtual environment. In *IECON'98 - Proceedings of the 24th Annual Conference of the IEEE Industrial Electronics Society* [6], pages 2028–2033.
10. Marc Erich Latoschik and Ipke Wachsmuth. Exploiting distant pointing gestures for object selection in a virtual environment. In Wachsmuth and Fröhlich [17], pages 185–196.
11. Britta Lenzmann. *Benutzeradaptive und multimodale Interface-Agenten*, volume 184 of *Dissertationen zur Künstlichen Intelligenz*. Dissertation, Technische Fakultät der Universität Bielefeld, Infix Verlag, Sankt Augustin, March 1998.
12. D. McNeill. *Hand and Mind: What Gestures Reveal about Thought*. University of Chicago Press, Chicago, 1992.
13. L. Nigay and J. Coutaz. A generic Platform for Addressing the Multimodal Challenge. *Human Factors in Computing Systems: CHI '95 Conference Proceedings*, pages 98–105, ACM Press, New York, 1995.

14. Ernst Pöppel. A hierarchical model of temporal perception. *Trends in Cognitive Sciences*, 1(2):56–61, May 1997.
15. Siegmund Prillwitz, Regina Leven, Heiko Zienert, Thomas Hanke, and Jan Henning. *HamNoSys Version 2.0: Hamburg Notation System for Sign Languages: An Introductory Guide*, volume 5 of *International Studies on Sign Language and Communication of the Deaf*, Signum Press, Hamburg, Germany, 1989.
16. C. J. Sparrel and D. B. Koons. Interpretation of Coverbal Depictive Gestures. *AAAI Spring Symposium Series*, pages 8–12. Stanford University, March 1994.
17. 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, Berlin Heidelberg New York, Springer-Verlag, 1998.
18. Ipke Wachsmuth. Communicative Rhythm in Gesture and Speech. This volume.
19. Alan Daniel Wexelblat. Research challenges in gesture: Open issues and unsolved problems. In Wachsmuth and Fröhlich [17], pages 1–12.