Modeling Multimodal Integration with Event Logic Charts

Gregor Mehlmann
Human Centered Multimedia
Augsburg University
Universitätsstraße 6a
86159 Augsburg, Germany
mehlmann@hcm-lab.de

Elisabeth André
Human Centered Multimedia
Augsburg University
Universitätsstraße 6a
86159 Augsburg, Germany
andre@hcm-lab.de

ABSTRACT

In this paper we present a novel approach to the combined modeling of multimodal fusion and interaction management. The approach is based on a declarative multimodal event logic that allows the integration of inputs distributed over multiple modalities in accordance to spatial, temporal and semantic constraints. In conjunction with a visual state chart language, our approach supports the incremental parsing and fusion of inputs and a tight coupling with interaction management. The incremental and parallel parsing approach allows us to cope with concurrent continuous and discrete interactions and fusion on different levels of abstraction. The high-level visual and declarative modeling methods support rapid prototyping and iterative development of multimodal systems.

Categories and Subject Descriptors

D.1.0 [**PROGRAMMING TECHNIQUES**]: General

Keywords

Multimodal Fusion, Interaction Modeling, Event Logic Chart

1. INTRODUCTION

During the last decades, researchers explored ways of enhancing human-computer interaction by developing multimodal dialogue systems. These systems overcome the shortcomings of unimodal dialogue systems that provide only a restricted set of communication modes and devices. They exploit the major characteristics of human interaction which is the coordinated use of multiple modalities such as speech, gestures, gaze, facial expressions and body postures. In the course of this research, it has been demonstrated that users often prefer multimodal interfaces over conventional graphical or unimodal user interfaces, because multimodal interfaces offer a more natural and effective interaction and better flexibility, adaptability and reliability [18, 19].

A critical challenge in the development of robust multimodal dialogue systems is the interpretation of the users' intention by integrating and understanding inputs distributed over multiple modalities. The key function of the modality fusion is the reduction of uncertainty and the mutual disambiguation of the various analysis results by combining partial semantic information provided by each modality [14].

Due to recent technical progress, today there is a new generation of devices on the market that are available and affordable for a large number of users and can be integrated more and more into our daily life. Such devices support features, such as markerless full body motion sensing [1], mobile eyetracking [2] and mobile biosignal capturing for affective interfaces [3]. These devices are boosting the evolution towards an unrestricted natural interaction through multimodal dialogue systems.

However, the recent technical development also poses new conceptual and technical challenges to the capabilities of multimodal fusion engines that have to be overcome to exploit the full potential of multimodal interaction. These are challenges such as the concurrent processing of discrete and continuous interaction forms, the contemporary and incremental parsing and fusion of user input as well as the regression analysis of input data for disambigation, reduction of confounding factors and a more precise decision making. Many state-of-the-art fusion engines do not address these challenges or are just able to cope with a part of these in very specific settings while lacking to present a uniform formalism or representation language for multimodal integration. In this paper we present a novel approach to a unified formalism for multimodal integration and interaction management that tackles these challenges.

2. THE CHALLENGES

In this section we present some major challenges for multimodal fusion engines and present an overview of terms and concepts of our solution approach.

2.1 Problem Description

Many multimodal fusion engines are restricted to a special combination of modalities or require some kind of primary modality. However, in a reusable and flexible formalism for multimodal fusion, all modalities should be treated equally, so that there exist no restrictions for the combinations of

[1] http://www.xbox.com/kinect
[2] http://www.smivision.com
[3] http://www.alivetec.com

different modalities and devices. For example, for the interpretation of sign-language, a primary speech modality obviously makes no sense, but it is important to capture the complex synergy between the various nonverbal signals.

Another shortcoming of many multimodal fusion engines is that they exclusively focus on discrete interaction in form of gestures or spoken commands that are processed after being completed. For example, pointing gestures are used to resolve deictic references in speech after information from both modalities has been received. However, in order to directly manipulate objects in the user interface with continuous movements, it is necessary to contemporarily process dynamic data and to directly forward it to the application while still allowing the concurrent interpretation of discrete interaction.

In order to gain user acceptance and to avoid annoying situations, the system's response times to the user's actions must be reasonably small and feedback needs to be continuously provided. For example, if the system takes too long to react to a command, the user may assume that the command was wrong or not understood. The user could repeat the command and would be confused when the command then gets carried out twice. Providing early feedback to the user creates better transparency and comprehension of the interaction. It also allows an early error recovery in the case of device recognition errors or demands of the system for refinement of user input and explicit ambiguity resolution. For this reason, user input needs to be parsed, fused and processed contemporarily and incrementally by the system.

Sometimes it is not sufficient to analyse only the user's last action and the momentary situation. For example, when regarding continuously provided data from modalities such as gaze, emotions or biological signals, the inspection of the data's time course can reduce the influence of device recognition errors or momentarily recognized unusual behavior of the user and, thus, resolve ambiguities and allow more precise decisions. For this reason, a fusion engine needs to be able to fall back on information about the user's past interaction and to infer generalized and fuzzy knowledge about the interaction history.

Finally, multimodal fusion can be realized during various processing stages [20]. Apart from a few exceptions, most fusion engines operate on only one of those abstraction levels for multimodal fusion. Offering fusion support across different levels of abstractions can help exploit the power of multimodal interactions.

2.2 Solution Approach

Our solution approach to these challenges pursues most widely declarative and visual modeling paradigms and exploits uniform representation languages. Multimodal fusion mechanisms are modeled with declarative specifications in a *multimodal event logic*. This event logic handles all modalities equally and allows us to express a diversity of temporal, spatial and semantic relations between events. It allows us to express structural and functional constraints for the unification of partial information distributed over events from multiple devices and modalities.

Context knowledge is modeled in a *knowledge base* in the form of semantic networks and type hierarchies. This knowledge is directly translated to predicates of the multimodal event logic, so that logic inference enables us to resolve references and to infer semantic relations between entities of the application context or the semantic content of events. The knowledge base enfolds an *event history* recording incoming events in a short-term memory. Together with generalized and fuzzy quantifier predicates of the event logic this creates an intuitive mechanism for ambiguity resolution and regression analysis.

The incremental step-wise multimodal parsing and fusion as well as the behavioral aspects of interaction management are modeled with a parallel and hierarchical *state chart language*. Transitions of the state chart model can be annotated with event logic constraints and are taken whenever such a formula is satisfied. It also allows the modeling of parallel fusion processes on different levels of abstraction and the simultaneous processing of discrete and continuous interactions through synchronized automata.

In the remainder of this paper we first discuss related work in Section 3 and investigate how existing approaches address the above mentioned challenges. In Section 4 we introduce the theoretical background and conceptual framework of our approach and present some illustrative examples that give a good idea of how the approach can be applied for the development of multimodal applications. In Section 5 we explain how the conceptual framework has been realized in a technical framework and which advantages it offers to designers of multimodal applications. Finally, in Section 6, we discuss the conceptual and technical contributions of our work and give a view on potential future directions.

3. RELATED WORK

Since the seminal work of Bolt [2], there have been researched various approaches to multimodal fusion engines [12]. In early work [17, 5] multimodal integration was essentially a procedural extension of an existing speech understanding or text understanding engine. These systems did not allow incremental or parallel parsing of input and, thus, did not allow continuous interaction or multiple parallel interpreation processes.

The first uniform representation formats were introduced by Koons et. al. [11] and Voo et. al. [22], where parallel unimodal parsers translated information from individual modalities to a common *frame-based* representation format. Multimodal fusion was reached by merging those semantic frames together in order to produce the combined interpretation. A more declarative approach was developed by Johnston [10] where multimodal integration was modeled as unification of *typed feature structures* [4]. However, this approach only handeled simple interactions with a single gesture in combination with speech and was therefore extended [7] utilizing techniques from natural language processing, such as *unification-based multimodal grammars* and chart parsing to enable handling of inputs with more than one gesture and more flexible and declarative encoding of temporal and spatial constraints. This approach was the first that allowed incremental parsing and the generic use of arbitrary modalities in contrast to earlier mainly speech-driven approaches. However, it did not address the challenge of the concurrent use of continuous and discrete interaction and had no concept for an event history.

While these unification- and frame-based approaches separate the parsing of the individual modalities and multimodal integration into separate processing steps, Johnston and Bangalore [9] described a highly efficient approach to multimodal understanding with *finite-state automata* in which

a single multimodal grammar specifies the integration and understanding of multimodal language. However this approach lacked the specification of complex temporal, spatial and semantic constraints and did not allow continuous and incremental parsing.

Other approaches take a more process-oriented approach to model the dynamics and behavioral aspects of multimodal interaction with *Petri nets* [16] or *temporally augmented state-transition networks* [13]. Issues such as parallelism, sequencing or synchronization of actions can be described by these models. These approaches are also able to cope with discrete and continuous interaction and allow incremental parsing, but they do not enable fusion on different levels of abstraction.

While the above mentioned works are exclusively concerned with multimodal fusion on the decision-level, recent work by Hoste et al. [6] introduces for the first time an approach that deals with the parallel fusion on different levels of abstraction. This approach is based on a declarative *rule-based* language operating over a common fact base. However, this approach does not address incremental parsing and interaction management.

The above investigated approaches have proven to be well suited for certain applications, but each of them misses to meet some of the requirements described in Section 2. The approach described in this paper goes beyond the current state-of-the-art since it enhances well proven concepts of the previous approaches and introduces new concepts to tackle the mentioned challenges. In the next section, we describe the details of the theoretical background and the conceptual framework of our approach.

4. CONCEPTUAL FRAMEWORK

In this section we present some major concepts of our multimodal event logic and show how it can be used in conjunction with state charts for incremental parsing and fusion of multimodal user input.

4.1 The Multimodal Event History

Multimodal events are represented as feature records that can carry a variety of information from different levels of abstraction. They can carry low-level data such as the coordinates describing the position of the user's gaze, but, also high-level information such as the semantic content of a speech act, the referenced object of a pointing gesture or information about the types, phases and the spatial extent of gestures. For illustration purposes, but without limiting the generality, in the following we assume a basic set of the most general features for a multimodal event as shown in Definition 1. This notion of an event can easily be adapted to existing standards [8] or to application specific requirements.

Definition 1. Let Σ be an alphabeth then a multimodal event is defined as a record $\langle n, m, s, e, d, c\rangle$ so that $n \in \Sigma^+$ is the identifier, $m \in \Sigma^+$ is the modality, $s, e \in \mathbb{N}_0^+$ are the start- and end times, $d \in \Sigma^*$ is the semantic content and $c \in]0, 1] \subset \mathbb{R}$ is the confidence value of the event.

Due to varying modality specific recognition times and device response times, the fusion engine receives events from different recognition modules with different delays. To handle those events in a sequence, which corresponds to the real chronological sequence, the fusion engine records them to an *event history*. The definition of a well-formed event history is presented in Definition 2.

Definition 2. A finite set $KB \subset \{\langle n, m, s, e, d, c\rangle : n, m \in \Sigma^+, d \in \Sigma^*, s, e \in \mathbb{N}_0^+, c \in]0, 1]\}$ is a well-formed event history if and only if each event in the event history has a unique identifier.

Logic inference on the event history allows to answer a diversity of queries concerning its content, such as the existence of events with specific attributes or temporal, spatial and semantic relations between events. The event history enfolds a *short-term memory* mechanism, which deletes events that have reached a certain age so that they are not considered anymore by the inference engine.

$$event(sevt_{23}, speech, 1050, 1450, put, 0.85)$$
$$event(sevt_{24}, speech, 1650, 2050, that, 0.85)$$
$$event(sevt_{25}, speech, 2250, 2750, there, 0.80)$$
$$event(gevt_{354}, gesture, 1750, 1850, ball_3, 0.95)$$
$$event(gevt_{355}, gesture, 2300, 2400, table_2, 0.95)$$
$$event(gevt_{356}, gesture, 2450, 2550, table_2, 0.85)$$
$$event(gevt_{357}, gesture, 2600, 2700, floor_1, 0.90)$$

Figure 1: Exemplary events of an event history.

Figure 1 shows an extract of a well-formed event history containing simple time-stamped event records with semantic contents and confidence values obtained by the modality specific recognition modules. This event history describes a scenario in which the user's voice and full body motions are observed in an environment containing the objects $table_2$, $ball_3$ and $floor_1$. The user gives the spoken command *put that there* and points to the ball and then to the table. During the spoken word *that* the system recognizes a pointing to $ball_3$, however, due to the user's hastily and imprecise movements the system recognizes ambiguous pointings to $table_2$ and $floor_1$ during the word *there*. This could also be caused by the recognition mechanism of the tracking device, since, in contrast to i.e. pen-based devices, in full body tracking, it is not always clear when a gesture starts and when it begins, so that some devices deliver continuous tracking samples. How our approach resolves this ambiguity with the help of generalized quantifiers of our event logic is explained in the next section.

4.2 The Event Logic Predicates

Our multimodal event logic includes a variety of first-order predicates to infer and compare attributes of events and to resolve temporal, spatial and semantic relations between individual events. Additionally, it enfolds higher-order generalized and fuzzy quantifiers to make statements about sets of events that share certain features.

4.2.1 Basic Feature Predicates

Basic features of an event record can be referred to with the *feature predicates* that are defined over the domain of event identifiers. These predicates are the foundation of our event logic and allow to infer the features of an event, such as its identifier, the description of its modality, the timestamps

of its start- and end time, information about the semantic content and the corresponding confidence value. They are used to define a variety of *comparison predicates* that allow to compare different events to each other with respect to their basic properties. For example, Definition 3 shows the definition of the comparison predicate *equal_mode* which tests whether two event identifiers n_i and n_j refer to two different events in the event history *KB* and whether their modality descriptions m_i and m_j are equal.

Definition 3. Let *KB* be a well-formed event history and let $n_i, n_j \in \Sigma^+$ then we define *equal_mode* as:

$$equal\_mode_{KB}(n_i, n_j) :\Leftrightarrow \exists m_i, m_j \in \Sigma^+. \begin{pmatrix} name_{KB}(n_i) \wedge name_{KB}(n_j) \wedge n_i \neq n_j \wedge \\ mode_{KB}(n_i, m_i) \wedge mode_{KB}(n_j, m_j) \wedge m_i = m_j \end{pmatrix}$$

4.2.2 Temporal Event Relations

Time is a key concepts to be represented in order to combine several events received from multiple modalities. Quantitative time allows to represent temporal evolutions related to a given amount of time or at a precise moment in time whereas qualitative time addresses temporal relations between events and the ordering of event such as precedence, succession and simultaneity. To define *qualitative temporal predicates* we refer back to Allen's time intervals [1] as shown in Definition 4.

Definition 4. Let *KB* be a well-formed event history and let $\langle n_i, m_i, s_i, e_i, d_i, c_i\rangle, \langle n_j, m_j, s_j, e_j, d_j, c_j\rangle \in KB$ then we define the quantitative temporal predicates:

$$\begin{aligned}
before_{KB}(n_i, n_j) &:\Leftrightarrow s_j > e_i \\
meets_{KB}(n_i, n_j) &:\Leftrightarrow e_i = s_j \\
concurrent_{KB}(n_i, n_j) &:\Leftrightarrow s_i = s_j \wedge e_i = e_j \\
during_{KB}(n_i, n_j) &:\Leftrightarrow s_j < s_i \wedge e_i < e_j \\
starts_{KB}(n_i, n_j) &:\Leftrightarrow s_i = s_j \wedge e_i < e_j \\
finishes_{KB}(n_i, n_j) &:\Leftrightarrow e_i = e_j \wedge s_j < s_i \\
overlaps_{KB}(n_i, n_j) &:\Leftrightarrow s_i < s_j \wedge s_j < e_i \wedge e_i < e_j
\end{aligned}$$

We define *quantitative temporal predicates* to make statements about the temporal distance between different events or between events and fixed points in time. Definition 5 shows the definitions of some of our predicates that infer the distance between two events of a lineary ordered set.

Definition 5. Let *KB* be a well-formed event history and let $\langle n_i, m_i, s_i, e_i, d_i, c_i\rangle, \langle n_j, m_j, s_j, e_j, d_j, c_j\rangle \in KB$ with $n_i \neq n_j$ and $t \in \mathbb{N}^+$ then we define the quantitative temporal predicates:

$$\begin{aligned}
dist_exactly_{KB}(n_i, n_j, t) &:\Leftrightarrow e_i < s_j \wedge s_j - e_i = t \\
dist_more_than_{KB}(n_i, n_j, t) &:\Leftrightarrow e_i < s_j \wedge s_j - e_i > t \\
dist_less_than_{KB}(n_i, n_j, t) &:\Leftrightarrow e_i < s_j \wedge s_j - e_i < t
\end{aligned}$$

4.2.3 Event Ordering Relations

Events of the same modality, user and device or multiple events that satisfy a certain constraint can be lineary ordered. In this case, we might be interested in the oldest, latest or just a random event from such a set of events. Additionally, we want to make statements about the ordering of these events, such as determining followers and ancestors or to express neighbourly relations between them. Our multimodal event logic enfolds various *linear ordering predicates* to express these relations. Definition 6 shows the denotational semantics of different overloadings for some of these predicates.

Definition 6. Let *KB* be a well-formed event history and let $\langle n_i, m_i, s_i, e_i, d_i, c_i\rangle, \langle n_j, m_j, s_j, e_j, d_j, c_j\rangle \in KB$ with $n_i \neq n_j$. Let $N \subseteq \{n : \exists\langle n, m, s, e, d, c\rangle \in KB\}$, $n \in N, m \notin N$ and ϕ be a well-formed formula of our event logic with the free variable $\sigma \in \Sigma^+$, then we define the ordering predicates:

$$\begin{aligned}
&oldest_{KB}(n, N) :\Leftrightarrow n \in N \wedge \forall n' \in N.(n \neq n' \Rightarrow before_{KB}(n, n')) \\
&oldest_{KB}(n, \sigma, \phi) :\Leftrightarrow oldest(n, \{n' \in \Sigma^+ : [n'/\sigma]\phi_{KB}\}) \\
&follows_{KB}(n_i, n_j) :\Leftrightarrow equal_mode_{KB}(n_i, n_j) \wedge after_{KB}(n_i, n_j) \\
&follows_{KB}(N, m) :\Leftrightarrow N = \{n' \in \Sigma^+ : follows_{KB}(n', m)\} \\
&nextto_{KB}(n_i, n_j) :\Leftrightarrow follows_{KB}(N, n_j) \wedge oldest_{KB}(n_j, N)
\end{aligned}$$

4.2.4 Types and Sematic Relations

Logic and semantic networks are compatible and supplementing formalisms. Semantic networks are an extension to predicate logic because they can be used to encode static and dynamic aspects of semantic knowledge about the domain, the task, the user as well as the types and attributes of events in the system. They constitute a clearly arranged graphical representation of semantic relations and can easily be maintained with graphical editors.

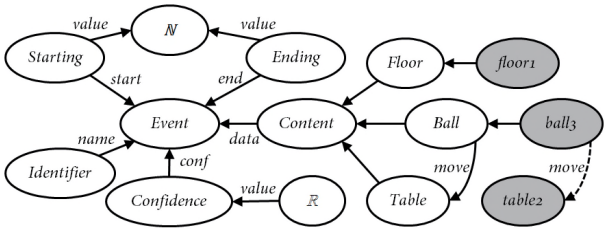

Figure 2: A semantic network describing event features and the *move* relation from balls to tables.

The semantic relations represented by a semantic networks can automatically be translated into entities and predicates of our event logic and be considered by the inference mechanism. Figure 2 shows a simple semantic network as it could be used as part of our knowledge base. It describes the basic features of a multimodal event as well as the semantic relation *move* between tables, balls and the floor.

4.2.5 Generalized and Fuzzy Quantifiers

In addition to infer relations between individual events in the event history, it is very helpful to make statemants about sets of events. For that purpose, we provide a variety of *generalized quantifiers* and *fuzzy quantifiers* for the multimodal event logic. Definition 7 presents the denotational semantics of some of the quantifiers that our event logic provides.

Definition 7. Let *KB* be a well-formed event history and let ϕ and ψ be well-formed formulas of the previously defined

predicates with the free variable $\sigma \in \Sigma^+$. Furthermore, let $x \in \mathbb{N}_0^+$ and $y \in [0,1]$, then we define the following generalized quantifiers:

$$forsome_{KB}(\sigma,\phi,\psi) \quad :\Leftrightarrow$$
$$\{n \in \Sigma^+ : [n/\sigma]\phi_{KB}\} \cap \{n \in \Sigma^+ : [n/\sigma]\psi_{KB}\} \neq \emptyset$$
$$forevery_{KB}(\sigma,\phi,\psi) \quad :\Leftrightarrow$$
$$\{n \in \Sigma^+ : [n/\sigma]\phi_{KB}\} \subseteq \{n \in \Sigma^+ : [n/\sigma]\psi_{KB}\}$$
$$formorethan_{KB}(x,\sigma,\phi,\psi) \quad :\Leftrightarrow$$
$$\left|\{n \in \Sigma^+ : [n/\sigma]\phi_{KB} \wedge [n/\sigma]\psi_{KB}\}\right| > x$$

The major advantage of those quantifiers is that they allow a very intuitive and elegant way of formalizing quantified and fuzzy statements about event sets. This mechanism can be exploited for regression analysis, disambiguation and to reduce the influence of negligible recognition errors.

Find the bindings for variable X, so that
The majority of *gestures* *during sevt25* *refer to*
object X *with a* *confidence higher than 0.8*

$formajor(\sigma,$ $(mode(\sigma,gesture)$ $\wedge$ $during(\sigma,sevt_{25})),$
$(data(\sigma,\ X\ ) \wedge conf\_higher\_than(\sigma,0.8)))$

Figure 3: Resolving ambiguities from the example in Figure 1 with the generalized $formajor$ quantifier.

Figure 3 illustrates the use of the quantifier $formajor$ with the unbound variable X. It shows how a natural language query can be translated straightforward to such a quantifier formula. The logic inference mechanism finds events in the knowledge base that unify with this expression and returns possible bindings for the unbound variable X under which the formula is satisfied. This specific formula describes how the ambiguities of the event history from the example in Figure 1 can be resolved after we have infered that the speech event *there* has identifier $sevt_{25}$. This formula regards only the majority of pointings during the speech event $sevt_{25}$ that refer to the same object with confidence higher 0.8 while neglecting the fact that there has also been a pointing gesture to another object. Therefore, we decide for the object $table_2$ that has been refered to by the majority of pointings during *there* and put object $ball_3$ onto $table_2$.

4.3 The Event Logic State Charts

Beside their general application in the specification of reactive and interactive system behavior, various state chart dialects have been shown to be a suitable method for modeling dialogue and interaction [3, 21]. In the following, we illustrate the integration of our event logic with a state chart language that is used for the incremental parsing and fusion of user input and the parallel management of continuous and discrete interaction.

4.3.1 Integrating State Charts with Event Logic

In this work we use a state chart language that was extended with features to facilitate dialogue and interaction management, such as local variable scoping and the parameterization of hierarchical components. Transitions of our state charts can be annotated with variable assignments and logic formulas of our event logic. Outgoing transitions of a state are taken whenever the logic formula on the transition is satisfied with respect to the current state of the event history. Initially unsatisfied formulas may become satisfyable whenever the event history is enriched with new events, context knowlede has changed or some timeout has been reached. States and superstates contain local variable definitions that can correspond to unbound variables in the logic constraints of a transition. If such a constraint can be satisfied, unification is done, which means that the interpreter binds that variable to the infered value so that the variable is instantiated in consecutive computation steps. When using a variable in its uninstantiated form, the variable is annotated with an upward directed arrow index ($X_\uparrow$). After it has been instantiated it is indicated with a downward directed arrow index within a logic formula ($X_\downarrow$).

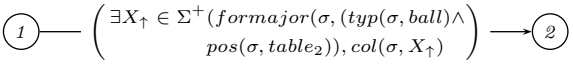

Figure 4: A constraint with an unbound variable.

Figure 4 shows an example of a transition labeled with a constraint containing the unbound variable X. This constraint is satisfied if the majority of objects with type *ball* lying on table $table_2$ have the same color which is then bound to variable X and can be used in the next parsing steps.

4.3.2 Incremental Parsing and Fusion

The mechanism of the step-wise evaluation of logic constraints and the consecutive use of the infered knowledge is exploited for the incremental parsing and fusion. It allows to continuously react to user input by executing system commands in the states of the state chart. Such commands can be used to give early feedback to the user, for example by highlighting entities of the user interface or directly executing parts of the user command. The user can be asked as soon as possible to refine a command if the information was not precise or complete, contains ambiguous statements or stands in contradiction with the current application state.

The example in Figure 5 illustrates the incremental parsing and fusion of multimodal input during the execution of a state chart whose transitions are labeled with logic formulas that need to unify with the content of the event history. It shows the specification of a state chart model which can be used to accept the well-known *"put-that-there"* input pattern [2]. It shows a subautomaton which is able to accept the speech part of that utterance accompanied by pointing gestures to two objects in the environment. We now explain the individual steps in more detail whereby we assume that the incoming events successively combine to the event history shown in Figure 1 and that the context knowledge enfolds the semantic relations encoded in the semantic network from Figure 2.

Initially, the automaton is parameterized from the calling process with the last speech event ℓ that has been processed by the calling process. In state ① the interpreter waits until the event history contains the oldest speech event after ℓ which carries the content *put*. When the event history re-

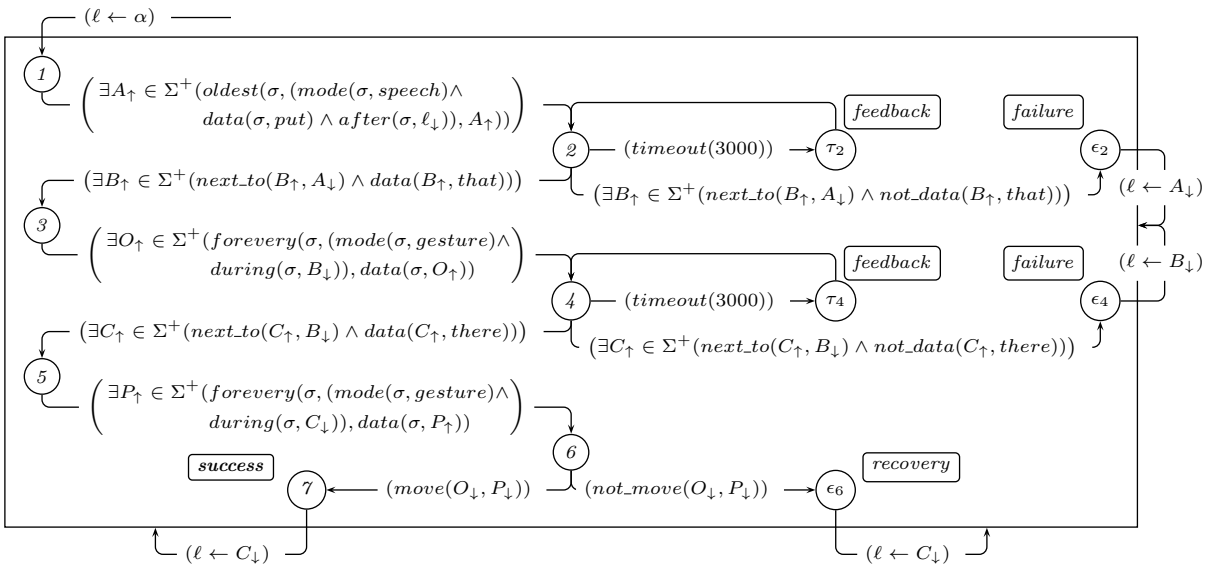

Figure 5: Incremental parsing of Bolt's [2] classical "put-that-there" with state charts and the event logic.

ceives or already contains such an event, then the transition ①→② is taken and the variable A is bound to the identifier of this event. In state ② the interpreter waits until the event history contains a direct successor of the event whose name is now bound to variable A. When the event history receives or already contains such an event, then the transition ②→③ is taken and the variable B is bound to the identifier of the second speech event. In state ③ the interpreter checks if all gesture events during the last speech event carry the same semantic content. In this case, the transition ③→④ is taken and the semantic content of those gesture events is bound to variable O. The transitions ④→⑤ and ⑤→⑥ are evaluated analogue to the transitions ②→③ and ③→④. If the execution reaches state ⑥, then the content of the first set of gestures nas been bound to variable O and the semantic content of the second set of gestures has been bound to variable P. Finally, in state ⑥ the interpreter checks if these semantic contents satisfy the constraint on transition ⑥→⑦ by inspecting the semantic network. This constraint is satisfied if the first set of gestures refer to a movable object (i.e. $ball_3$) and the second set of gestures refer to a valid target position for that object (i.e. $table_3$). Afterwards, the automaton is restarted with the last speech event $C_\downarrow$. The states τ_2 and τ_4 are reached whenever the user waits too long between two words and are used to ask the user for continuation. The states ϵ_2, ϵ_4 and ϵ_6 are reached whenever a constraint can not be satisfied due to a wrong word or the selection of invalid objects.

4.3.3 Continuous and Discrete Interaction

The incremental and parallel parsing and fusion approach using state charts has further major advantages. We are now able to model parallel and synchronized automata that allow us to cope with the concurrent appearance of discrete and continuous interaction forms.

Figure 6 shows an example in which two parallel automata are synchronized in order to switch from a discrete interaction to a continuous interaction and back. During the continuous interaction the automaton that is parsing the discrete interaction pattern remains continuously active. The example describes a scenario in which the user may use the speech commands *drag* and *drop* to take objects and move them around with the help of eye movements using an eye-tracker device. The left parallel automaton is modeling the recognition of the *drag* or *drop* commands. In state ① the interpreter waits until there exists an oldest speech event after ℓ which carries the words *drag* or *drop*. When the event history receives or already contains such an event, then the transition ①→② is taken and the variable X is bound to the identifier of this event. In state ② the interpreter checks if at least 80% of the gaze events during this speech event refer to the same object. The threshold of 80% is choosen here, since eye movements are usually very fast and can be easily distracted, so that there can be a considerable proportion of false measurements that we want to ignore. If the transition ②→③ is taken then the identifier of the focused object is bound to variable Y. Finally, in transition ③→④ the interpreter updates the knowledge base with a new command which contains the triggering speech event and the reference to the focused object.

The automaton on the right side is running in parallel to the left automaton in a concurrent process, waiting for relevant commands. In state ⑤ the interpreter is inspecting the knowledge base for the latest command. If there exists such a command then the transition ⑤→⑥ is taken and the identifier of the event that triggered the command is bound to variable X while the content of this command is bound to variable Y. In state ⑥ the interpreter checks the content of the triggering event and proceeds with transition ⑥→⑦ if the command was a request to drop an object. If the command was a request to drag an object, then the interpreter enters the transition loop ⑥→⑧→⑤→⑥. Within this loop, the interpreter inspects the event history for the latest gaze event and binds the content of the gaze event, which is the position or the object the user is looking at, to the variable P. Afterwards the interpreter checks if the

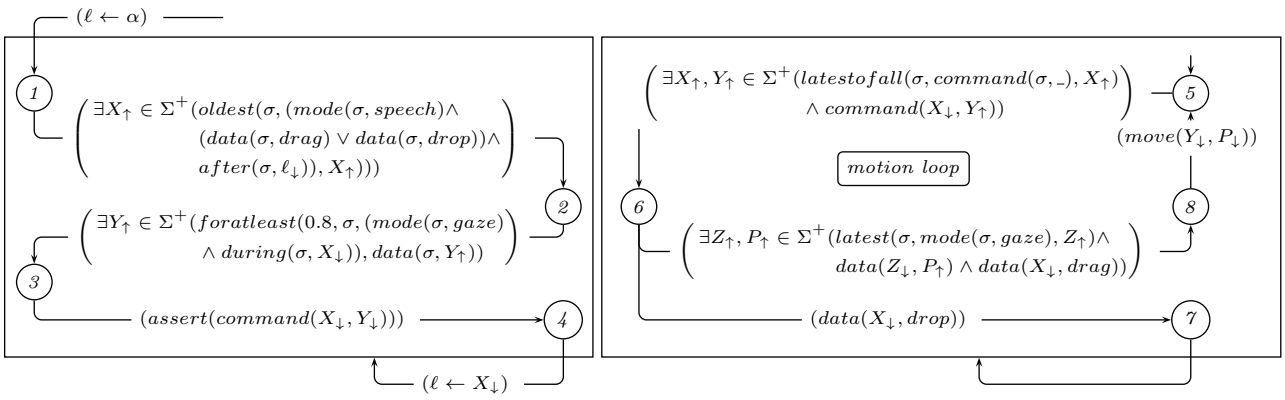

Figure 6: Using parallel processes for the concurrent processing of continuous and discrete interaction.

object reference in Y is movable to position P by inspecting the semantic network, before it looks for the latest command again. A system command within the loop can be used to update the position of the object in the user interface following the user's gaze until the user gives the command to drop the object.

5. TECHNICAL FRAMEWORK

We implemented our multimodal event logic, described in Section 4, and the functionality for the construction and modification of the event history in the logic programming language *Prolog* using the development environment and inference engine of *SWI-Prolog* [24]. The implementation of the event history as part of a Prolog knowledge base and the various Prolog predicates was straightforward. For example, Figure 7 shows the implementation of the *formajor* quantifier in SWI-Prolog.

$$
\begin{aligned}
&formajor(Template, Generator, Condition) :- \\
&\quad bagof(Template, Generator, Range) \\
&\quad bagof(Template, (Generator, Condition), Scope) \\
&\quad length(Range, R), length(Scope, S), S/R > 0.5.
\end{aligned}
$$

Figure 7: The *formajor* predicate in SWI-Prolog.

We used the modeling framework *SceneMaker* [15] for the integration of the event logic with state charts. SceneMaker's plug-in mechanism allowed us to integrate the Prolog inference engine with minimal effort. SceneMaker was very well suited for the rapid realization of our approach since its state chart language allows the *hierarchical refinement* of the model in order to reuse and easily extend already modeled components. *Parallel decomposition* can be used to specify parallel parsing and interaction management in concurrent processes. SceneMaker is implemented in Java and relies on an interpreter approach for the execution of the state charts. Java *reflect invocation* was used to call system commands for feedback and error recovery. SceneMaker's IDE enfolds a *visual state chart editor* and a *runtime visualization* mechanism which facilitates rapid prototyping and testing.

Figure 8 shows parts of the implementation of the example

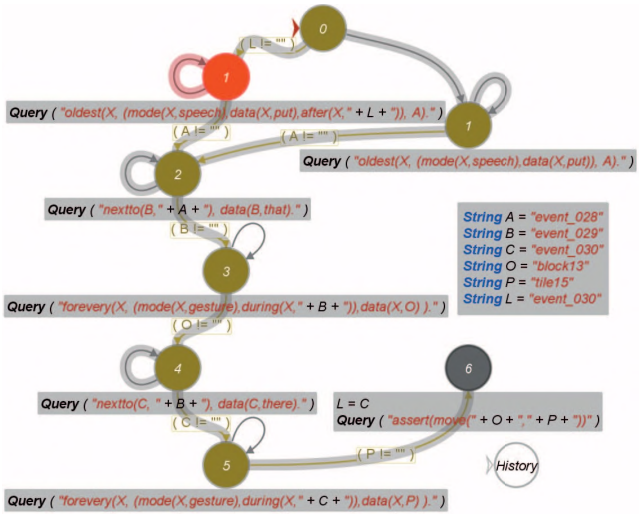

Figure 8: Incremental parsing variant of Bolt's [2] "put-that-there" with SceneMaker and Prolog.

from Figure 5 with SceneMaker. The runtime visualization mechanism is used to highlight states and transitions that are executed during the single parsing steps.

6. CONCLUSION AND FUTURE WORK

In this work we presented a novel approach to the modeling of modality fusion and dialogue management exploiting widely declarative and visual representation languages. The generic and extendable notion of multimodal events avoids restrictions for the combination of different modalities and devices. The event logic defines a variety of functional, temporal and spatial predicates and we use semantic networks for modeling semantic relations and context knowledge. The event history in combination with generalized and fuzzy quantifiers can be used for regression analysis and disambiguation. Our approach allows the contemporary and incremetal parsing and fusion of user input. This is used to realize an early error recovery and feedback delivery to the user. Since we pursue an incremental and parallel parsing approach, we are able to realize the simultaneous processing of discrete and continuous interactions. Our approach also

allows the modeling of parallel fusion processes on different levels of abstraction.

This paper describes only a part of our approach's potential which has additional advantages that will be addresses in future work. We are currently exploring the potential of our approach in processing parallel processes. Processing the same interaction with parallel interpreters allows for the concurrent creation of different interpretations providing different confidence values und, thus, being selected with different probabilities for dialogue management. In collaborative or cooperative settings where the actions of the different users need to occur temporally and spatially aligned to each other, parallel processes can be used to parse the interactions of each user in isolation and synchronization can be used to express the various constraints. We will explore this idea by means of a multi-touch table where multiple users plan a travel together by manipulating multiple objects in parallel using speech and gestures.

In addition to that we are currently working on the development of an integrated framework that unifies the data- and feature-level fusion pipeline of the *SSI* framework [23] with our approach. This will provide application designers with a powerful and flexible processing pipeline from the low-level sensor access over various feature-level fusion and machine learning strategies to high-level fusion, dialogue management and application context maintenance.

7. ACKNOWLEDGMENTS

The work described in this paper is funded by the EU under research grant CEEDS (FP7-ICT-2009-5).

8. REFERENCES

[1] J. F. Allen and G. Ferguson. Actions and events in interval temporal logic. *Journal of Logic and Computation*, 4:531–579, 1994.

[2] R. A. Bolt. Put-that-there : Voice and gesture at the graphics interface. In *Proceedings of SIGGRAPH '80*, pages 262–270. ACM, New York, NY, 1980.

[3] J. Brusk, T. Lager, A. Hjalmarsson, and P. Wik. Deal: Dialogue management in scxml for believable game characters. In *Proceedings of Future Play '07*, pages 137–144. ACM, New York, NY, 2007.

[4] B. Carpenter and F. Pereira. *Computational Linguistics*, volume 19, chapter The Logic of Typed Feature Structures. Cambridge University Press, Cambridge, England, 1992.

[5] P. Cohen. Integrated interfaces for decision support with simulation. In *Proceedings of WSC '91*, pages 1066–1072. IEEE Computer Society Washington, DC, USA, 1991.

[6] L. Hoste, B. Dumas, and B. Signer. Mudra: A unified multimodal interaction framework. In *Proceedings of ICMI' 11*, pages 97–104. ACM, New York, NY, USA, 2011.

[7] M. Johnston. Multimodal language processing. In *Proc. of the Int. Conf. on Spoken Language Processing*, 1998.

[8] M. Johnston. Building multimodal applications with emma. In *Proceedings of ICMI-MLMI '09*, 2009.

[9] M. Johnston and S. Bangalore. Finite-state multimodal integration and understanding. *Journal of Natural Language Engineering*, 11:159–187, 2005.

[10] M. Johnston, P. R. Cohen, D. McGee, S. L. Oviatt, J. A. Pittman, and I. Smith. Unification-based multimodal integration. In *Proc. of the Association of Computational Linguistice*, 1997.

[11] D. B. Koons, C. Sparrel, and K. R. Thorisson. *Intelligent Multimedia Interfaces*, chapter Integrating simultaneous input from spech, gaze and hand gestures, pages 257–276. AAAI, Menlo Park, CA, 1993.

[12] D. Lalanne, L. Nigay, P. Palanque, P. Robinson, J. Vanderdonckt, and J. F. Ladry. Fusion engines for multimodal input: A survey. In *Proceedings of ICMI' 09*, pages 153–160. ACM, New York, NY, USA, 2009.

[13] M. E. Latoschik. Designing transition networks for multimodal vr-interactions using a markup language. In *Proceedings of ICMI' 02*, pages 411–416, 2002.

[14] J. C. Martin. *TYCOON: Theoretical framework and software tools for multimodal interfaces*. AAAI Press, 1998.

[15] G. Mehlmann, B. Endraß, and E. André. Modeling parallel state charts for multithreaded multimodal dialogues. In *Proceedings of ICMI' 11*, pages 385–392. ACM,New York, NY, USA, 2011.

[16] D. Navarre, P. Palanque, R. Bastide, A. Schyn, M. A. Winckler, L. Nedel, and C. Freitas. A formal description of multimodal interaction techniques for immersive virtual reality applications. In *Proceedings of INTERACT '05*, volume 3585 of *LNCS*, pages 170–185. Springer-Verlag, 2005.

[17] J. Neal and S. Shapiro. *Intelligent Unser Interfaces*, chapter Intelligent Multi-Media Interface Technology, pages 45–68. Addison Wesley, New York, 1991.

[18] S. Oviatt. Advances in robust multimodal interface design. *IEEE Comput. Graph. Appl.*, 23(5):62–68, September 2003.

[19] S. Oviatt. *The Human-Computer Interaction Handbook*, chapter Multimodal Interfaces, pages 413–432. Mahwah, NJ: Lawrence Erlbaum and Associates, 2008.

[20] R. Sharma, V. I. Pavlovi, and T. S. Huang. Toward multimodal human-computer interface. *Proceedings of the IEEE, Special Issue on Multimodal Signal Processing*, 86:853–860, 1998.

[21] D. Traum, A. Leuski, A. Roque, S. Gandhe, D. DeVault, J. Gerten, S. Robinson, and B. Martinovski. Natural language dialogue architectures for tactical questioning characters. In *Army Science Conference*, 2008.

[22] M. T. Voo and C. Wood. Building an application framework for speech and pen input integration in multimodal learning interfaces. In *Proc. of the Int. Conf. on Acoustics, Speech and Signal Processing*, 1996.

[23] J. Wagner, F. Lingenfelser, and E. André. The social signal interpretation framework (ssi) for real time signal processing and recognition. In *Proceedings of INTERSPEECH' 11*, pages 3245–3248, 2011.

[24] J. Wielemaker, T. Schrijvers, M. Triska, and T. Lager. Swi-prolog. *CoRR*, abs/1011.5332, 2010.