Multi-Modal Definite Clause Grammar

Hideo Shimazu, Seigo Arita, and Yosuke Takashima
Information Technology Research Laboratories, NEC Corporation
4-1-1 Miyazaki, Miyamae Kawasaki, 216 Japan
{shimazu, arita, yosuke}@joke.cl.nec.co.jp

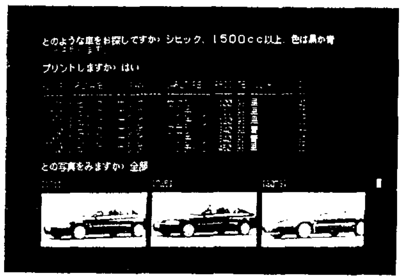
Abstract

This paper describes the first reported grammatical framework for a multimodal interface. Although multimodal interfaces offer the promise of a flexible and user friendly means of human-computer interaction, no study has yet appeared on formal grammatical frameworks for them. We have developed Multi-Modal Definite Clause Grammar (MM-DCG), an extension of Definite Clause Grammar. The major features of MM-DCG include capability to handle an arbitrary number of modes and temporal information in grammar rules. Further, we have developed MM-DCG translator to transfer rules in MM-DCG into Prolog predicates.

1 Introduction

This paper describes the first reported grammatical framework for a multimodal interface. Specifically, the authors have developed MM-DCG (Multi-Modal Definite Clause Grammar), an extension of DCG [Pereira and Warren, 1980] for multimodal input processing. The major features of MM-DCG include capability to handle an arbitrary number of modes and temporal information in grammar rules.

The motivation behind this research was two-fold. First, the extension to multimodal was found to be the minimum requirement for natural language interface systems to be installed in real applications. We have developed natural language interface for relational database (RDB) [Shimazu et. al., 1992; Arita et. al., 1992a; Arita et. al., 1992b]. Spoken user queries are transformed into SQL specifications, and dispatched to RDBMS. The retrieved results are displayed at a computer terminal. The results include not only table forms but also picture images, like Figure 1. When users see picture images on the terminal, they naturally want to generate following queries by referring to such picture images. For example, they want to say, "Show me the interior of this one" or "Are there the same type of cars as this car" while pointing at a specific picture on the display. If such multi-modal utterances be acceptable, the natural language interface will be more practical enough to be used in many real world applications.



とのような車をお探しですか シビック、1500cc以上、色は黒か青

プリントしますか はい

[illegible]

との写真をみますか 全部

Figure 1: Natural Language Interface Screen Image

Second, no study has yet appeard on developing formal grammatical framework for multi-modal interfaces. Although there have been many researches on multi-modal systems, these systems are built as task-specific expert systems. The capability of such systems to process multi-modal inputs is too limited to interpret complex multi-modal expressions. This is mainly due to the fact that they have not developed their systems on formal grammatical framework for multi-modal interfaces.

MM-DCG is the first reported grammatical framework for a multimodal interface. Multi-modal input processing rules can be written in MM-DCG simply and effectively. Rules in MM-DCG are translated into Prolog predicates easily.

2 Multi-Modal Input Processing

Consider a query example to a multi-modal interface with a screen image like Figure 1. A user states "Can this, attach this," pointing at a picture on the screen and clicking the mouse during the first "this" and then choosing an item from a menu during the second. The system must realize that the first point is to a specific automobile and the second is to the menu item "CD player". After integrating the two mouse pointing events into the two "this" in the utterance, the system must create an internal representation of this query that conforms to SQL specifications. In this example, even if the order of the two mouse clicking events is opposite, the system must generate the same SQL specification, but the interpretation will be more difficult. In order to interpret such complex combinations of multi-modal inputs, the following requirements exist:

(1) Modes should be interpreted equally and independently. In conventional multi-modal systems, natural language mode plays a major role, and other modes such as mouse input mode are auxiliary. Inputs of auxiliary modes are merged into corresponding natural language expressions in a surface level, and the merged natural language query is interpreted by conventional natural language parsers. Therefore, variety of accepted multi-modal expressions is very limited.

However, If each mode is treated with the same manner as that of natural language mode, syntax and semantics of inputs of each mode are defined with grammar formulation. Thus, complex expressions can be defined declaratively and more easily.

(2) Mode interpretation should be referred to one another. Inputs of each mode should be interpreted independently. However, the interpretation of such inputs should be referred by other mode interpretations. There are ambiguities which are solved only by integrating partial interpretations of related modes. For example, if user states "this car", pointing at an object which is overlapped on the car object, the ambiguity of the object pointing must be solved by comparing the two mode interpretations.

(3) Mode interpretation should handle temporal information. Temporal information of inputs, such as input arriving time, interval between two inputs, plays an important role to interpret multi-modal inputs. Consider an example that a user states "How much is this car", and points at a car picture a little after the utterance. If the interval is three *seconds*, the pointing event should be integrated with "this car" in the utterance. However, if the interval is three *minutes*, the event should not be integrated.

3 MM-DCG Design Decisions

This section describes major design decisions made in developing MM-DCG. Because MM-DCG is a superset of DCG, everything possible in DCG is also possible in MM-DCG. However, two major extensions are provided

3.1 Receiving Multiple Input Streams

MM-DCG can receive arbitrary numbers of different input streams, while DCG receives only one. Each mode is assigned an individual stream. Therefore, a single grammar rule in MM-DCG can allow the coexistence of grammatical categories in different modes, thus allowing for their integration. In addition, context sensitive information can be interchanged among categories of different modes in a single rule. Figure 2 illustrates a multi-modal input processing module which accepts three independent streams.

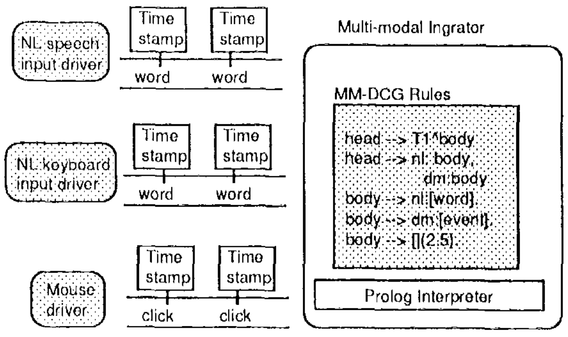


Figure 2: Multi-modal Input Processing Module

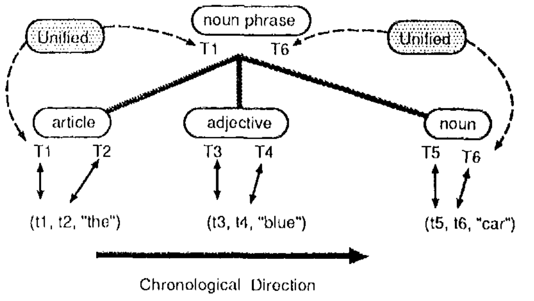


Figure 3: Time Calculation of Instantiated Semantic Categories

3.2 Calculating the Instantiated Time of Grammatical Categories

Inputs of a single mode invariably have ordering relations among them. A parser like DCG uses such order relations to analyze syntax, semantics, and pragmatics. Inputs of different modes, however, have no inherent ordering relations. Therefore, MM-DCG requires the attachment of both the beginning time and the end time to each individual piece of input data. MM-DCG automatically calculates the beginning time and the end time of any level of grammatical categories generated during parsing.

MM-DCG translator automatically generates the code which calculates the beginning and end times of any body goal in a grammar rule. The translator generates two extra arguments to store the beginning time and end time into each head and body goals in MM-DCG rules. The beginning time argument of the head is unified with the beginning time argument of the first body goal. The end time argument of the head is unified with the end time argument of the last body goal. Figure 3 shows the argument organization of noun_phrase rule.

Thus, for example, if a noun phrase category is instantiated by parsing "the blue car", the beginning time of the instantiated category becomes equal to the beginning time of "the", and the end time of the category is equal to the end time of "car".

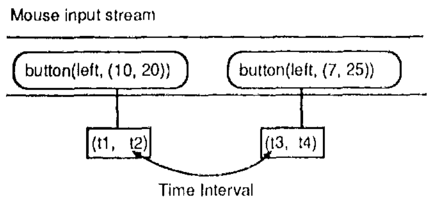


Figure 4: Timeout Concept

MM-DCG requires any input from every mode to have beginning and end times. Thus, each item in an input sequence will have the following structure:

```
input(beginning-time, end-time, <actual input>)
```

which means that the actual input was inputted from start-time and completed at end-time. Adding of this time information is easy for any of the sorts of input modes we are considering (i.e. speech recognition, keyboard inputs, mouse pointing, etc).

One other important item of notation: If a variable is explicitly bound within a goal, the variable returns the beginning and end times of the goal in the form of a functor. Thus,

```
Time^goal
```

means that "if goal succeeds, the beginning time and end time of the goal are returned in the variable Time." Using the time information of instantiated categories, rule writers can define chronological constraints among categories, for example, the following description expresses a constraint that pronoun category and pointing category must be both instantiated within a five seconds,

```
T1^pronoun, T2^pointing,
{Diff is T2 - T1, Diff < 5}
```

3.3 Defining Timeout in Rules

Timeout is a constraint of intervals between an input and its succeeding input of a stream (See Figure 4). If an interval between inputs of a stream becomes larger than a threshold defined in grammar rules, the timeout occurs, and the stream is regarded empty temporarily although there still exist inputs in it.

The following points rule means that "Receive mouse clicking inputs while the interval between two inputs is less than 5 seconds or until a stream becomes null, then return the list of the inputs"

```
points([]) --> mouse:[](5.0).
points([Pt | Pts]) --> point(Pt), points(Pts)
point(Loc) --> mouse:[button(left, Loc)].
```

4 Rules Written in MM-DCG

4.1 Syntax

MM-DCG syntax extends DCG in the following ways:

- A body goal may or may not be specified its consuming stream:

  If a body goal consumes inputs from specific streams, the goal must be accompanied by the stream names. For example, the following rule

  ```
  noun_phrase --> keyboard:pronoun.
  ```

  means that "if the pronoun category is found which is generated by inputs from the keyboard stream, noun_phrase is found." If a body goal is not accompanied by any stream name, the goal is regarded as consuming some amount of inputs from *all* modes. For example, the following rule

  ```
  noun_phrase --> noun.
  ```

  means that "if the noun category is found which is generated by inputs from certain streams, noun_phrase is found."

- A terminal symbol should always be accompanied by a specific stream name:

  For example, the following rule

  ```
  pointing --> mouse:[button(left, loc(X, Y)].
  ```

  means that "if a functor button(left, loc(X, Y)) is found at the mouse stream, pointing is found".

4.2 Rule Example

To demonstrate how MM-DCG rules are written, this section describes a simple grammar needed to handle "object" with multi-modal inputs.

Figure 5 shows the definition of "object". A rule writer defines existing streams specifically using a unit clause, active_stream/1. "Object" are specified by using either one of the above modes or their combinations.

The first object/1 definition interprets natural language specifications such as "the blue car". The second object/1 interprets a mouse clicking which points at a specific graphical object on the display. The third object/1 definition interprets a combination of a natural language utterance and a mouse pointing, such as stating "the blue car" while pointing at a graphical object on the display. A natural language utterance is interpreted at the noun_phrase body goal, and the identified object is bound to Obj1. A mouse pointing event is interpreted at the pointing body goal, and the identified object is bound to Obj2.

Then, Obj1 and Obj2 are compared their values in a Prolog predicate enclosed inside curly brackets { and }. Both variables should be equal. If not, because the interpretation of noun_phrase or pointing must be wrong, backtracking occurs.

As seen above, a single grammar rule in MM-DCG can allow the coexistence of grammatical categories in different modes, thus allowing for their integration. In addition, temporal and context sensitive information can be interchanged among categories of different modes in a single rule.

```
% stream definition
active_stream(speech, mouse, keyboard).
% For natural language mode
object(Obj)  --> noun_phrase(Obj).
noun_phrase(Obj) --> article, adjective(Attr, A_value), noun(Noun),
   {attribute(type, Noun, Obj), attribute(Attr, A_value, Obj)}.

article  --> (speech or keyboard): [the].
adjective(color, blue)  --> (speech or keyboard):[blue].
noun(automobile) --> (speech or keyboard):[car].

% For mouse mode
object(Obj) --> pointing(Obj).
pointing(Obj) --> mouse:[button(left, loc(X, Y))],{attribute(location, (X, Y), Obj)}.

% For combinations of modes
object(Obj1) --> noun_phrase(Obj1), pointing(Obj2), {Obj1 == Obj2}.
```

Figure 5: Grammar Description Example Using MM-DCG

5 Translating MM-DCG into Prolog

This section describes translation techniques of MM-DCG rules into Prolog predicates. First, we explain the translation method of MM-DCG rules with a *single* stream. Even in the single stream case, MM-DCG translation method is different from that of DCG. Then, the translation technique with multiple streams is explained.

5.1 MM-DCG Translation for a Single Stream

A head and body goals in a grammar rule are translated into a predicate with four extra arguments – two for the beginning time and the end time and two for expressing a consumed input stream. The latter two arguments are the same as the generated arguments when DCG rules are translated into Prolog predicates.

The beginning time argument of the head is unified with the beginning time argument of the first body goal, and the end time argument of the head is unified with the end time of the last body goal. For example, the following MM-DCG rule (for a single stream):

```
noun_phrase  --> article, adj, noun.
```

is translates into:

```
noun_phrase(T0, T, N0, N) :-
  article(T0, T1, N0, N1),
  adjective(T2, T3, N1, N2),
  noun(T4, T, N2, N).
```

or, in English,

There is a noun-phrase between N0 and N if there is an article between N0 and N1, and if there is an adjective between N1 and N2, and if there is a noun between N2 and N. The noun-phrase starts at T0, and ends at T. The article starts at T0, and ends at T1. The adjective starts at T2, and ends at T3. The noun starts at T4, and ends at T.

A rule with a terminal symbol is translated into a unit clause. For example,

```
noun --> keyboard:[window].
```

translates into:

```
noun(Ts,Te,[input(Ts,Te,''window'')|N],N).
```

A functor input/3 is inserted into the third argument forming the input stream of the predicate. The third argument of the functor input/3 is the actual input item, the "window" string in this example.

The first and second argument of input/3 is unified with the first and second argument of this unit clause respectively. Therefore, if a string "window" is input via the keyboard stream, the noun category is instantiated, and the beginning and end time of the noun category is the same as the start and end time attached to the "window" input.

5.2 Extension to Arbitrary Number of Streams

Extension from a single stream to multiple streams is easy. Each stream needs four extra arguments - two for timing information and two for expressing a consumed input stream. Thus, if there are n modes, $4n$ arguments are added into head and goals arguments.

For example, if there are two streams, the noun_phrase definition in the previous section is translated into the following prolog predicates with eight (2×4) extra arguments:

```
noun_phrase(Tx0,Tx,Nx0,Nx,Ty0,Ty,Ny0,Ny) :-
 article(Tx0,Tx1,Nx0,Nx1,Ty0,Ty1,Ny0,Ny1),
 adjective(Tx2,Tx3,Nx1,Nx2,Ty2,Ty3,Ny1,Ny2),
 noun(Tx4,Tx,Nx2,Nx,Ty4,Ty,Ny2,Ny).
```

5.3 Extractions of Temporal Information

If there is a variable binding within a goal like,

```
Time^goal
```

the goal is translated into a conjunction of two body goals (for a single mode):

```
(goal(T0, T1, N0, N),Time = (T0, T1) )
```

If there exist n streams, the variable Time is bound to a list of n time pairs, such as for two modes:

```
(goal(Tx0,Tx1,Nx0,Nx1,Ty0,Ty1,Ny0,Ny1),
 Time = [(Tx0, Tx1), (Ty0, Ty1)] )
```

6 Related work

The idea of understanding multi-modal inputs in conjunction with each other, as presented in this paper, is not particularly new. The idea of a multi-modal input combining motions and pointing has been explored in a number of contexts. The classic 1980 paper "Put-That-There" [Bolt, 1980] describes an early system that procedurally combined voice and gesture inputs. This idea was further explored in terms of integrating natural language and pointing by [Hayes, 1988], who related multi-modal inputs to anaphoric reference in natural language processing, particularly to the work of [Grosz, 1977] and [Sidner, 1979]. Recent work in the design of direct manipulation interfaces has also explored the notion of integrating a set of diverse inputs. Other papers exploring multimodal interfaces include [Allgayer et. al., 1989; Cohen et. al. , 1989; Cohen, 1991; Kobsa et. al., 1986; Wahlster, 1989]. Most of this work, however, has focused on the application of the ideas, and not on the principles for integrating the different inputs. [1]

7 Conclusion

In this paper, we have proposed the use of a grammar for dealing with input events in a multi-modal user interface. We proposed MM-DCG, a novel grammatical framework for a multimodal interface. MM-DCG is an extension of DCG for multi-modal inputs processing. The major features of MM-DCG include capability to handle an arbitrary number of modes and temporal information in grammar rules. We showed its use for a simple example. The translation technique of the MM-DCG rules into Prolog predicates was also presented. An initial implementation of MM-DCG has been developed at NEC Corporation, and is currently being used for the development of a prototype multi-modal interface.

References

[Allgayer et. al., 1989] Allgayer, J., Jansen-Winkeln, R., reddig, C., and Reithing N.,

[Arita et. al., 1992a]
Arita, S., Shimazu, H., and Takashima, Y., "Portable Natural Language Interface", Proc. of the 8th Human Interface Symposium, 1992, (in Japanese).

[Arita et. al., 1992b]
Arita, S., Shimazu, H., and Takashima, Y., "Simple + Robust = Pragmatic: A Natural Language Query Processing Model for Card-type Databases", Proc. of the 13th Annual Conference of the Cognitive Science Society, 1992.

[Bolt, 1980] Bolt, R.A., "Put-That There: Voice and Gesture at the Graphics Interface", Computer Graphics 14, 3, 1980.

[Clocksin and Mellish, 1981] Clocksin, W.F. and Mellish, C.S., "Programming in Prolog", Springer-Verlag, 1981.

[Cohen et. al. , 1989] Cohen, P.R., Dalrymple, M., Moran, D.B., Pereira, F.C.N., et al., "Synergistic Use of Direct Manipulation and Natural Language", Proc. of CHI-88, 1989.

[Cohen, 1991] Cohen, P.R., "The Role of Natural Language in a Multimodal Interface", 1991 International Symposium on Next Generation Human Interface, 1991.

[Grosz, 1977] Grosz, B. "The representation and use of focus in a system for understanding dialogs," *Proc. IJCAI 1977*, Boston, MA.

[Hayes, 1987] Hayes, P.J., "Steps towards Integrating natural Language and Graphical Interaction for Knowledge-based Systems", Advances in Artificial Intelligence - II, Elsevier Science Publishers, 1987.

[Hayes, 1988] Hayes, P.J., "Using A Knowledge Base To Drive An Expert System Interface With A Natural Language Component," in J. Hendler (ed.) *Expert Systems: The User Interface*, Ablex Publishing, 1988.

[Kobsa et. al., 1986] Kobsa, A., Allgayer, J., Reddig, C., Reithing, Nl, Schumauks, D., Harbusch, K., and Wahlster, W, "Combining Deictic Gestures and Natural Language for Referent Identification", Proc. of COLING-86, 1986.

[Pereira and Warren, 1980] Pereira, F., and Warren, D.H.D., p"Definite Clause Grammars for Language Analysis - A survey of the Formalism and a Comparison with Augmented Transition Networks", Artificial Intelligence, vol. 13, no. 3, 1980.

[Shimazu et. al., 1992] Shimazu, H., Arita, S., and Takashima, Y., "Design Tool Combining Keyword Analyzer and Case-Based Parser for Developing Natural Language DataBase Interfaces", Proc. of COLING-92, 1992.

[Shneiderman, 1991] *Designing The User Interface*, Addison Wesley Publ., Reading, MA.

[Sidner, 1979] Sidner, C. *Towards a computational theory of definite anaphora comprehension in English Discourse*, TR-537, MIT AI Lab, Cambridge, Ma.

[Wahlster, 1989] Wahlster, W., "User and discourse models for multimodal communication", in J.W. Sullivan and S.W. Tyler, editors, Intelligent User Interfaces, chapter3, ACM Press Frontiers Series, Addison Wesley Publishing, 1989.

[1] A survey of this work is beyond the scope of this paper, the interested reader is directed to the review in [Shneiderman, 1991].