The instructible agent Lokutor

Jan-Torsten Milde,
Fakultät für Linguistik und Literaturwissenschaft,
Computerlinguistik und Texttechnologie,
Universität Bielefeld,
Universitätsstr. 25, 33501 Bielefeld,
email:milde@coli.uni-bielefeld.de

Abstract

In this paper we describe Lokutor, a virtual human. Lokutor is a partially autonomous agent, inhabiting a 3D virtual environment. The agent can be controlled via natural language directives of the user and by direct manipulation using a graphical user interface. Its control system consists of a behavior-based reactive layer, a deliberative control layer and a motivational subsystem. Lokutor's domain knowledge, its behavior modules and its communicative behavior is encoded in an integrated multimodal format, which has been implemented using XML. Currently Lokutor is used to present information about a car.

1 Introduction

In natural environments humans rely on their ability to communicate using language. Natural language encodes information in a very compact way. Nevertheless in most cases processing language is seen as an isolated cognitive capability of an artificial system. Language understanding is restricted to the mapping of natural language expressions into an internal semantic representation, whereas language production takes an explicit semantic representation as input, from which a natural language utterance is generated. This point of view is inadequate while interacting with an situated agent living in an environment (virtual or natural).

The design of Lokutor reflects the definition of Franklin and Graesser, who define an autonomous agent to be: "*... a system situated within and as part of an environment that senses that environment and acts on it, over time, in pursuit of its own agenda and so as to effect what it senses in the future...*" (Franklin & Graesser, 1996).

Lokutor is an intelligent agent living in a simulated 3D environment. It has been designed to be as independent of the application domain as possible. To achieve this goal standard technologies have been used to implement the agent:

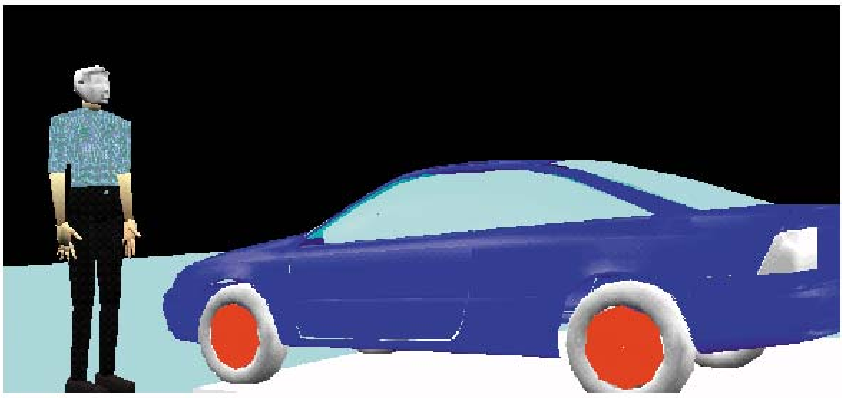

***Figure 1:** Lokutor and the Opel Calibra. The agent is able to walk through the environment, pointing and looking at details in the scene, opening and closing doors, trunk and bonnet of the Calibra. It uses the XML-encoded background knowledge to explain part of the functionality of the car.*

1. Java/Java3D for the implementation of the distributed simulation engine[1]
2. VRML 2 as the basis for the geometric object description[2]
3. H-Anim as a basis for the geometric joint description[3]
4. XML as a basis for the representation of the domain knowledge[4] (Holzner, 1998)

Lokutors current task is to present a car (an Opel Calibra) to a human user. The agent is able to convey information about the functionality of the car (e.g. how to open a door, how to open the tank lit, what type of fuel the car takes, the size of the space in the trunk etc). The Opel Calibra model has been converted from CAD data of a desgin model which has been kindly provided (just as its user manual) by Opel Germany. The user is able to interact with the agent by natural language directives.

Lokutor will follow these instructions, while integrating situative non-linguistic information in the analysis and interpretation of the directives (see Milde & Ahlers, 1999). Information given by the agent will be presented to the user in synthetic spoken natural language. Currently Lokutor is able to “speak” in English and German. The content of the information is retrieved from an XML annotated version of the cars user manual.

2 Multiagent simulation system

Our first approach to implement a multi agent system was based on a combination of VRML and Java technology (Jung & Milde, 1999). Unfortunately, the VRML technology integrated into the standard WWW browsers has shown to be very unstable and slow. It has been difficult to control the visualization process. Either control was accurate, but slow, or the agents would move smoothly but almost uncontrollably. Still the basic principle of separating visualization, world representation and agent control into three different processes seemed to be promising. We therefore switched from VRML to Java3D, which now is sufficiently matured to implement a stable and fast system.

As Java3D is able to import geometric objects described in VRML, the reimplementation of the visualization was relatively easy. Lokutors geometric appearance is based on a standard HANIM 1.0 figure. The skeleton structure of HANIM 1.0 allows definition flexible articulated agents. The arms, legs, the upper body and the head can all be moved seperately, making it possible to define realistic animations of the agents.

The world server encodes most of the state information of the simulated virtual world. Position and orientation of the agents are represented. The same holds true for a number of parts of the car. Virtual objects, like sensors, are also represented here. Based on this information the world server is able to calculate the current perceptual status of the agent. Lokutor is equipped with visual and tactile sensors. It is also possible to request the current sensomotoric status of the agent, mainly the joint angles of the limbs and wether the hand is open, closed or in a pointing position. Any number of control clients can connect to the server. This allows to control a number of interacting agents in the simulated environment. For the presentation task we only implemented one control client.

3 A hybrid control architecture

The control architecture (see figure 2) contains a deliberative system, which models „higher" cognitive competences, and a behavior-oriented[5] base system, which integrates language, perception and action on a lower level. This hybrid architecture (Milde, 1997) allows the distribution of the necessary competence and therefore the tasks of the whole system onto both subsystems. The deliberative system is responsible for the sequentialization of complex actions into simple basic actions and schedules the execution of these actions. However, because of the complexity and uncertainty of the real world, a fully detailed plan of the agents movements cannot be provided by the deliberative system. The behavior system on the other hand is embedded in the real world by means of sensors and actuators, which enable it to detect changes in the world and react to them immediately. It has the necessary competence for the autonomous execution of basic actions, but cannot aim at fulfilling given goals by itself. It is only through the coupling of the behavior system with a deliberative system, that the accomplishment of tasks can be realized by the interplay of goal-directed and reactive behavior.

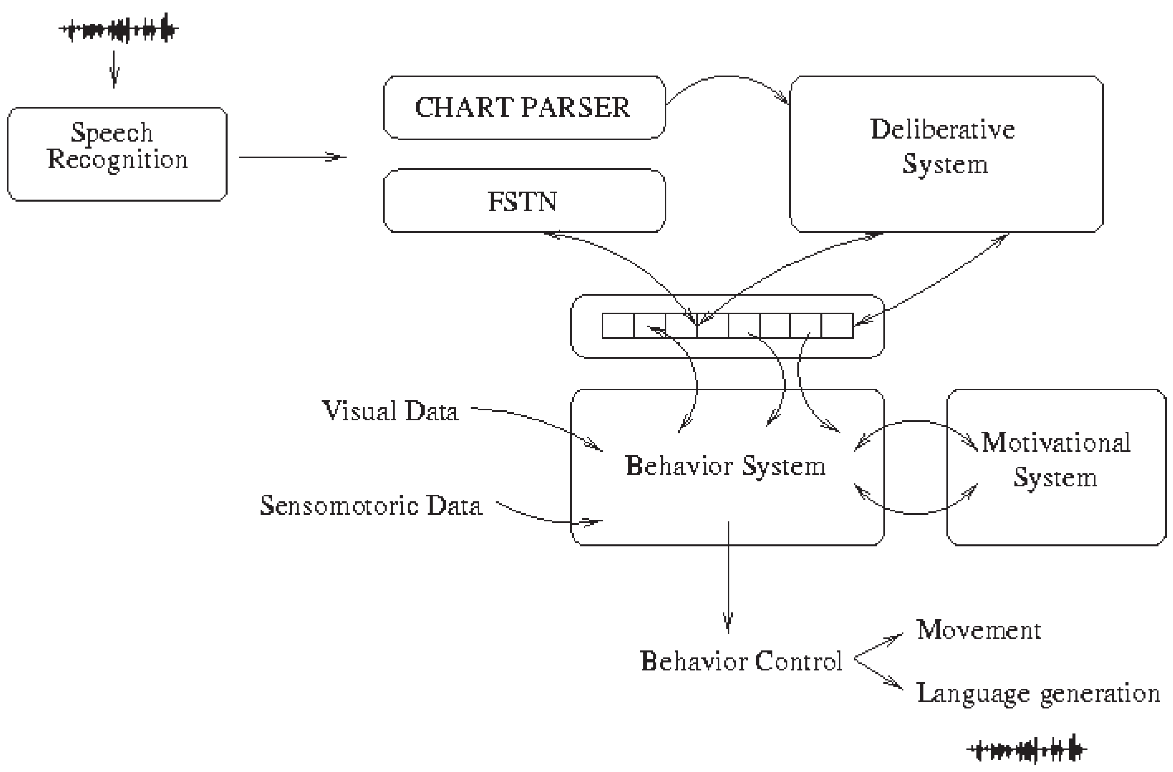

Figure 2: *The control architecture of Lokutor. The behavior system selects the next action of Lokutor based on sensoric date and linguistic information. The motivational system evaluates the internal state of the behavior system. The deliberative system controls longer term goals of the agent.*

Instructions cannot be processed by the base system directly. They provide resources for planning goals or action sequences and guide the planning process. First, instructions are parsed by the chart parser that builds up typed attribute-value pairs. The semantic part of those structures – based on the work of

Jackendoff (Jackendoff, 1990) – is passed on to the deliberative system, which is responsible for keeping track of long-time goals. The deliberative system uses this semantic part to initialize a corresponding action scheme (Lobin, 1999).

Action schemes contain explicit knowledge about the decomposition of higher level actions into basic actions. The resulting information blocks of an action scheme are mapped one after the other onto so-called internal sensors, which provide the basis for the communication between deliberative system and behavior system. Thus the necessary sequentialization of control parameters to initialize the corresponding basic actions in the behavior system can be produced. Suitable feedback from the base system allows the deliberative component to monitor the activity state of the behavior system and feed in the control parameters that are needed for the completion of the next subtask just in time.

The behavior system is partly autonomous. It can carry out basic actions and react to unexpected events without the help of an external control, but it is still controllable by a „higher“ system: The selection of basic actions is initialized by the deliberative system – as explained above – or through user interventions.

The behavior system consists of a hierarchy of behavior modules, each of it specialized for a certain task, which it can fulfil autonomously. In contrast to traditional, knowledge-based agent control, action related knowledge is not represented in a model of the world, but is distributed among all behavior modules. This leads to a reduction of complexity, because an error does not cause global replanning or the intervention of higher system components, but only local reactions inside the relevant behavior module. The modularization of the behavior system is motivated by the requirements concerning reactivity and autonomy of the behavior system on the one hand and by the expected user directives on the other hand: All possible linguistic sub-directives must be depictable to a – as small as possible – set of corresponding behavior modules.

Interventions refer to the direction and velocity of the agents movements and actions or to simple object-oriented actions. They are fed into the behavior system directly, thus allowing the immediate manipulation of ongoing behavior. The behavior system is responsible for the situated and time-adequate translation of sensor input into actuator output, treating information from the internal sensors just like any other sensor data. The integration of the different sensor inputs allows the situated interpretation of directives. As a consequence the processing of elliptical utterances like situation-dependent object references and indexical expressions, which can only be comprehended in the current context of sensing and acting, is made possible.

Language generation is based on textual information of the user manual. A text-to-phoneme system[6] transforms the textual information to a pre-speech level. The actual sound is then generated by the MBrola speech synthesizer[7] In our approach language instructions are treated as preprocessed sensoric input to the behavior system. Accordingly language generation will be issued by the behavior system as part of the action selection process. Suppose Lokutor is instructed as follows:

a) Öffne den Tankdeckel! `(Open the tank lid!)`

This instruction, once it has been processed by the language system, will be decomposed by the deliberative system into a moving towards the tank lid, a pointing at the object you are moving to and then to issue some explanation about the object you are pointing at action. Based on the current sensomotoric state, the explanation action filters the background knowlegde:

b) *Der Kraftstoffeinfüllstutzen mit Renkverschluß befindet sich an der rechten Wagenseite hinten. Tankdeckel aufschließen: Schlüssel einstecken und nach links drehen, den Deckel nach rechts ausrasten.* `(The tank lit is at right rear side of the car. Insert the key, turn left and open the tank lid to the right.)`

Most of the background knowledge is taken from the printed user manual, which is delivered to the customer with the car. It has been scanned and processed by OCR software to produce an electronic version.

An XML DTD has been developed which allows to annotate the text in a structured way. Also a number of attributes are defined in the DTD, which allows for the integratin of metaknowledge into this textual database:

```
<!ELEMENT agent (geom,scene,knowb)>
<!ELEMENT geom (#PCDATA)>
<!ELEMENT scene (#PCDATA)>
<!ELEMENT knowb (entry+)>
<!ELEMENT entry (name, desc)>
<!ELEMENT name (#PCDATA)>
<!ELEMENT desc (sit, action)>
<!ELEMENT sit (cond*)>
<!ELEMENT cond (att,val)>
```

```
<!ELEMENT att (#PCDATA)>
<!ELEMENT val (#PCDATA)>
<!ELEMENT action (move*, say*)>
<!ELEMENT move (basic*)>
<!ELEMENT basic (att,val)>
<!ELEMENT say (utt*)>
<!ELEMENT utt (#PCDATA)>
```

4 Outlook: Talking to each other

Currently the work on Lokutor has reached the level of a stable prototype implementation. It is possible to quickly set up the simulation system, define the basic animations of the agents including their sensoric abilities and connect the control system to the simulation. Lokutor runs a number of platforms without any difficulties, performing best under Win9X with a fast graphics adapter.

The hybrid control architecture has shown to produce believable behavior for the agent. Lokutor can be instructed, is able to follow longer term goals, while still being reactive. The agent is able to process natural language directives on different levels of complexity.

The next step is to populate the environment with a number of different Lokutors. Here Lokutor can be used as a means to do research on communicative agents. From a computational linguistic research standpoint this could lead to a better understanding of an extended speech act theory, which allows to model complex human communication. The experimental scenario will consists of two Lokutors standing face to face near a table on which a number of Baufix parts are lying. A set three experiments will be performed:

1) **Sorting game**: The parts are scattered over the table. Each Lokutor is selecting a part on the table and tries to grasp it and move it to its side of the table. Picking the parts has to be coordinated, such that each Lokutor is taking a piece at a time. If conflicts occur, these have to be resolved using natural language. The game ends, if all the parts are distributed amongst the agents.
2) **Maximizing game**: Each Lokutor starts sorting its pieces. First it selects a sensoric feature (color, size or type), then it will try to move every appropriate part to one side. Once all the pieces have been sorted, it will start asking the other Lokutor for more pieces. This is done as long, as either no more pieces will be exchanged, or the Lokutor is satisfied with the result, e.g. has reached

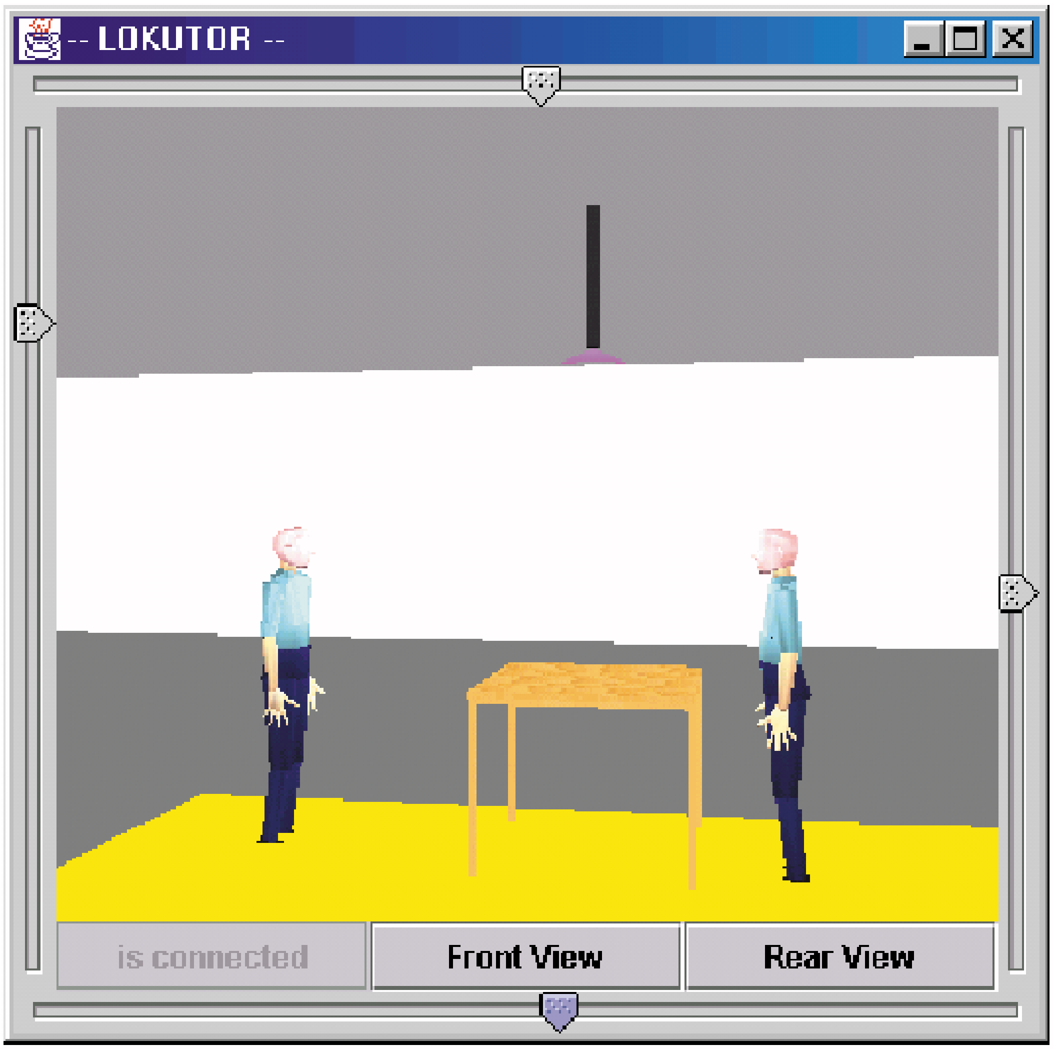

***Figure 3:** Two Lokutors interacting.*

a certain threshold level in its emotional system. Otherwise the experiment starts all over with a different sensoric feature.

3) **Imitation game**: In the center of the table a bridge consisting of two blocks and a bar is placed. The Lokutors are trying to build a parallel bridge. First, each Loktur tries to identify one of the supporting blocks, then tries to find a similar block on the table and place it next to the supporting block. These actions have to be coordinated as each Loutor is choosing the position indepently. Once the placement of the objects is correct, the bar has to be moved on top of the blocks. One Lokutor chooses a bar and picks it up. The second Lokutor is instructing the carrying Lokutor as where to move the bar. Once the end position of the bar is correct for one of the blocks, the bar is

handed over to the other Lokutor and the roles change: now the instructor become the carrier of the bar and vice versa.

In future work we will try to refine the generic declarative representation format of the underlying domain knowledge to allow more complex discourse structures when communicating with the agent. We think that the presented XML-based approach is a step into the right direction when building communicative agents for intelligent virtual environments.

References

R. A. Brooks, Intelligence Without Representation, in *Artificial Intelligence*, volume 47, pp. 139–159, 1991.

S. Holzner, *XML complete*, McGraw-Hill, New York, 1998.

R. Jackendoff, *Semantic Structures*, Current studies in linguistics series, 18, MIT Press, Cambridge, MA, 1990.

S. S. Franklin & A. Graesser, Is It an Agent, or Just a Program: A Taxonomy for Autonomous Agents. In J. P. Müller, M. J. Woolridge, and N. R. Jennings, editors, *Intelligent Agents III. Agent Theories, Architectures and Languages*, 21–35. Springer, Berlin, 1996.

B. Jung & J.-T. Milde. An open virtual environment for autonomous agents using VRML and Java. In *Procedings VRML'99*, 1999.

H. Lobin, *Handlungsanweisungen. Sprachliche Spezifikation teilautonomer Aktivität*, Deutscher Universitäts Verlag, 1999.

J.-T. Milde & Tobias Ahlers, Mensch-Maschine-Kommunikation in IVUs: Der kommunikative Agent Lokutor, in *Procedings of the first German workshop on Intelligent Virtual Environments*, KI 1999, Bonn, 1999.

J.-T. Milde, Kornelia Peters, and Simone Strippgen, Situated communication with robots, in *Proceedings of the first international Workshop on Human–Computer Conversation*, Bellagio, Italy, 1997.

K. Peters, Natürlichsprachliche Steuerung eines behaviorbasierten Roboters, Report 94/8, Situierte Künstliche Kommunikatoren, SFB 360, Universität Bielefeld, 1994.

L. Steels, The artificial life roots of artificial intelligence, in *Artificial Life Journal*, volume 1, MIT Press, Cambridge, 1994.

NOTES

1 http://java.sun.com/products/java-media/3D/index.html

2 http://www.vrml.org

3 http://ece.uwaterloo.ca:80/~h-anim/

4 http://www.w3.org/TR/REC-xml.html

5 A detailed description of the characteristics of behavior-oriented architectures can be found in Brooks, 1999 and Steels, 1994.

6 wxTTS, http://web.ukonline.co.uk/julian.smart/

7 MBrola, http://tcts.fpms.ac.be/synthesis