Instructing Cooperating Assembly Robots through Situated Dialogues in Natural Language[1]

A. Knoll, B. Hildebrandt, J. Zhang
Technische Fakultät, Universität Bielefeld
D-33501 Bielefeld, Germany

Abstract

We present an assembly cell consisting of two cooperating robots and a variety of sensors. It offers a number of complex skills necessary for constructing aggregates from elements of a toy construction set. A high degree of flexibility was achieved because the skills were realised only through sensory feedback, not by resorting to fixtures or specialised tools. The operation of the cell is completely controlled through natural language. Results from experiments in cognitive sciences and computer linguistics were incorporated to integrate natural language with vision as well as to control the construction dialogue between a human instructor and the robotic system. The experimental setup is described; a sample dialogue demonstrates the capabilites of the cell. A brief discussion of issues for further research concludes the paper.

1 Introduction

Endowing a single robot or a group of cooperating robots with the ability to carry a goal-directed conversation in natural language (*NL*) for performing non-trivial tasks is a demanding challenge not only from a robotics and a computer science perspective: it cannot be tackled without a deeper understanding of linguistics and human psychology [1]. Examples for tasks that would enormously profit from such an ability are service missions or the assembly of simple objects into more and more complex aggregates. NL input may be produced by a speech recognition system. More realistically – given the current state of technology – it is typed in by an instructor.

There are two conceptually different approaches to designing an architecture for incorporating NL input into a robotic system: the Front-End and the Communicator approach.

[1]This research is funded in part by the DFG under grant SFB360-C3 and SFB360-D4

1.1 The "Front-End" Approach

The robot system receives instructions in NL that completely specify a – possibly very complex – task the instructor wants to be performed. The input is analysed and in a subsequent separate step the necessary actions are taken. Upon completion of the task, i.e. after having carried out a script invoked by the instruction fully autonomously, the system is ready for accepting new input. This approach is ideal for systems that have to deal only with a limited set and scope of tasks, which do not vary much over time either. It much less lends itself to tasks that presuppose a high degree of flexibility during their processing.

Inadvertent changes of the environment resulting from the robot's actions, which would require a re-formulation of the problem, cannot be considered. Such situations cannot be dealt with unless the whole decisionmaking competence is transferred to the robotic system. For non-trivial tasks this is currently impossible; it is questionable whether it is at all desirable to try not to make use of the instructor's sensory system and intelligence (see the discussion of rationales for the introduction of sensor-based manipulation primitives in [5]). Neither is it possible to make specific references to objects (and/or their attributes) that are relevant only to certain transient system states because the instructor cannot foresee all of these states (c.f. the well-known AI "frame problem"). These references, however, are often indispensable for the system to work correctly, i.e. as intended by the instructor. With this approach the system cannot produce requests for specific and more detailed instructions because those, too, may arise only during the sequence of actions. Examples for this approach are [10, 6, 7]. To overcome the limitations of this approach, the concept of the "Artificial Communicator" was developed, which we briefly outline in the following subsection.

1.2 Communicator or Incremental Approach

If the nature of tasks cannot be fully predicted, it becomes inevitable to decompose them into (a host of) more elementary actions. Ideally, the actions specified are atomic in such a way that they always refer to only one step in the assembly of objects or aggregates, i.e. they refer to only one object that is to be assembled with another object or collection thereof (aggregates). The entirety of a system that transforms suitable instructions into such actions is called an *artificial communicator* (*AC*). It consists of cognitive NL processing, sensor subsystem and the robotic actors. From the instructor's point of view the AC should resemble a human communicator (*HC*) as closely as possible [9]. This implies several important properties of AC behaviour:

- All modules of the AC must contribute to an event-driven *incremental* behaviour: as soon as sufficient NL input information becomes available the AC must react. Response times must be on the order of human reaction delays.
- One of the most difficult problems is the disambiguation of instructor's references to objects. This may require the use of sensor measurements or NL input resulting from an AC request for more detailed information.
- In order to make the system's response seem "natural", some rules of speech act theory should be observed. The sequence of actions must follow a "principle of least astonishment", i.e. the AC should take the actions that the instructor would expect it to take. Furthermore, sensor measurements that are to be communicated about must be transformed into a human comprehensible form.
- It must be possible for the instructor to communicate with the AC about both scene or object properties (e.g. object position, orientation, type) and about the AC system itself. Examples of the latter are meta-conversations about the configuration of the robot arms or about actions taken by the AC.
- The instructor must have a view of the same objects in the scene as the AC's (optical) sensors.
- The AC must exhibit *robust* behaviour, i.e. all system states, even those triggered by contradictory or incomplete sensor readings as well as nonsensical NL input must lead to sensible actions being taken.

In other words: The AC must be seamlessly *integrated* into the assembly process. More importantly, it must be *situated*, which means that the situational context (i.e. the state of the AC and its environment) of a certain NL input is always considered for its interpretation. The process of interpretation, in turn, may depend on the history of utterances up to a certain point in the conversation. It may be helpful, for example, to clearly state the goal of the assembly before proceeding with a description of the atomic actions. There are, however, situations in which such a "stepwise refinement" is counterproductive, e.g. if the final goal cannot be easily described. Studies based on observations of children performing assembly tasks have proven to be useful in developing possible interpretation control flows.

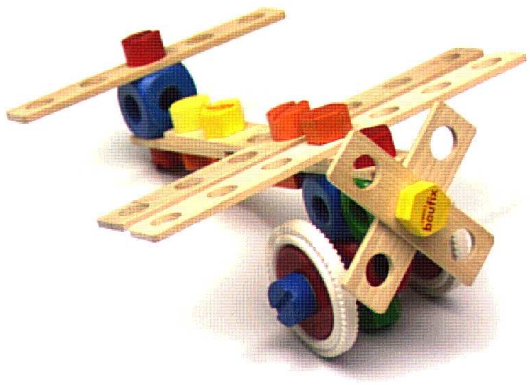

Figure 1: The fully assembled aircraft.

From an engineering perspective the two approaches can be likened to *open loop control* (Front-End Approach) and *closed loop control* (Incremental Approach) with the human instructor being part of the closed loop.

The research described in the following sections is embedded into a larger interdisciplinary research project aiming at the development of ACs for various purposes and involving scientists from the fields of computer linguistics, cognitive linguistics, computer science and electrical engineering.

2 Scenario and Object Naming

For studying situated goal-directed assembly dialogues, a prototypical scenario was chosen carefully. In this scenario a human instructor and an AC cooperate in building aggregates from elements of a toy construction set intended for children of the age of 4 years and up. The elements are made of wood (with little precision); their size is well suited to the parallel jaw grippers of our robots. The goal pursued in the sample conversations is the construction of the "aircraft" shown in fig. 1. Due to several mechanical constraints its complete construction is difficult for children. As observed during some of the experiments even some adults had problems assembling the aircraft although

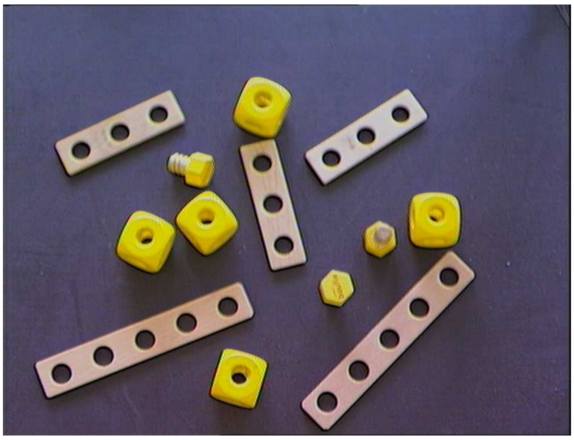

Figure 2: Randomly positioned construction elements: Cubes, Slats, Bolts.

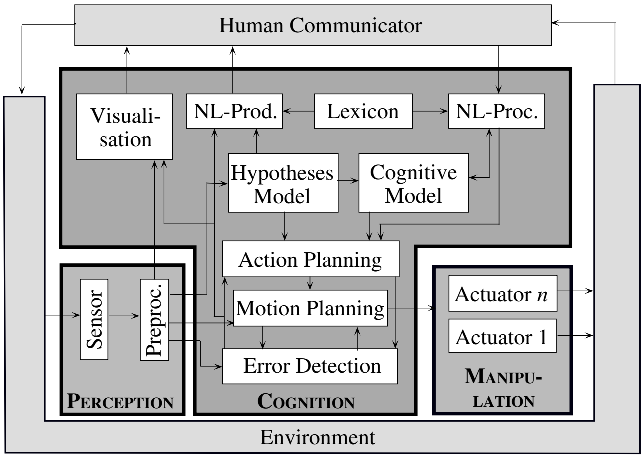

Figure 3: Architecture of the AC.

they were provided with the exploded view of the assembly. It remains to be shown that this can be done with robots using no specialised tools. In principle, however, it may one day become possible to replace the HC with an AC and to achieve the same goals through the same dialogue.

To illustrate only one individual problem occuring from a linguistic point of view we briefly turn to the question of *object naming* in this scenario.

In an assembly dialogue between HCs each object of the scenario may be referenced using a variety of different names. Before a sensible dialogue between HC and AC may take place, however, an unambiguous binding between an object and its reference name must be established. This binding must be identical on both the HC and AC side.

Since there is no common naming convention in natural language that is precise enough, a straightforward way of generating (initial) bindings is negotiation. Before entering the assembly, object names are assigned in an opening phase. The AC might, for example, point at one of the objects of fig. 2 (e.g. by highlighting it on a monitor) and ask the HC "What do we call / What do you want to call this object?" The HC's answer is then used as the name for the remainder of the assembly session.

While acceptable for testing purposes, such a procedure is obviously too inconvenient, time consuming and hence impractical in real-world applications involving dozens of objects. This is the reason, therefore, that the AC must possess the ability to react in a flexible manner to all (most) of the conceivable object names. It would be both difficult, cumbersome and intractable in the general case to compile all possible names for all possible objects in all possible situations. Fortunately, linguistic experiments have shown that rules may be postulated that HCs obey in assembly-type dialogues. These rules can be used to reduce the "name space" the AC must consider. Some of them follow:

- Even with simple items like the cube in fig. 2, HCs frequently switch between names. Apart from *cube* the object ist called *die, dice or block*.
- An object may be referenced not by its generic name but by its function in the situational context: the slat is named as such but also as *wing*, the cube may be called *nut* when used as the counterpart of the bolt.
- Particularly in this scenario objects are named after their geometrical shape where frequently a projection from three into two dimensions can be observed, e.g. the cube becomes a *square*.

The AC must recognise and cope with the principles and conditions under which these transformations occur [3].

3 Architecture of the AC

Even the construction of an aggregate of only a few elements may consist of a great number of elementary actions. Every assembly step resulting from an instruction comprises three distinct phases:

- The recording of the scene content using the sensory system;
- the processing of what is seen/sensed and the development of a plan for achieving a set (sub-)goal;
- the assembly of available elements with the actors.

Action	**Atomic**	**Complex**
Perceptive	Grab Image	Segment Image into Regions
Cognitive	Categorise Object in Scene	Make Plan for Object Motions
Manipulative	Close Gripper	Pick up Block

Table 1: Types of actions and examples for actions performed by the AC.

In other words: Every assembly step is composed of perceptive, cognitive and manipulative actions. Each of these may be atomic or complex (see Table 1 for some examples) and mirror (i.e. is the consequence of) specific instructions given by the HC.

While system architectures are conceivable that implement a temporally interleaved processing of perception, cognition and action, our system currently works strictly sequentially. At the beginning of each individual assembly step the scene is analysed visually. The objects are detected and their locations are computed. A geometrical model of the scene is generated. Once this model is available, the AC requests an instruction from the HC (the instructor). These instructions can be of the type

- Assembly (Construction): "Take the red screw";
- Scene Control: "The screw is/should be located on the left hand side of the bar";
- Meta-Level-Control: "Move the elbow up a little" or "Turn this camera a little clockwise";

where the latter type of meta-level instructions very rarely occurs in human construction dialogues.

The instructions are analysed linguistically and interpreted according to a hypotheses model of the scene and the history of the construction process, e.g. taking into account that a robot that has already grasped an object cannot grasp another one. As part of the cognitive phase a simple planner transforms complex into atomic actions.

Unlike standard motion sequence planners, this planner must also draw on knowledge obtained from cognitive linguistic observations. For example, an HC does not necessarily give all the instructions required for fulfilling the preconditions of a certain manipulative action. In some sense the problem is underdetermined; the planner must provide a solution within the given degrees of freedom. A simple example: The HC would not instruct the AC to grasp a screw (let alone a specific screw if more than one is available), before giving an instruction involving a screw. The

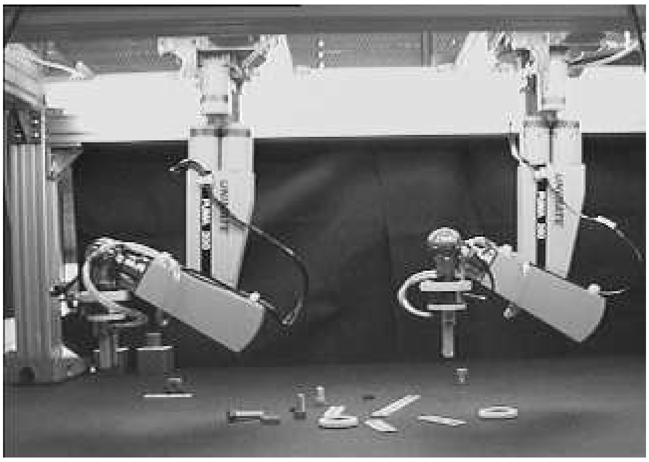

Figure 4: A view of the flexible assembly cell.

reasoning about what the HC may have meant and the necessary inferences are left to the AC's planner with no help other than the cognitive knowledge mentioned above. Currently, in such a situation, our system selects the object that the robot can grasp most easily (following a *principle of economy*). In the future this will be extended in such a way as to make an attention control possible, i.e those objects are chosen that are in the *focus of the discourse*.

Meta-level instructions/statements are necessary for interrupting the dialogue whenever the HC wants to guide the AC to a better sequence of actions than the latter is able to find autonomously. This is in contrast to most meta-level utterances in human dialogues, which normally deal with the (format of) the dialogue itself ("What are you doing there?", "Be more polite!").

Another important application of these instructions is error handling: imagine a situation in which the robot arm has run into a singularity while following move instructions by the HC. The typical HC, of course, has no comprehension of this problem. In such a case the AC must explain the (imminent) error, and a dialogue must be conducted about possible (consensual) ways leading out of the error situation. Sometimes errors pertaining to the actuators may be anticipated. If in such a case proper use is made of the NL-production facility (fig. 3), errors may even be *prevented*.

A further source of errors are utterances by the HC that the AC does not understand correctly. If the AC fails to comprehend the meaning of a statement, the HC must recognise the AC's problem and act accordingly. For this reason the linguistic components were so designed as to provide transparent messages whenever an error occurs. There are three classes of errors: lexical, syntactical and semantical. The reason for a lexical error is a certain (uncommon) word missing in the system's lexicon or a word having been misspelled. A syntactic error is reported when the parser cannot combine the individual words, i.e. it cannot compile a sensible syntactical structure. A semantic error occurs if the action required by the HC cannot be taken. This normally happens when the preconditions of the action are not met (and the necessary steps cannot be inferred); in particular if the necessary objects are not present in the scene.

After completion of the perception-cognition-manipulation sequence for a single assembly step, this cycle is repeated until the aggregate is finished.

The current architecture of the AC is summarised in fig. 3. The AC consists of the three components perception, cognition and action. The environment is the area/volume which both the AC and the HC can perceive and manipulate. The purpose of the visualisation module is the transformation of AC internal states into a human comprehensible form, i.e. they are adapted to the human cognitive system [4].

4 Experimental Setup

To complement the AC's cognitive component a manipulation unit or *cell* was built using standard robots that come as close as possible to the geometry of the human system of cooperating hand/arm systems. The similarity of the geometry often makes it easier for an HC giving instructions to the AC to image himself into the problems arising from the AC's point of view. The cell mimics the situation of an assembly carried out by an HC sitting at a table (and possibly being given instructions). In such a setting the construction elements are placed on the table, the HC's arms/hands cooperate from above the table.

Since humans can easily put together the elements of the construction set without using tools, it is required that the cooperating manipulators be flexible enough to emulate all necessary skills. Even though the grippers are much less capable than the human hand, no additional mechanical devices such as fixtures or specialised tools are used.

The set-up resulting from these (and further mechanical) requirements is shown in fig. 4. The following components are utilised:

- Two manipulators (Unimation Puma-260) are installed overhead on a stationary massive frame of aluminium profiles. They possess overlapping work spaces.

- A pneumatic parallel jaw gripper with integrated force/torque sensor is mounted on each manipulator wrist.

Figure 5: Robot 1 holding screw in hole, robot 2 holding cube and approaching.

- Miniature colour cameras directly above the force/torque sensors provide a view of the objects to be handled/grasped. They are used for the visual servoing tasks.

- An overhead 3-CCD colour camera provides images of the scene that are used for determining the obejct locations.

The system is controlled by several Sparc-stations working in parallel. The robots are controlled by Multi-RCCL[2].

Up to now the following assembly skills have been implemented on both manipulators:

Pick-up: Most elements of the construction set can be picked up from any location on the table. The approach of the end effector's tip to the desired grasping point is controlled in real time using "self-viewing visual servoing"[8].

Put-down: Elements or aggregates can be put on the table or on other objects. Prior to releasing the gripper controlled forces and torques may be applied to the object.

Peg-in-hole: Most combinations of objects that can be passed through one another can be handled. If necessary, a reflex can be activated that lets one of the robots find the center of the hole by following a spiral path under force control.

Screwing: This is by far the most complex operation available. It requires sensitive force/motion control. It involves the

- approach phase in which the true thread position is determined;

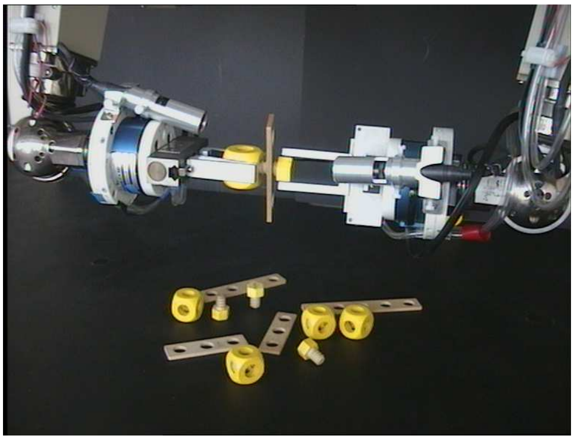

Figure 6: Screwing by cooperating robots.

- detection of the contact angle between screw and start of the thread;
- re-grasping of the bolt head after completing one revolution;
- application of the tightening torque.

The latter is particularly difficult because the wooden screws tend to block. Special types of adaptive fuzzy controllers for force control have proven to be superior in performance to standard PID controllers [11].

Taken together, these skills make the construction of some aggregates of the aircraft model possible, e.g. the "elevator control" of the aircraft shown in fig. 1, which involves a slat, a bolt and a cube with a thread. In the following section we shall reconstruct the dialogue leading to the complete aggregate.

5 A Sample Dialogue and Results

The following is a short sample dialogue which was carried out in order to build the "elevator control" aggregate out of three elementary objects. The objects were laid out on the table in a way similar to fig. 2 (i.e. there were many more objects on the table than necessary). The instructor had a complete image in his mind of what the assembly sequence should be. Alternatively, he could have used the assembly drawings in the construction kit's instructions and translated them into NL.

1 *What do you want me to do?*
2 **Pick up a bolt.**
3 *What do you want me to do?*
4 **Grasp the three hole slat.**
5 *What do you want me to do?*
6 **Put the screw on the slat.**
7 *I cannot do that. The objects are in locations not suitable for this kind of action.*
8 **Put the screw through the center hole of the slat.**
9 *What do you want me to do?*
10 **Take the blue cube.**
11 *There is no blue cube available.*
12 **Take a cube. Take the yellow cube.**
13 *What do you want me to do?*
14 **Screw the bolt into the cube.**
15 *What do you want me to do?*
16 **Put the aggregate on the table**
17 *What do you want me to do?*
18 **The work is finished.**

Lines input by the HC are typeset in bold face, the other text is AC output. The first AC input request in *Line 1* is output after the determination of all object locations, types and colours. The necessary classification and subsequent steps are based on the colour image obtained from the overhead colour camera.

After linguistic processing involving the instances in fig. 3 the HC input in *Line 2* first triggers the action planner, which decides which bolt to grasp and which robot to use. Since the HC did not specify either of these parameters, both are selected according to the principle of economy. In this case, they are so chosen as to minimise robot motion. The motion planner then computes a trajectory, which is passed to the RCCL subsystem. Since there are enough bolts available, the AC issues its standard request for input once the bolt is picked up.

HC input *Line 4* results in the other robot picking up the slat. Given the locations of the slat and the bolt, *Line 6* does not make sense and is thus rejected (*Line 7*). The correct instruction in *Line 8*, however, is followed (note that the *bolt* is now called *screw*). This instruction requires the peg-in-hole module mentioned above. The left part of Fig. 5 shows the position of robot 1 and the two objects after the completion of this operation.

HC input *Lines 10, 11 and 12* demonstrate that error messages are adapted to the previous input and that redundancies in an input line are accepted without producing an error. The latter is less important in our current "keyboard setting" because most HC will correct their erroneous, incomplete or ambiguous input before entering the instruction. It will, however, become very important when speech processing is added to the system.

The right part of fig. 5 shows how robot 2 (holding the cube) approaches the bolt held by robot 1 as the first step in the screwing operation triggered by the instruction of *Line 14*. The screwing is shown in fig. 6. Many uncertain parameters have to be taken into account; in particular the bolt axis is never in line with the effector's z-axis. Using the adaptive force control mentioned above, however, angles between the two axes of up to 15 degrees can be accomodated without blocking (if the thread of the bolt is not excessively worn out).

Line 16 instructs the robot to put down the finished aggregate and release the gripper after applying a certain force in the direction perpendicular to the table surface.

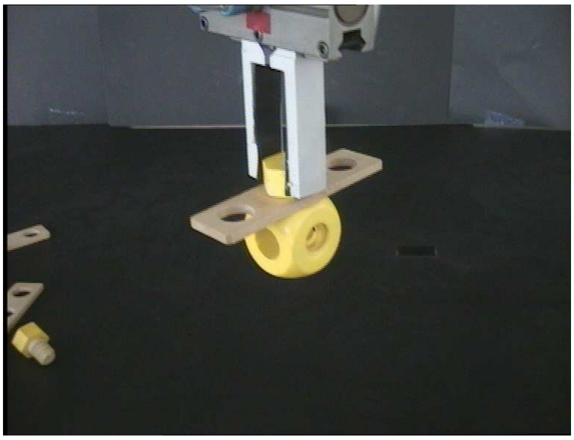

Figure 7: Finished aggregate: "Aircraft Elevator Control."

6 Conclusions

A scenario was defined that consists of only a limited set of construction elements but offers a rich variety of different tasks. It may serve equally well as the basis for construction experiments in cognitive linguistics (between HCs) and for benchmarking the perceptive, cognitive and manipulative skills of a real-world robotic system. Our present core assembly cell has shown that in a limited domain like assembly (or disassembly) robotic systems may be realised that – by making appropriate use of human intelligence and negotiating some key decisions with the user – may be of great help even to unexperienced users. Further research will concentrate on making the system more flexible on the cognitive side, i.e. by extending and improving its lexicons and grammar, and on the manipulation side by adding further skills while retaining its mechanical simplicity (no fixtures). Furthermore, error handling and real-time error detection will be important issues along with research into the incrementality and the robustness of the whole system.

References

[1] C. Grangle and P. Suppes. *Language and Learning for Robots*. CSLI Publications, Stanford, Ca., 1994.

[2] V. Hayward and R. Paul. Robot manipulator control under Unix RCCL: A robot control "C" library. *Int. Journal of Robotics Research*, 5(4), 1986.

[3] W. Heydrich and H. Rieser. Public information and mutual error. Technical report, SFB 360, 95/11, Universität Bielefeld, 1995.

[4] B. Hildebrandt, R. Moratz, G. Rickheit, and G. Sagerer. Integration von Bild- und Sprachverstehen in einer kognitiven Architektur. *Kognitionswissenschaft*, 4:118...128, 1995.

[5] G. Hirzinger, B. Brunner, J. Dietrich, and J. Heindl. Rotex - the first remotely controlled robot in space. In *Proc. IEEE Conference on Robotics and Automation*. IEEE Comp. Soc. Press, 1994.

[6] K. Kawamura and M. Iskarous. Trends in service robots for the disabled and the elderly. In *Proc. IROS '94 – IEEE/RSJ/GI Int. Conf. on Intell. Robots and Systems*. IEEE Press, 1994.

[7] T. Laengle, T. Lueth, E. Stopp, G. Herzog, and G. Kamstrup. KANTRA - a natural language interface for intelligent robots. Technical report, SFB 314 (VITRA) - Bericht Nr. 114, Universität des Saarlandes, 1995.

[8] P. Meinicke and J. Zhang. Calibration of a "self-viewing" eye-on-hand configuration. In *Proc. IMACS Multiconf. on Comp. Eng. in Syst. Appl.*, Lille, France, 1996.

[9] R. Moratz, H. Eikmeyer, B. Hildebrandt, A. Knoll, F. Kummert, G. Rickheit, and G. Sagerer. Selective visual perception driven by cues from speech processing. In *Proc. EPIA 95, Workshop on Appl. of AI to Rob. and Vision Syst.*, TransTech Publications, 1995.

[10] P. Restaino and R. Meinicoff. The listeners: Intelligent machines with voice technology. *Robotics Age*, 1985.

[11] J. Zhang, Y. Collani, and A. Knoll. On-line learning of B-spline fuzzy controller to acquire sensor-based assembly skills. In *Proc. IEEE Int. Conf. on Robotics and Automation*, Albuquerque, 1997.