

Progress in Programming the HRP-2 Humanoid Using Spoken Language

Peter Ford Dominey, Anthony Mallet, Eiichi Yoshida

Abstract— In order for humans and robots to cooperate in an effective manner, it must be possible for human users to efficiently communicate to the robot its role in accomplishing the coordinated task. Spoken language is an obvious candidate. The current research analyses how spoken language can be used by human users to communicate with the HRP-2 humanoid to program the robot's behavior in a cooperative task. The task involves the humans and the HRP-2 working together to assemble a piece of furniture. The objectives of the system are to 1. Allow the human to impart knowledge of how to accomplish a cooperative task to the robot, i.e. to program the robot, in the form of a sensory-motor action plan. 2. To do this in a semi-natural and real-time manner using spoken language. In this framework, a system for Spoken Language Programming (SLP) is presented, and experimental results are presented from this prototype system. In Experiment 1, the human programs the robot to assist in assembling a small table. In Experiment 2, the generalization of the system is demonstrated as the user programs the robot to assist in taking the table apart. The SLP is evaluated in terms of the changes in efficiency as revealed by task completion time and number of command operations required to accomplish the tasks. Lessons learned are discussed, along with plans for improving the system, including developing a richer base of robot action and perception predicates that will allow the use of richer language.

I. INTRODUCTION

The development of robotics platforms has reached a technical level such that there are now available highly articulated multiple degree of freedom humanoids that are physically capable of locomotion, object manipulation, and an essentially unlimited set of sensory motor behaviors. This sets the scene for the corresponding technical challenge: How can non-specialist human users program these robots for human robot cooperation? Crangle and Suppes [1] stated: "(1) the user should not have to become a programmer, or rely on a programmer, to alter the robot's behavior, and (2) the user should not have to learn specialized technical vocabularies to request action from a robot." In this context it is crucial that there exists a straightforward way for humans and robots to communicate with one another. This

Manuscript received September 30, 2006. Supported in part by French Minister of Research under grant ACL-TTT, and the CNRS Joint Robotics Laboratory, LAAS Toulouse. P. F. Dominey is with the CNRS, 67 Bd Pinel 69675 Bron Cedex, France (phone: 33-437-911266; fax: 33-437-9112110; e-mail: dominey@isc.cnrs.fr).

A. Mallet is with the LAAS, CNRS, Toulouse France (e-mail: anthony.mallet@laas.fr).

E. Yoshida is Co-Director of the CNRS/AIST French-Japanese Joint Robotics Laboratory, with the LAAS, CNRS, Toulouse France (e-mail: eiichi.yoshida@laas.fr).

communication should in particular allow humans to specify to the robot what it is supposed to do, and when and how it should do it. For communication between human team members, spoken language provides a very rich and direct vector [1]. Language essentially provides a vector for the transmission of meaning between agents, and should thus be well adapted for allowing humans to transmit meaning to robots.

Construction grammar (CxG) provides a linguistic formalism for achieving the required link from language to meaning [3]. Indeed, grammatical constructions define the direct mapping from sentences to meaning. Meaning is represented in a predicate-argument (PA) structure as in (2), based on generalized abstract structures as in (3). The power of these constructions is that they are based on abstract "variables" that can take an open set of arguments .

- (1) John put the ball on the table.
- (2) Transport(John, Ball, Table)
- (3) Event(Agent, Object, Recipient)

We previously developed a system that generates PA representations (i.e. meanings) from video event sequences, and demonstrated that when humans performed events and described what they were doing, the resulting <sentence, meaning> input pairs allowed our learning system to acquire a set of grammatical constructions [4]. The resulting system could describe new events and answer questions with a rich set of learned grammatical constructions.

PA representations can be applied both to commanding actions as well as describing them. Hence the CxG framework for mapping between sentences and their PA meaning can be applied to both as well. In either case, the richness of the language employed will be constrained by the richness of the perceptual and action PA representations of the target robot system. In the current study we start from a restricted initial condition in which there are no high level predicate-argument representations. Instead, we have access only to the command of joint angles. Part of the challenge is to provide an intermediate layer of commands that are well adapted to a class of manipulation tasks.

Thus, similar to the language based task analysis in [5] an essential part of the analysis we perform concerns examining a given task scenario and determining the set of action/command primitives that satisfy two requirements: 1. They should allow a logical decomposition of the task into units that are neither too small (i.e. move a single joint) nor too large (perform the whole task). 2. They should be of

general utility so that other tasks can be performed with the same set of primitives.

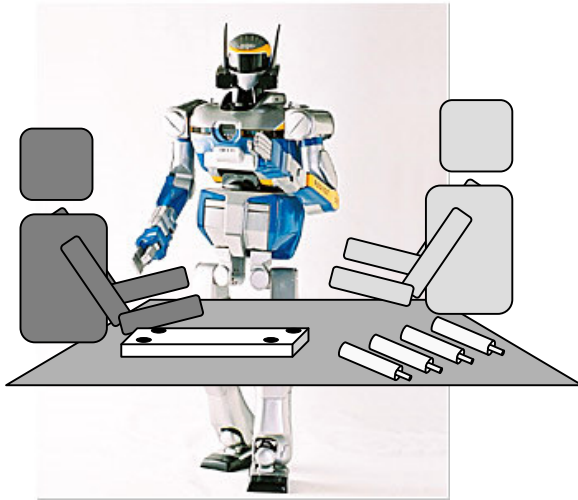


Fig 1. Stylized view of the cooperative interaction scenario. The two human users and the HRP-2 will cooperate in constructing the table.

II. A SCENARIO FOR HUMAN-ROBOT COOPERATION

A. The Scenario

Figure 1 illustrates the HRI scenario that we analyze in this research which involves two humans and the HRP-2 cooperating in the construction of a small table. The construction task will involve fixing the legs to the surface of the table with wood screws. User1 on the left interacts with the robot and with User2 on the right via spoken language.

User1 will command the robot to prepare to receive one of the table legs that will be passed to it by User2. The robot waits until it receives a “continue” signal from User1, and will then pass the leg to User1 who will take the leg, and then ask the robot to hold the table top steady allowing User1 to attach the leg to the table. User1 then tells the robot to release the table. At this point, the first leg has been fixed to the table, and the “get the leg and attach it” sequence can be repeated.

B. On-line commanding

The simplest solution for controlling the robot in this task, which involves no teaching (or “programming”) by the user is for User1 simply to tell the robot and User2 what to do, step by step. In a novel task that the user is not fully familiar with, this on-line commanding is perhaps the most useful method, because it allows the user to become familiar with the step-wise nature of the execution of the task.

C. On-line commanding with repetitive a subsequence

On-line commanding allows the user to be responsive to new situations, and to learn him/herself by taking the robot through a given task or tasks. On the other hand, for tasks that are well defined, the user should be able to program the

robot by saying the sequence of commands and storing it before the actual execution. In between these two conditions there may arise situations in which during the course of solving a cooperative problem with the robot, the user comes to see that despite the “open endedness” of a given problem set, there may repetitive subtasks that occur in a larger context in which some uncertainty can exist. In this type of situation, the human user may want to teach the robot about the repetitive part so this can be executed as an autonomous “macro” while the user still remains in the execution loop for the components that require his/her decision.

The table assembly task corresponds to this situation. For each of the four legs the robot should receive the leg from User2, pass it to User1 and then hold the table surface in place while User1 fixes the leg to the table, before repeating the same procedure for the next leg. After 1 or two repetitions of this exercise, for the first leg or two, User1 should have a good idea of how this repeating subsequence goes, and can thus teach it to the robot so that the entire behavior can be accessed by a single command.

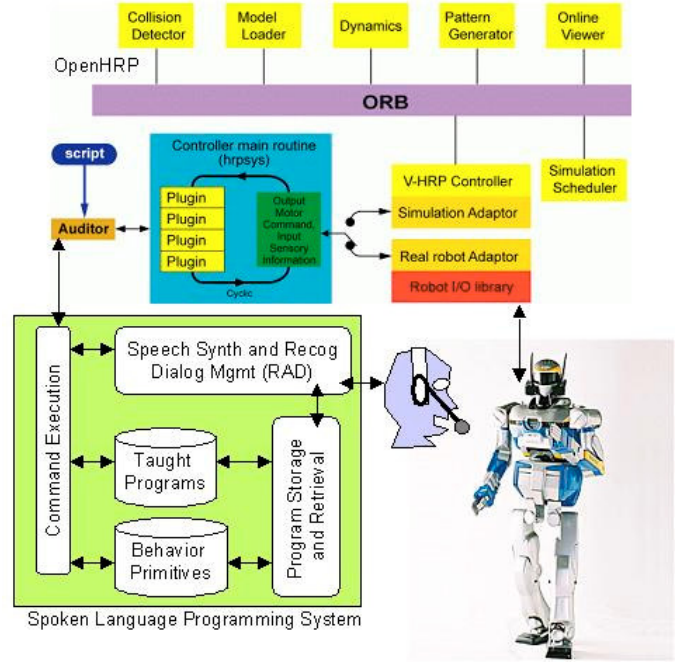


Fig. 2. Implementation architecture

III. IMPLEMENTATION

Based on the requirements derived from this scenario, we can now begin to allocate these requirements to different components of the system. The current studies are performed with the Kawada Industries HRP-2 humanoid robot [6] under the control of the OpenHRP controller [7]. The HRP-2 has 30 controlled degrees of freedom, 8 of which are used in this study. The spoken language interface technology is provided by the CSLU RAD system. This runs on a PC Pentium III Windows machine, which communicates with the OpenHRP controller via wireless

internet with an ssh connection. The system is quite modular however, and the robot controller for the OpenHRP can be replaced by the controller for other robots. We have used the AIBO ERS7 with a WIFI interface replacing the serial port interface, the Lynxmotion 6DOF robot arm, and Khepera mobile robots both with a serial port controller (see discussion). Part of the novelty here is the use of the HRP-2 with many more effective degrees of freedom and possibilities for rich cooperative interaction.

A. Dialog Management

Dialog management and spoken language processing (voice recognition, and synthesis) is provided by an “off the shelf” public domain product, the CSLU Rapid Application Development (RAD) Toolkit (<http://cslu.cse.ogi.edu/toolkit/>). RAD provides a state-based dialog system capability, in which the passage from one state to another occurs as a function of recognition of spoken words or phrases; or evaluation of Boolean expressions.

In the mixed initiative dialog system we developed, the system prompts the user with “I am ready” and waits for the user to respond with one of the commands (Table 1) and these are immediately executed. The user can also issue commands for programming the robot (Table 2). These commands include “learn” and “ok” which indicate the beginning and end of a macro sequence to be stored. Thus, in a single session, a user might first operate in direct mode to become familiar with how to solve a given problem, then pass into macro learning mode, generate a new program and run it in order to simplify subsequent task execution.

B. HRP2 Embedded control software

The set of primitive actions the robot can execute requires the integration of a large variety of heterogeneous functionalities. For instance, simply moving an arm in a real environment requires perception capabilities, path planning algorithms or collision avoidance routines in order to make sure the robot moves safely. Actions can furthermore be parameterized by the human (e.g. asking to the robot to go to a given room, or changing the speed of a motion by adding the word “fast” or “slow” to the command, etc.) and can thus be integrated in a more generic way.

In order to endow the robot with high level actions, complex functions such as sensor and actuator control, servo-control, monitoring, image processing, environment modeling, localization and trajectory generation have to be integrated and cooperate, exchange data or even synchronize their execution. We also have to offer mechanisms that allow them to be started, interrupted and restarted upon request (control flow), and define the way they are fed with adequate input data and the way they export processed data (data flow).

To achieve this, we propose a modular architecture based on a network of software components called *functional modules* defined as follows [8]: a functional module is an

active software entity that encapsulates one or several algorithms and is in charge of a specific functionality with respect to the missions of the robot. The functional module is a request server that manages all the communications with other modules, runs the functional algorithms when required and paces them by using its own execution threads. Modules are not connected together: they are controlled by the upper layer called Supervision, which is in charge of controlling the proper execution of high-level tasks by using the set of services they provide.

Within the supervision layer, and provided that sufficient functionalities are available, we would then be able to develop generic and complex primitive actions that could be remotely triggered on demand by the operator. These actions could range from navigation tasks such as going to a particular room, possibility parameterized by the speed, to generic manipulation tasks or vision-based routines like object recognition. The integration of this software is under development.

C. HRP-2 Specific Commands

The final behavioral result of a spoken action command that is issued either directly or as part of a learned plan is the execution of the corresponding action on the robot. Based on the preliminary analysis of the table-building scenario described above, a set of primitive actions was identified for the HRP2. Each of these functions corresponds to a particular posture that is specified as the angles for a subset of the 30 DOFs. These actions have been implemented by hand-written python scripts that specify final, hardcoded, joint angles and motion durations for the given postures. The python script execution is triggered remotely by the CSLU toolkit, and communicates directly with the low-level OpenHRP framework (Fig. 2). The motion is achieved by linearly interpolating joint angles between the starting and final configurations, for each specific action. We have chosen these simple actions in order to demonstrate the feasibility of the overall approach in the table-building scenario, and more complex functions are currently under development.

Table 1. Action Commands

Motor Command	Resulting Actions
Prepare	Move both arms to neutral position, rotate chest to center, elevate left arm, avoiding contact with the work surface (5 DOF)
OpenLeft	Open left hand (1 DOF)
CloseLeft	Close left hand (1 DOF)
Give it to me	Rotate hip to pass the object in left hand to user on the right (1 DOF)
Hold	Center hip, raise right arm preparing to hold table top (5 DOF)
Right open	Open right hand (1 DOF)
Right close	Close right hand (1 DOF)

D. General learning and control commands

In addition to the HRP-2 specific motion commands, the system requires a set of commands that allow the user to control the actual programming and program execution. These commands and their consequences are presented in Table 2. When the user invokes the “Learn” command, the dialog system begins to encode the sequence of the subsequent commands that are issued. The user proceeds to issue action commands to effect the desired task that will make up this sequence. When the user has finished the part of the task he wants to program, he issues the “OK” command. This results in the action sequence being written to a file. Now, when the “Macro” command is issued, this file is read into an array, and the commands are sequentially executed. During these executions, the behavioral scenarios above also identified the requirement for a conditional wait, in which the execution of a stored sequence waits for the user to finish what he is doing which the user signifies with the “continue” command. Thus, when the “wait” condition is issued, the system pauses until the “continue” command is issued.

Table 2. Learning and Control Commands

Commands	Correspondence
Learn	Begin encoding subsequent commands
OK	Store encoded command sequence in macro
Macro	Execute the stored macro
Wait	Interrupt command execution until a spoken “continue” command is issued
Continue	Terminate the “wait” pause and resume execution.

IV. EXPERIMENTAL RESULTS

In order to evaluate the implemented system, we performed two experiments that involved human-robot cooperation via programming. In the first experiment, two human users and the robot cooperate to construct a small table. In the second experiment, they cooperate again, this time to take the table apart. In our initial development, only the first experiment was planned, but when it was successfully accomplished we wanted to determine if the set of action primitives we had developed were indeed general enough to extend to a new but related task of disassembling the table.

Each of the two experiments will be presented, followed by an analysis of the effects of the ability to program on measures of efficiency of task completion.

A. Experiment 1: Assembling the table

In the following paragraphs we provide the literal transcription of the human-robot dialog, annotated with explanation about the corresponding human and robot behavior and key system functions in $\langle \rangle$ brackets. Robot

language will be indicated in *italics*, and human language in plain text.

1) Phase 1. Spoken language commanding.

The users begin their first interaction with the robot. User1 controls the task via spoken language. Because the task is new to him, he proceeds for the moment only by commanding the robot, with no programming.

Dialog: \langle Robot initiates the interaction \rangle *I am ready.* Prepare. *You said prepare?* Yes. \langle Because of possible errors in speech recognition, we choose to have an explicit validation on each command as illustrated here \rangle . *Preparing to grasp.* \langle Robot raises left hand \rangle *I am ready.* Left open. *You said left open?* Yes. *Opening left hand.* \langle Robot opens left hand \rangle *I am ready.* \langle User2 places the table leg in the robot’s left hand as in Fig3B \rangle Left close. *You said left close.* Yes. *Closing left hand.* \langle Robot closes left hand to grasp the table leg \rangle *I am ready.* Give it to me. *You said give it to me?* Yes. *Giving you the object.* \langle Robot turns to the right, holding the table leg out to User1 \rangle *I am ready.* Left open. *You said left open?* Yes. *Opening left hand.* \langle As the Robot opens the hand, User1 reaches for and grasps the table leg (as in Fig. 3C) \rangle *I am ready.* Hold. \langle the first of several primitives required for the robot to hold the table while User1 attaches the leg. \rangle *You said give it to me?* No. \langle This is an example of a speech recognition error. \rangle *I am ready.* Hold. *You said hold?* Yes. *Getting ready to hold.* \langle The robot orients to the work surface and raises its right hand. *I am ready.* Right open. *You said right open?* Yes. *Opening right hand.* \langle With the robot’s right hand open, User1 places the table top within the robot’s grasp (Fig. 3D). *I am ready.* Right close. *You said right close?* Yes. *Closing right hand.* *I am ready.* Wait. *You said wait?* Yes. *Waiting for your signal* \langle During this wait period, User1 attaches the leg to the table, and then gives the “continue” command when ready. \rangle Continue. *I am ready.* Right open. *You said right open?* Yes. *Opening right hand.* \langle User1 takes the table from the robot and prepares for the next leg. \rangle *I am ready.*

2) Phase 2: Programming the Robot

At this point, User1 has used spoken language tele-operation to enlist the robot’s assistance in attaching the first leg to the table, as indicated in Fig. 3A. Given the knowledge of the task User1 has gained, he is now ready to “program” the robot for the repetitive component of the task by using the “learn” command.



Fig 3A. HRP-2 is in the ready position. The first leg has been attached. User1 is now familiar with the task, and issues the “learn” command.



Fig. 3D. After the “hold” command, HRP-2 has right hand raised and open, ready to hold the table surface in place. User1 prepares to position the table top in the robot’s open right hand;.



Fig. 3B. After the “prepare” command, HRP-2 has left hand in grasping position. User2 positions the leg in the gripper, and User1 issues the “close left” command. HRP-2 closes left hand to grasp the leg.



Fig. 3E After the “right close” command, HRP-2 has the table held securely in its right hand, allowing User1 to use both hands to attach the leg to the table.



Fig. 3C. After the “give it to me” command, HRP-2 rotates waist towards user1 with left hand in giving position. User1 prepares to take the object after issuing “left open” command..



Fig. 3F. After the “open right” command, HRP-2 releases the table and User1 takes the table. At this point, the OK command is given, and the sequence is stored as a macro. User1 then invokes this for the third and fourth legs with the command “macro.”

Dialog: Learn. *You said learn?* Yes. *I'll start learning now. I am ready.* Prepare. *You said prepare?* Yes. *Preparing to grasp.* <by assuming neutral position and then raising left arm> *I am ready.* Wait. *You said wait?* Yes. <this pause allows User2 to place the leg in the robots left hand> *Waiting for your signal.* Continue. *I am ready.* Left close. *You said left close.* Yes. *Closing left hand.* <To grasp the leg being passed by User2 (Fig 3B)> *I am ready.* Give it to me. *You said give it to me?* Yes. *Giving you the object.* <By rotating towards User1>. *I am ready.* Left open. *You said left open?* Yes. *Opening left hand.* <User1 takes the leg (Fig 3C)>. *I am ready.* Hold. *You said hold?* Yes. *Getting ready to hold.* *I am ready.* Wait. *You said wait?* Yes. *Waiting for your signal.* <While User1 puts the table into the right hand Fig (3D)>. Continue. *I am ready.* Right close. *You said right close?* Yes. *Closing right hand.* *I am ready.* Wait. *You said wait?* Yes. *Waiting for your signal.* <While User1 attaches the leg (Fig 3E)>. Continue. *I am ready.* Right open. *You said right open?* Yes. *Opening right hand.* <User1 takes the table>. *I am ready.* OK. *You said OK?* Yes. *OK we will store this plan.* *I am ready.*

3) First Execution of Stored Program Macro

Macro. *You said macro?* Yes. *Running the macro.* *Preparing to grasp.* *Waiting for your signal.* Continue. *Closing left hand.* <Robot takes leg from U2> *Giving you the object.* *Opening left hand.* <U1 takes leg>. *Getting ready to hold.* *Waiting for your signal.* <U1 Continue. *Closing right hand.* *Waiting for your signal* (screwing table leg). Continue. *Opening right hand.* *In line macro finished.* *I am ready.* <Third leg is now attached. User1 takes the table. Same procedure with macro for attaching fourth leg.>

During the assembly of the first two legs, User1 issued 10 distinct commands for each of the two legs. Installing legs 1 and 2 took 3min 25sec and 3min 24sec. Once the program was learned, for legs 3 and 4, a single command initiated the program, and the user was only required to issue 3 “continue” commands in order to indicate to the robot that he was ready to proceed. Execution time was reduced to 2:11 and 2:33 respectively. The ability to execute a suite of primitive behaviors with a single command had a significant impact on the execution time for completion of the task. Fig 4 indicates the average execution times for the individual commands under the explicit command conditions without the programmed macro (CMD) at 25.14 sec/operation, and under macro program execution conditions (PRG) at 15.00 sec/operation. We performed a statistical analysis of the variance (ANOVA) in these completion times examining the effects of Repetition (i.e. first and second trial in either the CMD or PRG mode), and Programming condition (i.e. CMD vs PRG). Only the Programming condition had a significant effect on the completion times (ANOVA, Programming Effect: $F(1,6) = 109, p < 0.0001$).

These results demonstrate that (1) spoken language can be used as a form of tele-operation, (2) that this can provide the

basis for allowing users to program conditional execution sequences, and (3) that the programming not only eliminates the need to enumerate the successive commands, it also significantly reduces the task completion time.

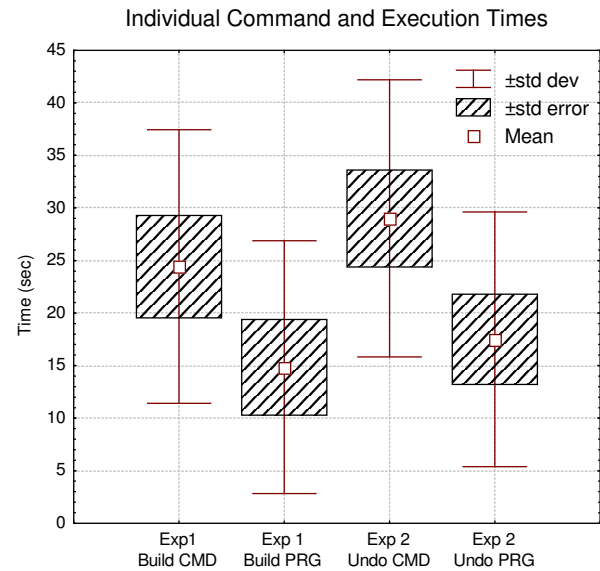


Fig. 4. Average command execution times for the Building (Exp1) and Undoing (Exp2) task using spoken language for on-line commanding (CMD) and for macro programming (PRG).

B. Experiment 2: Disassembling the Table

Part of the utility of a spoken language capability for robot programming (as for traditional programming) will be to allow the human user to program arbitrary behaviors, rather than just a single behavior. Experiment 2 thus tests whether the system that was developed for assembling the table could equally well serve to allow the user to program the robot to aid in disassembling the table.

Disassembly resembles the reverse process of assembly. User1 will first ask the robot to hold the table while he removes the leg. User1 will then put the robot into the “give it to me” posture, with the left hand open, give it the leg, and then issue to “prepare” command so that the robot passes the leg to User2 who then puts it away. For ease of reading, we dispense with the portion of the dialog in which the robot asks for confirmation of each command.

1) Phase 1. Spoken language commanding.

I am ready. Hold. *Getting ready to hold.* <see Fig 3D> *I am ready.* Right open. *Opening right hand.* *I am ready.* Right close. *Closing right hand.* *I am ready.* Wait. *Waiting for your signal.* <User1 unscrews the first leg while the robot holds the table – see Fig 3E.> Continue. *I am ready.* Right open. *Opening right hand.* <User1 takes the table and places it on the work surface – see Fig 3F> *I am ready.* Prepare. *Preparing to grasp.* <Right hand descends, left hand raises, preparing to take the leg> *I am ready.* Give it to me *Giving you the object.* <robot orients left hand to

User1>. *I am ready. Left open. Opening left hand. I am ready. Left close. <User1 places the first table leg within the robot's left hand.> Closing left hand. I am ready. Prepare. Preparing to grasp <Robot orients to User2> . I am ready. Left open. <User2 takes the leg.> I am ready.*

2) Phase 2: Programming the Robot

As in the previous experiment, after one run with the first leg, User1 is now ready to program the robot.

Learn. *I'll start learning now. Hold. Getting ready to hold. I am ready. Right close. Closing right hand <to grasp the table>. Wait. Waiting for your signal <while User1 removes the leg> Continue. I am ready. Right open. Opening right hand <so user1 can take the table.> I am ready. Prepare. Preparing to grasp. <robot raises left hand>. I am ready. Give it to me. Giving you the object <robot orients open left hand to user> I am ready. Left close. Closing left hand <to grasp the leg>; Prepare. Preparing to grasp. <orienting the left hand with table leg to User2> I am ready. Left open. Opening left hand <so that User2 can take the leg>. OK. *Ok we will store this plan.**

3) First Execution of Stored Program Macro

I am ready. Macro. Running the macro. Getting ready to hold. Closing right hand. Waiting for your signal. <while User1 unscrews the leg>. Continue. Opening right hand <> Preparing to grasp. Giving you the object. Closing left hand. Preparing to grasp. Opening left hand <to give to User2>

The second execution of the macro is identical.

4) Performance analysis

As in Experiment 1, the use of the programming capability for the third and fourth leg (executed in 2:51 and 2:51 respectively) yielded significant reductions in execution time as compared with the first two legs (executed in 3:57 and 4:11 respectively). To compare performance in the two experiments we performed a 3 way ANOVA with the factors Experiment (Exp1 vs. Exp2), Programming (with vs. without, i.e. PRG vs CMD), and Repetition (First vs. second repetition in each condition). Figure 4 indicates that both for Exp1 and Exp2 the completion times were elevated for the CMD vs PRG conditions, i.e. action execution was slower when programming was not used. The ANOVA revealed that only the Programming effect was significant ($F(1,6) = 277, p < 0.0001$).

V. DISCUSSION

Over the past several years we have experimented with spoken language control of different robot systems including the AIBO ERS-7, the Khepera mobile robot, and the Lynx-6

arm (see <http://dominey.perso.cegetel.net/RobotDemos.htm> for video demos) [4, 9-12]. Part of our goal in these efforts has been to develop a generic system for commanding and programming robots that can be rapidly adapted to new robotic platforms.

1) Lessons learned

In this context, the current research has yielded for the first time, the ability for a human user to employ spoken language to program a humanoid robot in real time to participate with humans in two distinct cooperative and complex object manipulation tasks. Despite this positive outcome, however, we have not yet fully exploited the potential richness of the predicate-argument structure of grammatical constructions. There are two important considerations to note here. First, a 3 month field study with an interacting robot [13] concluded that expectations on language-based interfaces have often been too high, and that “we need rich empirical experience of the use of robust and simple systems in order to formulate new and relevant questions,” justifying our simplified (and successful) approach. Second, this simplified approach has aided us in generating requirements for higher level predicate argument representations for robot action and perception that will allow us to more deeply exploit the communicative richness of natural spoken language.

2) Future Research

In particular, we have concluded that both for perception and action we require an interface to the robot based on predicated-argument (PA) representations. The framework for this ongoing development is presented in Fig. 5. As stated, we have previously demonstrated how grammatical constructions can be used to make the mapping between predicate-argument representation of action, and natural language sentences that describe that action [4]. In Fig. 5, this corresponds to the Perception path from the robot to Event Perception that generates an perceptual PA representation of the event. This is then passed to the “PA meaning to Sentence transformation” (based on the learned grammatical construction), which is finally converted to synthesized speech. This has been implemented in [4]. In order to exploit this PA capability in the domain of programming the HRP-2, we must provide the robot with a richer vocabulary of perceptual and action PA operators. Perceptual PAs (PPAs) would be of the nature Left-of(leg1, table-surface), or Attached(leg2, table-surface). Action PAs (APAs) would be of the nature Transport(Robot, leg1, user1) (i.e. an action in which the robot gives leg1 to user1). These PAs will be under the control of the Supervision function described in Section III.B. The encoding of action sequences as Composite PA sequences will be a direct extension of the current macro capability.

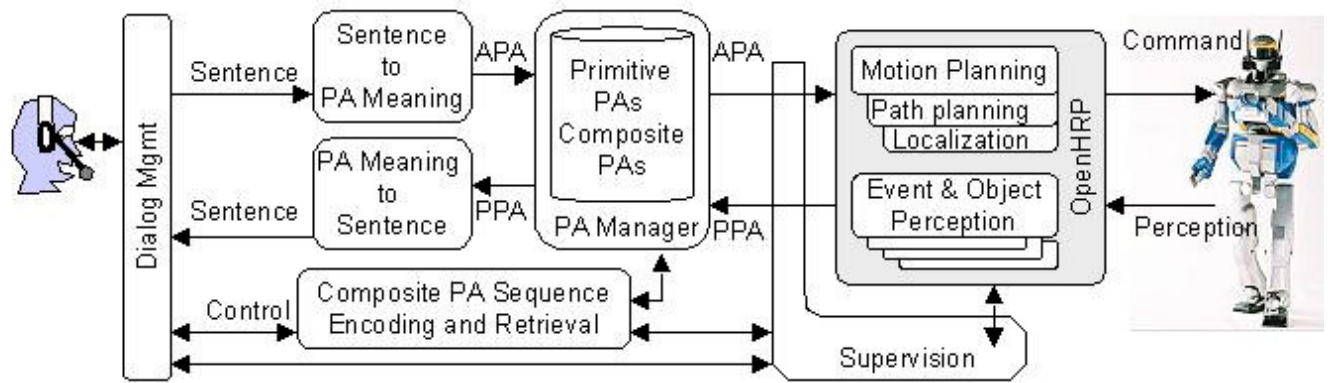


Figure 5. Spoken language control architecture. The fundamental unit of meaning is the predicate-argument (PA) representation. APA – Action Predicate-Argument representation. PPA – Perceptual Predicate-Argument representation. The Supervision layer coordinates all robot perception and action functions in OpenHRP. It is connected to Dialog Management, Composite Sequencing and PA Manager from which it receives APAs and returns PPAs. PPAs are built from the data produced mainly by perception modules of OpenHRP. The APAs are dynamically translated into complex sequences and executed by motion generation modules. Robot command : Command sentence transformed to its APA and sent to Supervision for Motion Planning. Robot perception : Perceptual (vision) input transformed to its PPA and sent to PA-Meaning-to-Sentence transformation, and then to speech synthesis by Dialog Management. Programming : Sequences of APAs can be learned as in the current study for simple commands. Execution can be made conditional based on perceptual states in PPAs.

A principal effort will be in developing new APAs and PPAs through learning. This can be achieved in part via well documented action learning methods that include demonstration and imitation [14-17]. Likewise, we have recently developed a system that generalizes over multiple experiences to acquire a representation of human-robot cooperative actions [12]. Interestingly the same system can be used to learn perceptual relations, as it does not differentiate between PAs with respect to whether they represent actions or perceptions. Indeed, in this manner, action sequences that are conditional on certain perceptual conditions holding can be learned. Thus, by starting with a simple system, we have achieved initial success in the spoken language programming of the HRP-2, and laid the groundwork for continued progress in this domain.

ACKNOWLEDGMENT

The authors thank Jean-Paul Laumond, Co-Director of the French-Japanese Joint Robotics Laboratory of the CNRS for support, and comments on the ms. We also thank Hajime Saito of General Robotix for invaluable assistance in technical aspects of the interface to OpenHRP.

REFERENCES

- [1] Crangle C. & Suppes P. (1994) Language and Learning for Robots, CSLI lecture notes: no. 41, Stanford.
- [2] Pickering MJ, Garrod S. (2004) Toward a mechanistic psychology of dialogue. *Behav Brain Sci.* Apr;27(2):169-90
- [3] Goldberg A. Constructions: A new theoretical approach to language. *Trends in Cognitive Sciences* 2003; 7: 219-24.
- [4] Dominey PF, Boucher (2005) Learning To Talk About Events From Narrated Video in the Construction Grammar Framework, *Artificial Intelligence*, 167 (2005) 31-61
- [5] Lauria S, Buggmann G, Kyriacou T, Klein E (2002) Mobile robot programming using natural language. *Robotics and Autonomous Systems* 38(3-4) 171-181
- [6] K.Kaneko, F.Kanehiro, S.Kajita, H.Hirukawa, T.Kawasaki, M.Hirata, K.Akachi, and T.Isozumi, "Humanoid robot hrp-2," in *Proceedings of the 2004 IEEE International Conference on Robotics & Automation*, vol. 2, 2004, pp. 1083-1090.
- [7] F. Kanehiro, N. Miyata, S. Kajita, K. Fujiwara, H. Hirukawa, Y. Nakamura, K. Yamane, I. Kohara, Y. Kawamura, and Y. Sankai, "Virtual Humanoid Robot Platform to Develop Controllers of Real Humanoid Robots without Porting," *Proc. Int. Conference on Intelligent Robots and Systems*, pp. 1093-1099, 2001.
- [8] Alami R., Chatila R., Fleury S., Ghallab M., Ingrand F.. (1998) An architecture for autonomy. *International Journal of Robotic Research*, Vol.17, N°4, pp.315-337.
- [9] Dominey, P.F., 2003. Learning grammatical constructions from narrated video events for human-robot interaction. *Proceedings IEEE Humanoid Robotics Conference*, Karlsruhe, Germany
- [10] Dominey, P. F., Boucher, J. D., & Inui, T. (2004). Building an adaptive spoken language interface for perceptually grounded human-robot interaction. In *Proceedings of the IEEE-RAS/RSJ international conference on humanoid robots*.
- [11] Dominey PF, Alvarez M, Gao B, Jeambrun M, Weitzenfeld A, Medrano A (2005) Robot Command, Interrogation and Teaching via Social Interaction, *Proc. IEEE Conf. On Humanoid Robotics 2005*
- [12] Boucher J-D, Dominey PF (2006) Programming by Cooperation: Perceptual-Motor Sequence Learning via Human-Robot Interaction, *Proc. Simulation of Adaptive Behavior*, Rome 2006.
- [13] Severinson-Eklund K., Green A., Hüttenrauch H., Social and collaborative aspects of interaction with a service robot, *Robotics and Autonomous Systems* 42 (2003) 223-234
- [14] Nicolescu M.N., Mataric M.J. : Learning and Interacting in Human-Robot Domains, *IEEE Trans. Sys. Man Cybernetics B*, 31(5) 419-430.
- [15] Zöllner R., Asfour T., Dillman R.: Programming by Demonstration: Dual-Arm Manipulation Tasks for Humanoid Robots. *Proc IEEE/RSJ Intern. Conf on Intelligent Robots and systems (IROS 2004)*.
- [16] Calinon S, Guenter F, Billard A (2006) On Learning the Statistical Representation of a Task and Generalizing it to Various Contexts. *Proc IEEE/ICRA 2006*.
- [17] Goga, I., Billard, A. (2005), Development of goal-directed imitation, object manipulation and language in humans and robots. In M. A. Arbib (ed.), *Action to Language via the Mirror Neuron System*, Cambridge University Press (in press).