Grammar-Based Robot Control
(Extended Abstract)

Richard Kelley
University of Nevada, Reno
1664 N. Virginia St.
Reno, Nevada 89557-0042
rkelley@cse.unr.edu

Monica Nicolescu
University of Nevada, Reno
1664 N. Virginia St.
Reno, Nevada 89557-0042
monica@cse.unr.edu

Mircea Nicolescu
University of Nevada, Reno
1664 N. Virginia St.
Reno, Nevada 89557-0042
mircea@cse.unr.edu

ABSTRACT

To allow a robot to function in unstructured environments, a control architecture should be flexible and should facilitate the creation and representation of arbitrarily complex sequences of actions. Such an architecture should also facilitate simple and natural interaction between a robot and its human collaborators. Human language has both representational flexibility and naturalness to human operators; in this paper we show how the structures of natural language can serve as the basis of an effective robot control architecture.

Categories and Subject Descriptors

I.2.8 [**Artificial Intelligence**]: Problem Solving, Control Methods, and Search; I.2.7 [**Artificial Intelligence**]: Natural Language Processing—*language parsing and understanding*; I.2.9 [**Artificial Intelligence**]: Robotics—*operator interfaces*

General Terms

Algorithms, Theory, Languages

Keywords

grammar-based control, language understanding, human-robot interaction

1. INTRODUCTION

As robots move from the factory into homes, offices, and other environments that are complex and unpredictable, it will be increasingly important for the software that controls those robots to be both adaptive and flexible. Such software should also be explicitly designed to promote effective human-robot interactions.

In this paper, we contend that a control architecture based on natural language will satisfy all of these requirements. In particular, we introduce *grammar-based control*, which uses the grammatical structure of natural language as a guide to parse commands received from a human operator. Natural language is flexible, extensible, and easy for the vast majority of humans to use in some capacity. We aim to build control architectures that inherit some of these abilities by mimicking the structures of natural language as closely as possible.

Cite as: Grammar-Based Robot Control, (Extended Abstract), Richard Kelley, Monica Nicolescu, Mircea Nicolescu, *Proc. of 8th Int. Conf. on Autonomous Agents and Multiagent Systems (AAMAS 2009),* Decker, Sichman, Sierra and Castelfranchi (eds.), May, 10–15, 2009, Budapest, Hungary, pp. 1153–1154

2. RELATED WORK

In recent years, researchers have begun looking into using natural language approaches to control a robot. In [3], speech-based natural language commands are used to control a robot. To train the system, though, expertise is required in speech recognition, language processing, and robotics. The system uses an explicitly defined grammar that is domain dependent. Speech recognition also is trained under the assumption that the domain of application is known. Our system finds inspiration in the overall approach taken in [3], but we work to avoid some of the challenges that seem to limit the system described there.

The work described in [1] advocates integration of language and action, claiming such integration is faithful to our understanding of human communications and substantially improves human-robot interaction. We show that this idea can inspire the design of a complete architecture that links language and behavior with the goal of creating effective interactions.

3. CONTROL ARCHITECTURE

Our control architecture starts with a command from a human operator, represented as a sentence in a natural language (in our case English), and uses a probabilistic parser to build a tree structure representing the grammatical structure of the command. That structure is then used to generate a sequence of behaviors that the robot executes, along with any side conditions that the robot has been programmed to execute at all times.

The command can be generated from several different types of interface: typing, speech, gestures, etc. In our case, we use a simple gesture system in which a user makes gestures using a Nintendo Wii remote. The gestures are classified into English words, and a sequence of such words is interpreted as a sentence to be parsed.

Once we have a command in textual form, the first step is to build a tree that represents the grammatical structure of the command, including the parts of speech of all of the words in the command. To do this we use a probabilistic context-free grammar. This allows us to determine a most probable parse for a given sentence.

The result of this process is a tree whose leaves are the words of our input command, and whose internal nodes represent possible grammatical constructions, such as “NP” for

Table 1: Some grammatical constructs, tags, and their interpretations in our architecture

Construct	Tag	Our usage
Sentences	S1	Command Sequences
Simple declarative clauses	S	Behaviors
Noun phrases	NP	Part or all of the robot
Noun phrases	NP	A known external object
Verb phrases	VP	Partially specified action
Coordinating Conjunction	CC	Sequencing of actions
Personal pronouns	PRP	"you," the robot

noun phrases. For a list of some of the grammatical constructions our system works with at this time, see Table 1.

Once we have a tree representation of our command, the final step is to build an executable sequence of actions that the command represents. The general strategy employed here is a left-to-right postorder traversal of the tree. As we descend through the tree, we construct objects representing partially specified actions in a sequence. When we reach the leaves of the tree, we use the information at those leaves to ascend up the tree, completing the specifications of the actions that have already been constructed as we go. The particular objects that are created depend on the tags encountered during descent, and the ways in which those objects are initialized depend on the particular words in the command that are encountered during ascent through the tree.

An adaptive system must be capable of learning; our system implements a preliminary form of learning. Given a word and a behavior sequence, we can command our controller to associate that behavior with the word so that in the future, the human operator need only use the word to describe a potentially complicated sequence of actions.

4. IMPLEMENTATION

We have implemented a preliminary version of the system using the Wii remote interface and a collection of very basic primitive actions that are predefined. In this section we'll focus on three types of commands that our system can process: simple commands, conditional commands, and sequenced commands.

The simplest commands are those such as `activate camera`. When the controller is done executing such commands, it passes control to the next command it has received, or waits for the next command to arrive. Such commands are useful for one-shot actions such as activating or deactivating a sensor.

Simple commands are clearly inadequate for real-world scenarios. To handle basic decision-making, we introduce *conditions* that the controller uses to decide when an action should start or stop. Such conditions might include `obstacle`, which is true when a robot's range finder detects an object within a predefined threshold. This would include commands such as `advance until obstacle`.

The last class of commands is arguably the most important: we implement the sequencing of actions using the coordinating conjunction "and." To carry out this sequencing, our algorithm continues as in the case of simple commands. However, once a behavior has been fully specified by processing all of the subtree at an S node, the algorithm continues from left to right and encounters a subtree containing a CC as its root node, and which in turn has as its sole child "and." Upon encountering this branch, the algorithm moves on and, for well-formed commands, encounters another S node whose subtree represents another command. It then creates another entry in the action sequence and initializes it. This may be done, in principle, an arbitrary number of times.

During execution, the controller steps through the sequence of behaviors and executes each one in turn. The process finishes when either the entire sequence of behaviors has been executed, or one behavior in the sequence begins executing but has no stopping condition.

As an example, we have the command

```
you start camera and you advance
```

Which has the robot start its camera and advance. This command results in the parse tree shown in Figure 1. In this case termination for the first action is not a problem, for the reasons discussed above. The robot activates its camera, and then begins execution of the second action, during which the robot rotates in a fixed way determined by the internal structure of the predefined "advance" action.

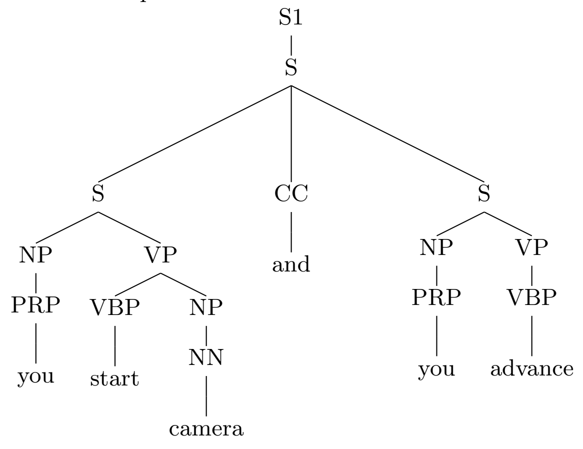

Figure 1: The parse tree of a command consisting of two simpler commands in sequence.

5. CONCLUSION

In this paper we presented a robot controller based on the grammar of natural language. We contend that such a controller obtains the flexibility, naturalness, and ease of use that robot controllers will need as robots become more ubiquitous in all areas of society in the near future.

6. REFERENCES

[1] T. Brick, P. Schermerhorn, and M. Scheutz. Speech and action: Integration of action and language for mobile robots. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems*, October 2007.

[2] E. Charniak. A maximum-entropy-inspired parser. In *Proceedings of NAACL-2000*, 2000.

[3] S. Lauria, G. Bugmann, T. Kyriacou, and E. Klein. Mobile robot programming using natural language. *Robotics and Autonomous Systems*, 38(3-4), 2002.