Hack and Kluge

CPT Pete Beim, Ian Horswill and **Ivan Yen**
Computer Science Department
Northwestern University
{beim,ian,iyen}@cs.nwu.edu

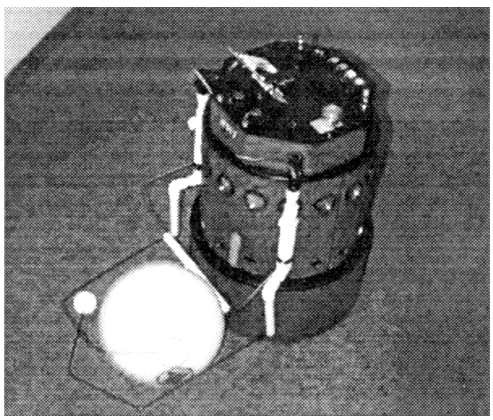

Figure 1: Kluge the robot during a fetch-and-deliver task.

Introduction

Hack and Kluge are high performance, low cost vision-based robots developed at Northwestern University. Hack and Kluge have a number of interesting features:

- They use a novel architecture that supports problem solving, reasoning, and instruction following, without a centralized world model
- All inferences are grounded in perception and updated at 5-10Hz
- Nearly all sensing is performed using real-time vision
- All computation is performed on-board
- They use very low cost hardware (equivalent systems could be built for under $10,000)

Hardware

Hack and Kluge use a modified RWI B14 enclosure that has been shortened to reduce weight and improve mechanical stability. Almost all sensing is based on color vision, although some amount of odometry is also used. All computation is performed on-board. The main processor is an MIT/DIdeas Cheap Vision Machine (a 50MFLOP digital signal processor with 256K words of SRAM and a memory-mapped RGB frame grabber). The CVM is connected to a Chinon CX-062 NTSC color surveillance camera. Ancilary microcontrollers drive the motors and monitor low bandwidth inputs such as user switches. Kluge is also equipped with a simple set of mandibles designed for gripping and carrying toy balls. Both machines have DECTalk speech synthesizers.

Active vision operators

The robots run a number of active vision operators in parallel. They have a simplified reimplementation of the Polly navigation system (Horswill 1993) that provides vision-based collision avoidance, boundary following, freespace following, and junction detection. The robots also have a simplified implementation of current theories of human visual search (Treisman and Gelade 1980; Koch and Ullman 1985). The system implements both pop-up search and conjunctive serial search using return-inhibition (Chapman 1990). However, the system also performs real-time color-based tracking of the selected objects using a modified K-means clustering technique. The system can run all vision algorithms, including three copies of the tracker, and recompute all inferences, at 5-10Hz depending on the number of inference rules and tracking operations.

Role-passing architecture

Symbolic problem solvers traditionally require a pre-existing database of true facts expressed in some suitable KR language. While convenient for the problem solver, this places an unreasonable burden on the perceptual system which must:

- Fill the database with all the information that might be needed by the problem solver
- Translate its internal representations into the model's KR language
- Update the database in real time

- Inform the problem solver when premises have changed and inferences must be updated

These problems are severe enough that symbolic problem solvers running on real robots do not update their databases with perception, or only update carefully selected portions of it. Often times, the bulk of the model is entered by hand.

While one can imagine many ways of modifying SMPE architectures to make the database easier to generate, we have taken the approach of directly interfacing the problem solver to the perceptual system, eliminating the database entirely. We have developed a technique called *role-passing* that preserves the efficiency and parallelism of behavior-based systems while supporting restricted forms of logical reasoning. A role-passing system consists of:

- A set sensory *pools*
- A set of parameterized behaviors
- A set of inference rules for triggering behaviors

A sensory pool is a collection of identical sensory processes (e.g. visual trackers) that can represent a specific kind of sensory information about an object. Each sensory process can be dynamically bound to one or more indexical-functional names, or *roles*. Behaviors communicate with pools by specifying the role about which they need information. The sensory process bound to that role then answers with the appropriate information. In a sense, role-passing provides a more programable version of deictic representation (Agre 1988). Using role-passing, we have developed techniques for mapping a useful subset of modal logic with equality into fast feed-forward networks grounded by sensory data (Horswill 1997). We have also shown how backtracking inference systems such as Prolog can be driven by perceptual data using role-passing (Horswill 1995).

At present, Hack and Kluge implement a *tracker pool* for tracking the visual positions of objects, an *odometer pool* for dead reckoning the positions of objects out of view, and a *description pool* for remembering visual descriptions (color, size) of objects.

Example task

A typical task for Kluge would be to follow a natural language instruction such as "bring the blue ball here," "follow me," or "bring Hack the green ball." A simple finite-state parser reads the instruction and binds the appropriate sensory processes to the roles OBJECT (the object to carry or follow) and DESTINATION (the place or object to deliver to), then sets the appropriate goal, usually either DELIVER or FOLLOW.

Once the initial role bindings are established, the inference engine posts the goal of having the OBJECT in the mandibles. This progates through the network to post the subgoal of being near the OBJECT, which posts the subgoal of binding the OBJECT in the tracker pool. The inference network then triggers a visual search for OBJECT, causing the tracker pool bind a tracker to OBJECT and set it looking for the color specified in the description pool. When the tracker acquires its target, the inference network marks the distance and orientation of OBJECT as known, allowing the driving behaviors of the robot to activate. After driving to and grabbing the object, the goal of the OBJECT being in the hand becomes satisfied and the inference engine posts the goal of being near the DESTINATION, repeating the visual search and driving processes.

Since all inference is performed in a feed-forward network, inferences are updated every clock tick (100-200ms). This allows the robot to respond to contingencies as soon as they are sensed. For example, if someone steals the ball while the robot is trying to deliver it, the inference network immediately marks the ball as not being in the hand, retracts the goal of being near the DESTINATION, and reposts the goal of acquiring the OBJECT.

Acknowledgments

Support for this research was provided in part by the National Science Foundation under grant number IRI–9407273. The Institute for the Learning Sciences was established in 1989 with the support of Anderson Consulting, part of the Arthur Anderson Worldwide Organization.

References

Philip E. Agre. The dynamic structure of everyday life. Technical Report 1085, Massachusetts Institute of Technology, Artificial Intelligence Lab, October 1988.

David Chapman. Vision, instruction, and action. Technical Report 1204, Massachusetts Institute of Technology, Artificial Intelligence Lab, April 1990.

Ian Horswill. Polly: A vision-based artificial agent. In *Proceedings of the Eleventh National Conference on Artificial Intelligence*, pages 824–829, Cambridge, MA, 1993. AAAI, MIT Press.

Ian Horswill. Visual routines and visual search. In *Proceedings of the 14th International Joint Conference on Artificial Intelligence*, Montreal, August 1995.

Ian Horswill. Real-time control of attention and behavior in a logical framework. In W. Lewis Johnson, editor, *Proceedings of the First International Conference on Autonomous Agents*, pages 130–137, Marina del Rey, CA USA, February 1997. ACM SIGART, ACM Press.

C. Koch and S. Ullman. Shifts in selective visual attention: Towards the underlying neural circuitry. *Human Neurobiology*, 2:219–227, 1985.

Anne M. Treisman and Garry Gelade. A feature integration theory of attention. *Cognitive Psychology*, 12:97–136, 1980.