Using snarpy to Connect a KR System to Pyro

Debra T. Burhans
Department of Computer Science
Canisius College
2001 Main Street
Buffalo, NY 14208
burhansd@canisius.edu

Alistair E. R. Campbell
Department of Computer Science
Hamilton College
198 College Hill Road
Clinton, NY 13323
acampbel@hamilton.edu

Introduction

This work involves the development of a layered, heterogeneous architecture for cognitive robotics that can be used to connect a knowledge representation (KR) system to Pyro (Python robotics). We have focused on connecting a Semantic Network Processing System (SNePS) (Shapiro & Rapaport 1992) agent to a Pyro robot, however, the framework we have developed, snarpy (an architecture linking SNePS with Pyro), is extremely flexible and allows for the integration of any formal knowledge representation, reasoning, and acting system with Pyro.

Cognitive Robotics is focused on higher-level actions and perceptions rather than lower-level details of robot interaction with a world. Pyro is a Python-based environment for robotics (Blank *et al.* 2006; Blank, Meeden, & Kumar 2003; http://www.pyrorobotics.org), and indeed, for cognitive robotics, that provides underlying support for a variety of actual and simulated robots by making available a large set of high-level primitives for interacting with sensors and effectors. We have built additional layers on top of Pyro so that we can use a powerful KR system, capable of reasoning and communicating about its actions as it performs them, to enhance the capabilities of Pyro agents. While our interest and investigations have to date involve using the KR system to control the robot's actions and plans, the KR component of the system could just as easily serve as an auxiliary, powerful reasoner for a Pyro agent to call upon.

Architecture

The architecture we have implemented, GLAIR (Grounded Layer Architecture with Integrated Reasoning) (Hexmoor, Lammens, & Shapiro 1993), is one of a number of multi-layer models that have been proposed for robot control. In GLAIR, a robotic agent is realized in 5 distinct layers of processing. The top layer (KL) is where conscious reasoning and acting decisions occur. The lowest layer (SAL) is where the agent physically operates in its environment. The layers in between (PMLa-c) serve to mediate and translate high-level decisions to actuators and sensors (PML is a perceptual-motor-layer). In our architecture a Pyro brain comprises the PML-c, namely, the lowest perceptual-motor level which interacts directly with the robot (SAL).

Our cognitive robot is realized on a single computer in an architecture involving two concurrent processes. The higher brain is a Lisp process running SNePS; the lower brain is a Python process running Pyro. The higher brain encompasses the first three GLAIR layers: KL, PMLa, and the lisp side of PMLb. The lower brain encompasses the Python side of PMLb, PMLc and SAL, including the robot machinery itself. The architecture is shown in figure 1.

The large gray rectangle on top represents the Lisp process; the one on the bottom represents Python. The GLAIR software layers are indicated with hashed regions and labeled. The robot dog represents the SAL. While the figure includes an AIBO robot it could be replaced by any of the robots supported by Pyro. Control is transferred from the SNePS system through the GLAIR layers to Pyro, which signals the hardware. RobotIPC is a communications protocol we designed for this project to facilitate interprocess communication using Unix message queues. While it is associated with a Lisp and a Python process in this example it could just as easily be employed to connect agent components in other languages.

Experiments

To test the feasibility of the snarpy architecture, we performed two experiments. The first (obstacle-avoid) served to demonstrate the way in which an existing Pyro brain[1] can be mapped onto the proposed architecture and involves simple obstacle avoidance with a simulated Pioneer robot in a two-dimensional world. The second (cha-cha) involves using a single brain to evoke the same behavior in two different robots.

The obstacle-avoid system ultimately controls a simulated Pioneer robot. The original Pyro brain mentioned above polls the range sensor data to see if it's getting too close to an object in front of it or towards its side. If the robot is clear it continues to move in its present direction, if not it adjusts by turning away from the object in its path.

In our system the Pyro brain, representing the PML-c, merely communicates data from the robot (SAL) to the

[1] Avoid.py, included with Pyro

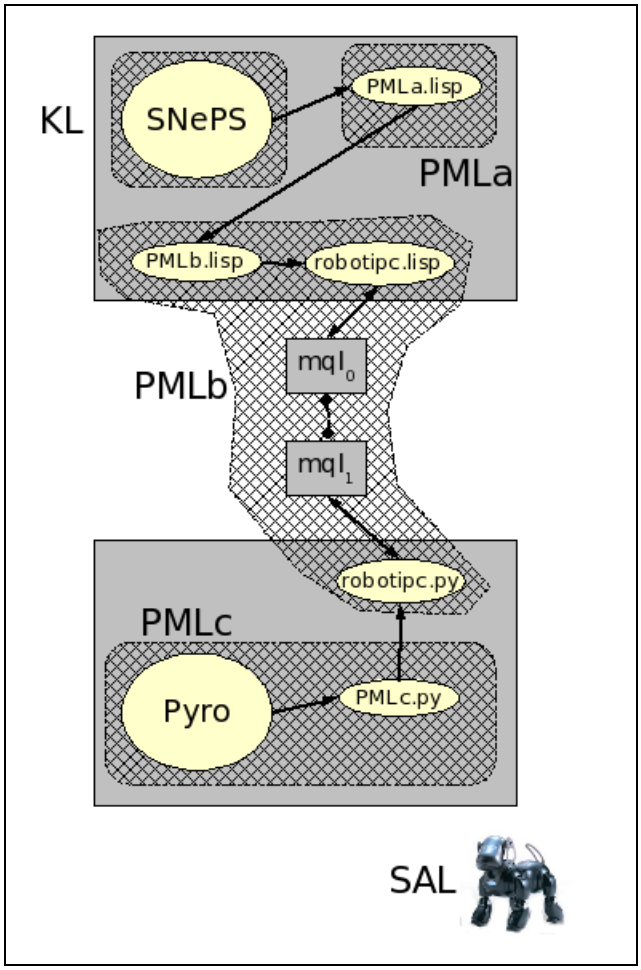

Figure 1: The snarpy Robot Architecture

PML-b when asked, and causes the robot to move in a particular manner when told, with this message coming via the PML-b.

The PML-b mediates between the PML-c and the PML-a, and does not have direct access to the KL. It is this level that spans Python and Lisp and enables message passing between the two processes.

The PML-a is where primitive actions are defined. These actions are accessible to the KL, but can not be reasoned about by the KL. For example, the agent may have a KL concept of avoiding an obstacle and a plan for avoiding an obstacle but the actual required actions, such as associating specific sensor readings with movements, might be realized as a set of primitives at the PML-a level (subconscious rather than conscious).

In practice there is no clear *a priori* demarcation between these levels: decisions about what belongs at the conscious and subconscious levels depend on the viewpoint of the system designer. In terms of performance the snarpy agent is indistinguishable from the original Pyro agent, a user would not be able to tell from observing the robot in the simulated world which was which.

The cha-cha system can be used, without changes, to control either a simulated Pioneer or an AIBO robotic dog. This is due to the fact that the system only involves forward and backward movement and does not entail processing any sensor data: the sensory capabilities of the AIBO and Pioneer are very different. The purpose of this experiment was simply to test the effects of running the same snarpy configuration on two very different robot platforms. The agents basic cognitive process is simply to do the cha-cha five times. In the simulated world the Pioneer can be observed moving foward and backward five times. Similarly, the dog moves foward and backward five times. This experiment demonstrates the value of including Pyro as part of the snarpy architecture with its ability to generalize across robots at the level of a Pyro brain.

Summary and Future Work

We have introduced a new member of the GLAIR family of cognitive robotics architectures. We have briefly outlined the architecture of snarpy agents. Future agents will have increased capability at the knowledge level to make conscious decisions about actions and will connect to more interesting sensor and actuator interfaces in the Pyro framework such as vision. In particular, we are developing an embodied AIBO Wumpus-hunting agent that will employ a SNePS developed KL for the problem(Shapiro & Kandefer 2005).

By using SNePS at the top of a layered architecture, cognitive robotic agents are capable of reasoning, acting, and responding to human queries about what they know, what they have seen and what they have done.

References

Blank, D.; Kumar, D.; Meeden, L.; and Yanco, H. 2006. The Pyro toolkit for AI and robotics. *AI Magazine* 27(1). special issue on robots and robotics in education.

Blank, D. S.; Meeden, L.; and Kumar, D. 2003. Python robotics: An environment for exploring robotics beyond legos. In *ACM Special Interest Group: Computer Science Education Conference*.

Hexmoor, H.; Lammens, J.; and Shapiro, S. C. 1993. Embodiment in GLAIR: a grounded layered architecture with integrated reasoning for autonomous agents. In Dankel II, D. D., and Stewman, J., eds., *Proceedings of The Sixth Florida AI Research Symposium (FLAIRS 93)*, 383–404. The Florida AI Research Society.

http://www.pyrorobotics.org. Pyro Website.

Shapiro, S. C., and Kandefer, M. 2005. A SNePS approach to the wumpus world agent or cassie meets the wumpus. In Morgenstern, L., and Pagnucco, M., eds., *IJCAI-05 Workshop on Nonmonotonic Reasoning, Action, and Change (NRAC'05): Working Notes*, 96–103. Edinburgh, Scotland: IJCAI.

Shapiro, S. C., and Rapaport, W. J. 1992. The SNePS family. *Computers & Mathematics with Applications* 23(2–5):243–275.