Sensor-to-symbol Reasoning for Embedded Intelligence

David Kortenkamp and **Patrick Beeson**
TRACLabs Inc.
korten@traclabs.com

Nick Cassimatis
RPI
cassin@rpi.edu

Abstract

Sensor-to-symbol conversion lies at the heart of all embedded intelligent systems. The everyday world occupied by human stakeholders is dominated by objects that have symbolic labels. For an embedded intelligent system to operate in such a world it must also be able to segment its sensory stream into objects and label those objects appropriately. It is our position that development of a consistent and flexible sensor-to-symbol reasoning system (or architecture) is a key component of embedded intelligence.

Introduction

An embedded, intelligent system has sensors that connect it to the world. These sensors generate low-level data that represent the physical characteristics of the world. For many applications, including those involving humans or higher-level intelligent systems, these low-level data streams must be converted to objects that represent collections of sensory data and those objects must be labeled with symbols. It is our position that a sensor-to-symbol architecture that regularizes this connection between the sensed physical world and the symbols that represent that world is a critical component of embedded intelligence and one that has been under-researched.

As an example, suppose an apple is placed in front of the system's sensors and the symbol "apple" is generated or linked to. The sensor-to-symbol architecture maintains the coherence of the symbol over time. For example, if something temporarily occludes the apple from the sensors and then the apple re-appears the architecture will not create a new apple symbol. If another apple appears in a different place at about the same time then a second apple symbol needs to be created as the same apple cannot be in two places at once. This is called the symbol grounding problem for computer systems (Harnad 1990; Steels 2002) or the symbol anchoring problem in robotics (Coradeschi and Saffiotti 2003).

Of course, the examples just given assume that the sensor-to-symbol architecture has the sensing capability to recognize an apple and also knowledge about how to recognize an apple. The former can be determined by analyzing the sensing characteristics of the system on which the sensor-to-symbol architecture is running. The latter can be done by having an *ontology* that captures the inherent properties of symbolic objects. The sensor-to-symbol architecture can then connect sensors to ontological objects. In order to be general purpose the sensor-to-symbol architecture needs to allow for different combinations of sensors (both physical sensors and virtual sensors, which are software algorithms that run on sensed information) and ontologies without requiring changes to the architecture's core reasoning mechanisms.

An overarching goal of a sensor-to-symbol architecture should be to do for sensing what planning representations and algorithms have done for action selection. That is, allow for the creation of dynamic combinations of sensors to achieve symbolic sensing goals. What has been *ad hoc* in the past will become formalized with general purpose reasoning mechanisms and descriptive representations. In this analogy, the object ontology is equivalent to the planning world model. Formal descriptions of sensors are equivalent to actions descriptions in languages such as STRIPS (Fikes and Nilsson 1971), PDDL (Fox and Long 2003) and ANML (Smith and Cushing 2008). And the reasoning substrate is equivalent to general planning search engines such as Graphplan (Kambhampati, Parker, and Lambrecht 1997) or O-plan (Currie, Tate, and Bridge 1991) or FF plan (Hoffmann and Nebel 2001). Just as planning systems offer flexibility with respect to solving wide ranges of problems the architecture described in this report offers a flexible, scalable and extendable sensor- and ontology-based system for use by applications such as robotics and intelligence, surveillance and reconnaissance (ISR) tasks.

Characteristics

The characteristics of a general sensor-to-symbol architecture can be grouped into several broad categories. The first category concerns reasoning about sensors and ontological symbols. The second category concerns converting sensors to symbols. The third category involves changing the system's behavior based on its sensors.

Sensor reasoning

Typically sensor-based systems are built around a pre-specified set of sensors. For example a specific UAV or

UGV. Adding (or subtracting) a sensor to these systems requires rewriting software. Similarly, the objects that such systems are built to recognize are also pre-specified. A characteristic of a general sensor-to-symbol architecture is that the sensors and the objects can be read in at system run time. Thus, the same architecture (and reasoning mechanisms) could be used for a robot searching for biological weapons and for a satellite searching for tanks in the desert. If a new sensor or sensing algorithm becomes available it could be immediately connected into the reasoning mechanisms. If a sensor is lost (due to damage or environmental constraints) the system would be able to determine what objects could still be distinguished. Similarly, if a new ontology is given to a robot or system then new objects could be recognized to the limit of the sensors. Thus, changing a satellite from searching for tanks to searching for hostiles planting IEDs would be a matter of changing ontologies, assuming the right sensory algorithms existed in both cases.

Sensor to symbol conversion

In many domains, especially surveillance, a human is tasked with watching a video feed or other sensory stream and detecting useful objects that are of interest to either that person or someone else. This is time-consuming and costly. A sensor-to-symbol architecture can watch various sensory streams and match the outputs of simple or complex sensory algorithms onto ontological classes. Thus, given sufficient sensory coverage and a sufficiently complete ontology the sensor-to-symbol problem can be solved. This is comparable to how a planner takes a complete set of actions, an initial condition and a goal conditions and maps actions to progressive world state changes. Thus freeing the human resources to operate at a higher level of abstraction.

Changing system behavior

The sensor descriptions required by the architecture describe the environmental conditions necessary for the sensor to work (e.g., light) and characteristics of the sensor such as range and line of sight. These descriptions can be used to tailor a system's behavior to match the sensors. For example, a radiation sensor that detects the level of radiation allows for a gradient search for the source of the radiation. A thermal sensor can see through some occlusions. A texture sensor needs to touch the object. Given the sensors and the objects the system's information gathering behaviors can radically change. This has the characteristic of automatically optimizing system behavior based on its sensors and its sensory goals.

Related work

The sensor-to-symbol problem has also been called the symbol grounding or symbol anchoring problem. A special issue of the Robotics and Automation Journal stated that "Anchoring is the problem of how to create, and to maintain in time, the connection between symbol- and signal-level representations of the same physical object" (Coradeschi and Saffiotti 2003). There have been several different approaches to this problem, although only a few general solutions have been proposed and implemented.

Saffiotti and his colleagues have developed an anchoring module capable of maintaining the correspondence between sensor data and symbolic descriptions referring to objects. It is also capable of tracking and acquiring information from observations derived from sensor-data as well as information from a priori symbolic concepts (Coradeschi and Saffiotti 2000). Their module is able to determine whether objects have been previously perceived to avoid creating duplicate symbols. They do not have the ability to easily change out sensors and ontologies, not do they deal with adding new objects or classes of objects to an ontology. More recently they have looked at diverse sensors, even olfactory sensors, and human interaction (Loutfi, Coradeschi, and Saffiotti 2005).

Another approach is to use perceptual markers to connect symbols to sensors. This came out of human visual research (Ullman 1984) and has mostly been associated with combining both the position of objects and their task function in the environment (Agre and Chapman 1987). This early work was extended for behavioral robots by Horswill (Horswill 1993). Markers were brought into the 3T robot control architecture (Bonasso et al. 1997) where they were used to connect information coming from the robot perception system to task symbols at the execution level (Kortenkamp, Huber, and Wasson 1999; Wasson, Kortenkamp, and Huber 1998). This research did not focus on avoiding duplicate symbols nor on extending an ontology. It did allow for maintaining a connection between symbols in the task (e.g., "track Bob") and perceptual information ("Bob is wearing a red shirt"). Shapiro and his colleagues also integrated anchoring mechanisms into their layered architecture called GLAIR and focus on aligning physical-level representations with those used for reasoning (Shapiro and Ismai 2003). Santos and Shanahan focus on abstracting stereo information into symbols and develop a theory of discourse for these symbols (Santos and Shanahan 2001).

Researchers at Vanderbilt and at NASA Johnson Space Center have developed an ego-sphere representation of objects around a robot that are relevant to its tasks (III et al. 2001; Johnson et al. 2002). The ego-sphere stores an object's location relative to the robot and updates that location based on the robot's motion as well as the last perceived motion of the object. This allows a robot, for example, to reach out and grab an object that it is not looking at using the last known information.

Ben Kuipers' group at the University of Texas has been exploring the interplay of sensor information and symbolic information in a number of ways. Their early work was focused on navigation and the process by which sensory information becomes symbols on both a topological and metric map (Kuipers and Byun 1991). This was pushed even further with work in which the robot was given almost no information about its sensors and needed to perform a complete mapping between sensors and symbolic places in the world (Pierce and Kuipers 1994). The most recent research has focused more on discovering objects using basic sensory information (Modayil and Kuipers 2004). Kuipers' student Joseph Modayil took this a step further in his PhD thesis and endeavored to learn an object ontology directly from sensory input (Modayil 2007).

Conclusion

A proper sensor-to-symbol architecture will require significant progress along several research fronts. In particular, the following research questions all stem from the core question: **How does the incoming, continuous sensor stream get turned into objects (symbols)?**:

1. Which of these objects/symbols is stored for later retrieval?
   - How does this depend on on-going tasks?
2. What properties of the objects/symbols get stored?
   - For recognition
   - For tasks (functions)
   - How are these properties determined?
   - How are they stored?
3. How are objects/symbols related to each other?
   - Spatially
   - Functionally
   - Task
   - Ontology
4. Can this process be jump-started?
5. What if it is not jump-started (tabula rosa)?
6. How can objects/symbols be maintained?
   - E.g., not having two instances of the same object in memory
   - Modeling dynamic objects
   - Persisting objects that are no longer seen
7. How can objects be deleted (forgotten)? Should they be?

References

Agre, P. E., and Chapman, D. 1987. Pengi – an implementation of a theory of activity. In *Proceedings of the Fifth National Conference on Artificial Intelligence*.

Bonasso, R. P.; Firby, R. J.; Gat, E.; Kortenkamp, D.; Miller, D. P.; and Slack, M. 1997. Experiences with an architecture for intelligent, reactive agents. *Journal of Experimental and Theoretical Artificial Intelligence* 9(1).

Coradeschi, S., and Saffiotti, A. 2000. Anchoring symbols to sensor data: preliminary report. In *Proc. of the 17th AAAI Conf.*, 129–135. Menlo Park, CA: AAAI Press. Online at http://www.aass.oru.se/~asaffio/.

Coradeschi, S., and Saffiotti, A. 2003. An introduction to the anchoring problem. *Robotics and Autonomous Systems* 43(2-3):85–96. Special issue on perceptual anchoring. Online at http://www.aass.oru.se/Agora/RAS02/.

Currie, K.; Tate, A.; and Bridge, S. 1991. O-plan: the open planning architecture. *Artificial Intelligence* 52:49–86.

Fikes, R. E., and Nilsson, N. J. 1971. Strips: A new approach to the application of theorem proving to problem solving. *Artificial Intelligence* 2.

Fox, M., and Long, D. 2003. Pddl2.1: An extension to pddl for expressing temporal planning domains. *Journal of Artificial Intelligence Research* 20:2003.

Harnad, S. 1990. The symbol grounding problem. *Physica D* 42.

Hoffmann, J., and Nebel, B. 2001. The ff planning system: Fast plan generation through heuristic search. *Journal of Artificial Intelligence Research* 14:253–302.

Horswill, I. 1993. Polly: A vision-based artificial agent. In *National Conference On Artificial Intelligence (AAAI-93)*.

III, R. A. P.; Hambuchen, K. E.; Kawamura, K.; and Wilkes, D. M. 2001. The sensory ego-sphere as a short-term memory for humanoids. In *Proceedings IEEE-RAS International Conference on Humanoid Robotics*.

Johnson, C.; Koku, A.; Kawamura, K.; and Peters, R. A. 2002. Enhancing a human-robot interfacing using sensory egosphere. In *Proceedings of the IEEE International Conference on Robotics and Automation (ICRA)*.

Kambhampati, S.; Parker, E.; and Lambrecht, E. 1997. Understanding and extending graphplan. In *In Proc. 4th European Conference on Planning*, 260–272.

Kortenkamp, D.; Huber, E.; and Wasson, G. 1999. Integrating a behavior-based approach to active stereo vision with an intelligent control architecture for mobile robots. In Kraetzschmar, G. K., and Palm, G., eds., *Hybrid Information Processing in Adaptive Autonomous Vehicles*. Berlin: Springer-Verlag.

Kuipers, B. J., and Byun, Y.-T. 1991. A robot exploration and mapping strategy based on a semantic hierarchy of spatial representations. *Robotics and Autonomous Systems* 8.

Loutfi, A.; Coradeschi, S.; and Saffiotti, A. 2005. Maintaining coherent perceptual information using anchoring. In *Proc. of the 19th IJCAI Conf.*

Modayil, J., and Kuipers, B. 2004. Bootstrap learning for object discovery. In *Proceedings of the IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS-04)*.

Modayil, J. 2007. *Robot Developmental Learning of an Object Ontology Grounded in Sensorimotor Experience*. Ph.D. Dissertation, Computer Sciences Department, University of Texas at Austin.

Pierce, D., and Kuipers, B. 1994. Learning to explore and build maps. In *Proceedings of the Twelfth National Conference on Artificial Intelligence (AAAI-94)*.

Santos, P. E., and Shanahan, M. P. 2001. From stereoscopic vision to symbolic representation. In *In Working notes of the AAAI Fall Symposium on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, 37–43. AAAI Press.

Shapiro, S. C., and Ismai, H. O. 2003. Anchoring in a grounded layered architecture with integrated reasoning. *Robotics and Autonomous Systems* 43(2-3):97–108.

Smith, D., and Cushing, W. 2008. The anml language. In *In Proceedings of the 9th International Symposium on Artificial Intelligence, Robotics and Automation for Space (i-SAIRAS)*.

Steels, L. 2002. Grounding symbols through evolutionary language games. In Cangelosi, A., and Parisi, D., eds., *Simulating the Evolution of Language*. London: Springer Verlag. chapter 10, 211–226.

Ullman, S. 1984. Visual routines. *Cognition* 18.

Wasson, G.; Kortenkamp, D.; and Huber, E. 1998. Integrating active perception with an autonomous robot architecture. In *Proceedings of the International Conference on Autonomous Agents*.