2002

A Developmental Approach to Intelligence

Doug Blank
Bryn Mawr College, dblank@brynmawr.edu

Deepak Kumar
Bryn Mawr College, dkumar@brynmawr.edu

Lisa Meeden

Citation

Blank, D.S., Kumar, D. and Meeden, L. (2002). A Developmental Approach to Intelligence. In Proceedings of the Thirteenth Annual Midwest Artificial Intelligence and Cognitive Science Society Conference, Edited by Sumali J. Conlon.

A Developmental Approach to Intelligence

Douglas Blank & Deepak Kumar
Department of Math & Computer Science
Bryn Mawr College
Bryn Mawr, PA 19010
{dblank,dkumar}@brynmawr.edu

Lisa Meeden
Computer Science
Swarthmore College
Swarthmore, PA 19081
meeden@cs.swarthmore.edu

Abstract

In this paper we present a developmental approach toward creating intelligent systems. Although embracing an emergent paradigm, we also propose an architecture such that symbols have their proper place. To explore this methodology, we are developing our system on real robots. Robotic approaches usually invoke the so-called "symbol grounding problem" (Harnad 1990). However, our developmental approach bypasses the problem by generating the symbols. Thus, in a sense, the symbols are inherently grounded to the physical objects they designate. We argue that a general theory of intelligence that takes a top-down perspective will be inherently incomplete.

Creating Intelligent Systems

The road to creating completely autonomous, intelligent systems turned out to be much rougher than anyone expected. At the onset of the trip, almost 50 years ago when the field of artificial intelligence (AI) was being created, researchers imagined that it would be a Sunday afternoon drive. Just two years into the trip, Herbert Simon and Allen Newell wrote:

> It is not my aim to surprise or shock you... [b]ut the simplest way I can summarize is to say that there are now in the world machines that think, that learn and create. Moreover, their ability to do these things is going to increase rapidly until—in the visible future—the range of problems they can handle will be co-extensive with the range to which the human mind has been applied. (Simon & Newell 1958)

Simon and Newell weren't alone in their early optimism. For example, Marvin Minsky, another founder of AI, said in 1967:

> ...within a generation the problem of creating 'artificial intelligence' will be substantially solved. (Minsky 1967)

Understandably, people outside the field asked persistently "are we there yet?" In the following two decades, the early hype turned to lost hope. By 1982, Minsky was quoted as saying that "the AI problem is one of the hardest science has ever undertaken." (Kolata 1982). Today, artificial intelligence is an active research area; however, there are few researchers actually pursuing the goal of creating completely autonomous, intelligent systems.

AI researchers, cognitive scientists, and philosophers are divided in opinion on how one could build an intelligent system. However, the field has developed a number of models and techniques which could form the foundation of a theory of machine intelligence. We see these models and techniques falling into two opposing methodological camps defined broadly as follows.

Symbolic AI is a top-down, centralized methodology based on the formal properties of well-defined symbols and logical reasoning. Expert systems and theorem provers are two examples of such systems.

Emergent AI is a bottom-up, decentralized methodology based on the self-organized interaction of many local subsystems. Artificial neural networks and evolutionary systems are two examples of such techniques. This camp is most often called "subsymbolic," but it has been argued elsewhere that the most important features of the models in this category are the emergent properties they exhibit. (Blank 2001).

Researchers are split over which of these two very different paradigms is the proper framework onto which to build a general theory that will scale-up to general intelligence. Most AI researchers probably advocate a symbolic framework, especially for generally intelligent systems.

Perhaps the best known quest for general machine intelligence is Douglas Lenat's Cyc project (Lenat 1998). Lenat uses symbolic AI techniques in the Cyc project to reason about the world by using a database of commonsense symbolic knowledge. For example, Cyc has been told that "trees are usually outdoors," that "once people die they stop buying things," and that "glasses of liquid should be carried right side-up" (Cycorp 2002). Cyc applies logical reasoning to all relevant facts in order to make deductions, plans, etc.

One can imagine the infinite bits of trivia that Cyc would need to be told in order to reason about our world in an intelligent manner. Furthermore, such a system must have a representation that will scale-up as the size of the knowledge base increases. Human performance actually speeds up with additional experience; however, typical top-down

systems bog down with increased information.

Rather than spoon feed each nugget of wisdom into a database, we are taking the approach that a system can experience the world as we do. In this approach, commonsense facts need not be explicitly elaborated, rather the system would be able to reason with knowledge gained through experience.

Our ultimate and longterm goal is to have a system with which we could talk, in English, about any general topic. For that, we will of course need a connection to a symbolic system. However, before we even begin to think about that level, we hope to have symbol-like representations form in a developmental, bottom-up fashion. Therefore, we are interested in robotics systems with the following properties:

Embodied: The robot is physically embedded. That is, it has a three dimensional physical structure in real space.

Sensors: It is endowed with some sensory apparatus that is used to perceive its environment.

Actuators: It is capable of physically changing the environment in some way, either by moving around or manipulating objects.

Symbols: Modeling of the behavior of the robot is carried out in a symbolic representation and reasoning system.

Our decision to use robots to explore general intelligence does not necessarily dictate that we use emergent techniques. There are a wide variety of methods that have been applied to robotics. But because the intelligent system problem is so difficult, it is no surprise that most of the solutions available thus far are either *ad hoc* and/or are based on very small-scale empirical studies, largely carried out in simulated domains.

We are, therefore, committed to using real robots to experience a realistic environment due to our choice to explore the emergent paradigm. The rest of this paper defines such an approach to building an intelligent robot.

Developmental robotics

A number of cognitive science researchers have recently argued that understanding the developmental processes in the brain is crucial to understanding intelligence (Elman *et al.* 1996). This surge of interest in development has led to the formation of a new conference on development and learning (ICDL[1]) and to a new subfield of robotics.

Rather than considering the task-oriented, largely top-down, approach to designing robots and behavior models, we are interested in a holistic, bottom-up approach to designing robot behaviors. In our approach, robots begin with a reflex model and slowly, over time, via interactions with the environment, acquire the knowledge necessary to exist in the environment in a purposeful way. Robots learn not only about their environment, but also about their own capabilities via this process (Kuipers 2000).

The bottom-up approach to self-learning, learning about its environment, and the actions it can perform, can be largely unsupervised, though incorporated in a hierarchical learning and control architecture described below. Ultimately, we envision this learning process to result in a symbolic model, not necessarily different from those used in top-down approaches. However, in the developmental approach, the robot's learning process gives "birth" to these symbolic concepts. As with the top-down approaches, the architecture is multi-layered but with a different decomposition since each layer is responsible for the development of 'higher-level' concepts. One of the goals or our research is to explore the range of middle-level representations required. At present, we are concentrating on at least three different levels: self-motivated sensory-motor control, quasi-symbolic representations of learned purposeful behaviors, and symbolic representations of acquired concepts and behaviors.

Related Work

Weng, one of the first advocates of the developmental approach to robotics, argues that automated development relieves human engineers from the explicit design of representations (Weng *et al.* 1999). His project, called SAIL (Self-organizing, Autonomous, Incremental Learner), relies on human trainers to guide a robot's behavior through explicit instruction. The robot's controller can never be directly altered by its teachers, but can only be updated via learning. In contrast, our approach is not teacher directed, but instead relies on self-discovery and a bootstrapping process to learn.

Heikkonen and Koikkalainen were some of the first researchers to demonstrate the feasibility of using a self-organizing map that associates sensors and motors as a controller (Heikkonen & Koikkalainen 1997). Furthermore, they showed that this association map could be developed incrementally through trial and error explorations using a simple learning rule. Their experiments were done in simulation and focused on the specific task of reaching a goal point given a global goal heading. In contrast, our work focuses on real, physical robots, which only have local sensor information, and does not attempt to solve any particular task. The goal of our work is broader—the discovery of concepts rather than success at a given task.

Millan uses reinforcement learning combined with a self-organizing map to incrementally develop a robot controller (J. del R. Millán 1997). Rather than learning from scratch, his architecture uses two types of bias: domain knowledge and advice. These can serve to accelerate learning and to focus the search on the most promising areas of the action space first. Although the right kind of bias can be beneficial, the wrong kind of bias can be detrimental. Because robots have very different sensory abilities than humans, the kinds of distinctions that are easy for us may be hard for robots and vice versa. Thus human advice may not be very helpful. Our approach allows the robot to discover appropriate categories based on its own sensory abilities.

Kuipers and Beeson have recently shown how a robotic system can "bootstrap" itself from knowing little about itself, to having detailed knowledge about its sensors and physical form (Kuipers & Beeson 2001). One major dif-

[1] www.egr.msu.edu/icdl02

ference between their work and our own is that our system's internal representations do not need to be justified to an outside observer. For example, our system uses self-organized, distributed representations to encode information about sensors and motors, whereas Kuipers and Beeson's system must show that information explicitly in a manner that makes sense to human observers. In addition, we aim to make our system self-motivated such that it has the ability and "desire" to explore unknown areas.

Neural networks for modeling development

Our architecture is primarily constructed from two types of neural network models: self-organizing maps and simple recurrent networks.

Self-organizing maps (SOMs) were pioneered by Kohonen in the 1980's and 1990's(Kohonen 2001). Briefly, a SOM is a mapping of a typically high-dimensional input vector to a particular cell in a low-dimensional matrix. The matrix is topologically arranged in a unsupervised manner such that very similar input vectors map to the same cell, and slightly similar input vectors map to nearby cells.

Specifically, similarity is computed by comparing an input vector with a model vector associated with each cell. The model vector that is closest (as determined by the smallest sum of squared differences to the input vector) is designated as the winner. The model vector of the winner and the model vectors of the cells in its neighborhood are updated to more closely match the given input vector.

The SOM idea is quite simple and effective. Any information that can be turned into a vector of numeric values can be self-organized into such a map. The idea has been applied in a wide variety of problems ranging from creating maps for classifying the World Wide Web to analyzing financial data. Resulting SOMs are useful for two related reasons: their ability to automatically find abstractions, and their ability to help visualize complex data (Kohonen 2001).

The simple recurrent network (SRN) was created by Elman in the late 1980's (Elman 1990). To understand its significance, one must realize that there are two main classes of artificial neural networks: those that are feed-forward (all activation flows in one direction) and those that are recurrent (activation is allowed to flow forward and backwards). In order to deal with time-dependent tasks, a feed-forward network usually requires a fixed window of the past inputs, while a recurrent network can take the current input alone and build up a contextual memory of the past inputs. Elman's SRN has the simplicity of a feed-forward network with the power of a recurrent network. Like SOMs, SRNs are also simple and effective. They have been applied to analyzing many types of sequential processing, including language and music.

Overview of a developmental architecture

Figure 1 shows an overview of the archtecture of our system. The architecture is both hierarchical and cyclical. It is hierarchical in that there are four distinct levels and each level builds abstractions based on the representations formed at the previous levels. It is cyclical in that the subsequent levels can feed their discovered abstractions back to previous

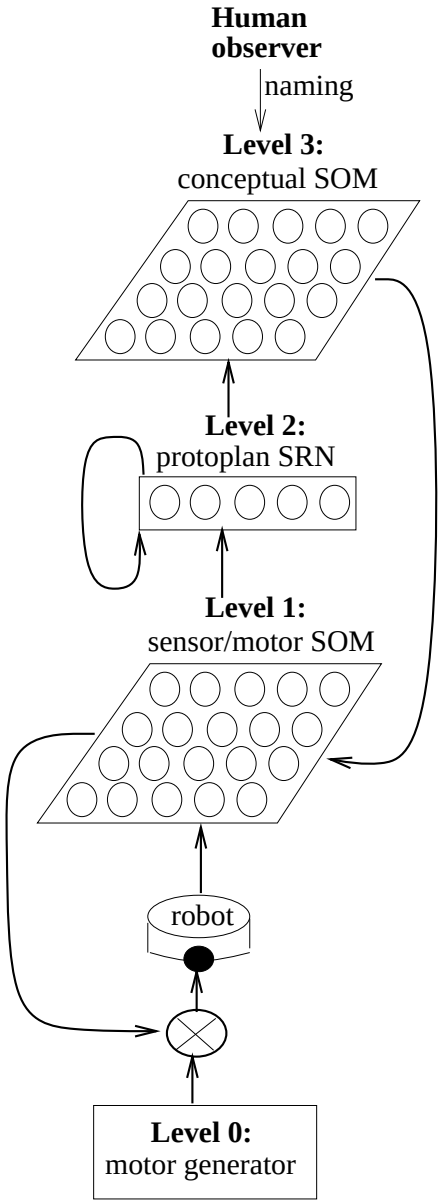

Figure 1: Architecture of our system

levels. This combination of hierarchy and feedback is essential to creating a continually evolving understanding of the world and how to behave in it. In this way, simple reactive behavior can develop into time-dependent planned behavior.

Innate knowledge is provided at Level 0, but in order to eliminate any preconceived notion of representation we assume that the relevant set of features at every subsequent level is unknown. Instead, the neural network components discover appropriate representations through the robot's interactions with the environment.

In order to start the entire developmental process, the system needs to experience the world. Only then will the robot have motor and sensor data to organize. However, it is exactly this control that we wish to develop. Our solution to this chicken and egg dependency is to begin with a very basic control law and bootstrap our way up.

Level 0 is a motor generator intended to model innate reflexes. These reflexes could be represented as a collection of basic if-then rules from perceptual situations to actions. Initially this is all the system knows how to do.

Level 1 observes the sensor and motor values that are produced as Level 0 controls the robot. Through this observation, Level 1 begins to form associations between sensors and motors and abstractions about sensors and motors within its self-organizing map. Eventually, when Level 1 has successfully captured the control information from Level 0, it can subsume control of the robot.

This subsumed control is similar in function to that defined by (Brooks 14 23). However, our method of subsumption is a by-product of the architecture of the system rather than designed by human engineers.

It is important to note that Level 0 may only know how to respond to particular sensor modalities, such as range sensing, and know nothing about how to respond to other modalities, such as vision. However, while Level 1 is learning to mimic Level 0, it is also learning about how all of the sensory modalities are correlated with one another. Thus, Level 1 will be able to respond appropriately to visual input, although Level 0 could not. This bootstrapping effect is crucial to the developmental process.

Level 2 observes the sensor/motor associations developed within the self-organizing map of Level 1. Level 2 is trained to predict what the next Level 1 state will be given its current state. This prediction task will force Level 2 to use its recurrent connections to recognize *sequences* of sensor/motor associations through time. Previous work (Meeden 1994; 1996) has shown that this type of simple recurrent network will develop representations of multi-step behaviors, that have been termed *protoplans*.

Level 3 observes the protoplans developed within Level 2 and begins to categorize them. The behavior concepts formed in Level 3 are fed back into Level 1 and can serve as longer-term goals. A human observer can watch the robot behaving and attach names to the various behaviors such as "approaching a wall", "following a wall", "avoiding an obstacle on the left", and so on. Once these labels have been attached to the emergent concepts of Level 3, the robot can be directed by a planner to perform particular behaviors by activating the appropriate units in Level 3.

Level 3 is where traditional symbolic representations are formed. We intend to use the SNePS BDI architecture to interface as a planner with this level (Kumar 1996). This architecture has an integrated model of acting and inference that is based on several semantic, architectural, and ontological commitments that are important to the integration of low-level learning abilities (Kumar & Meeden 1998). Agents modeled using the BDI architecture are capable of representing conceptual entities about which they can have beliefs, reason about them, and act on them. The agent's representations of conceptual entities could be formed as a result of interactions with a human as well as via concept discovery from lower level learning mechanisms.

Next we present the experimental results from our initial explorations into the feasibility of this developmental architecture.

All of our experiments are being carried out on actual physical robots. We have access to several robots of varying sensorimotor capabilities: Pioneer 2's, Nomads, B21R, and Kheperas. Given the developmental nature of our approach, it will be essential to carry out the entire developmental process from scratch on each robot platform. Our aim is to demonstrate the viability of our architecture by reporting consistent results on several different robot platforms.

Initially, we will primarily be using the Khepera II robots, which are a second generation version of Kheperas (Mondada, Franzi, & Ienne 1993). Each Khepera II has two motors on a small, circular body. The robots come equipped with six infrared sensors in the front and two in the back that are sensitive to light as well as obstacles. Additionally, the robots have a video camera capable of delivering a color image of size 510x492 pixels (see Figure 2). The robot is tethered to a host computer where all the modeling and analysis is performed.

Preliminary experimental results

A developmental approach necessitates incremental testing. Currently we have only begun to test Level 0 and Level 1. However, at the conclusion of this section we will point to some abstractions that are already forming in Level 1 that hold promise for the success of the proposed architecture.

Our initial Level 0 uses two fuzzy logic rules (similar in spirit to Saphira-style behaviors (Konolige & Myers 1998)) to do obstacle avoidance. This purely reactive controller slows down as it nears an obstacle, while at the same time turning away from the obstacle. This Level 0 is a perfect, reactive obstacle avoidance controller based only on the eight infrared sensors. It is perfect in that it never runs into an obstacle. It is reactive in that it only makes decisions based on current IR readings, and nothing else.

During the training of Level 1, the sensor and motor values produced by Level 0 are concatenated together into a single vector and passed to the self-organizing map as input. The inputs to the map include a 48x32 black and white camera image, 8 infrared readings, and 2 motor values, for an input vector of length 1546. Figure 3 shows a typical training image taken with the Khepera's onboard camera. The

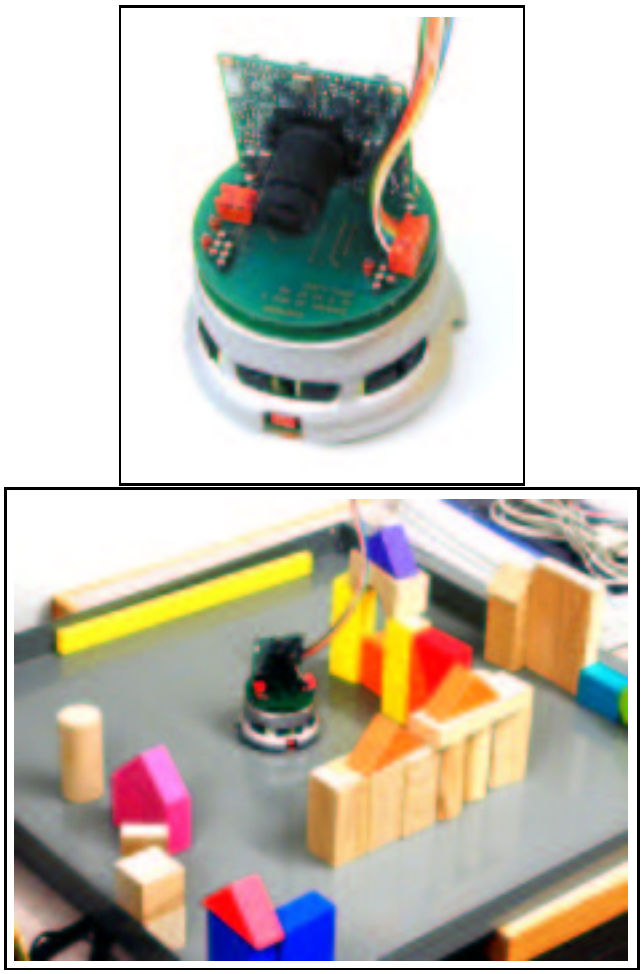

Figure 2: A Khepera-II Robot with a 2-D camera (above) and its world (below)

Figure 3: A typical 48x32 grayscale image taken from the Khepera's onboard camera. A house-shaped tower of blocks can be seen.

Step	Lmotor	Rmotor	Step	Lmotor	Rmotor
A1	0.58	0.58	B1	0.61	0.58
A2	0.97	0.06	B2	0.61	0.58
A3	0.81	0.16	B3	0.58	0.55
A4	0.84	0.16	B4	0.13	0.90
			B5	0.58	0.58

Table 1: Two behavior sequences from the training data.

map itself contains 24x16 units with a hexagonal neighborhood.

After training, Level 1 can be used as the controller, subsuming Level 0. This process works as follows. The current sensor values are saved into a vector as before, but the positions in the vector associated with the motor values are set to don't care values. This vector is used to activate the trained map of Level 1, and the closest matching model vector is found. From this model vector, the previously associated motor values can be retrieved and used to control the robot.

In experiments performed so far, the Level 1 map is only an approximation of the original controller. Because the model vector from the Level 1 map is actually an averaging of all of the matching motor/sensor vectors, and also nearby cells' model vectors, it tends to blur fine-grained differences. For example, using the model vectors to control the Khepera captures about 93% of the performance of the original Level 0 controller.

In order to better understand the abstractions formed in the map of Level 1, we analyzed several sequences of actions from the training set shown in Table 1. In steps A1-A4, the Khepera robot encountered an obstacle on its left and made a hard right turn. In steps B1-B5, the robot encountered an obstacle on its right and made a quick left turn before continuing straight ahead.

Figure 4 shows the trajectories through Level 1's map of the winning units as these steps were processed. Note that in both sequences, at the instigation of the turn in step A2 and step B4, the winning node is in a topologically distant area of the map from where it was on the previous time step. The map has made a clear distinction between going straight and turning. Also note that the trajectory of a left turn is quite different from the trajectory of a right turn. There are certainly many more subtle distinctions being made within the map as well, such as when to make a hard turn versus a gradual turn based on the pattern of IR readings.

It is clear that the Level 1 map has already made some important abstractions about the robot's interaction with its environment. Although we have not yet begun to test Level 2, it's recurrent network is capable of predicting the distinct trajectories from Level 1, and from this facility it can begin to build time-dependent summaries of behavior, which are a key to creating grounded plans.

Towards self-motivation

We believe that the above-described hierarchical and cyclical learning architecture is of the type necessary for creating an intelligent system. However, most learning systems do not continually keep learning; they do not keep building

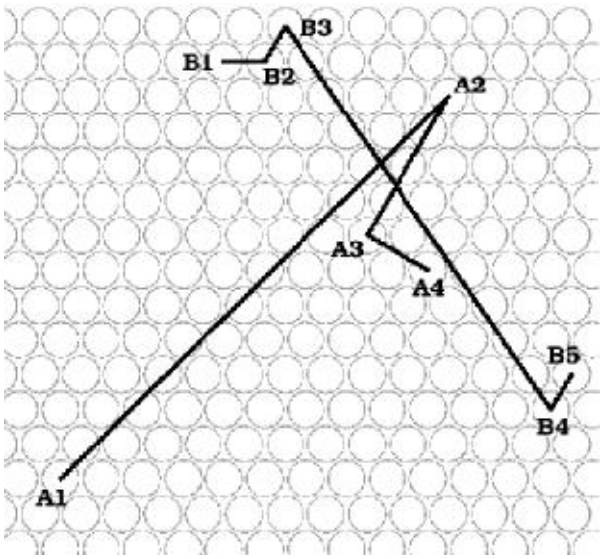

Figure 4: Two trajectories through a portion of Level 1's senor/motor map. Sequence A1-A4 occurred during a sharp right turn. Sequence B1-B5 occurred during a quick left turn. See Table 1 for the associated motor values.

more sophisticated representations on top of previously built representations (Thrun 1995). In fact, most learning systems make just a tiny leap from what they know, to what they almost know. What is missing?

We believe that a model of self-motivation could drive the system to continued development. Such internally driven artificial systems are rare. One such mechanism is the competitive arms race of co-evolutionary systems (see, for example, (Juillé & Pollack 1996)). The basic idea of the competitive arms race is that two populations are pitted against one another, and gradually one-up each other in a spiralling increase of fitness. This works if the two populations begin about equal, and remain relatively even throughout the race.

Our goal is to find a developmental equivalent to a co-evolutionary competitive arms race. We imagine a mechanism in the system such that it would get "bored" in environments that it can easily predict, but retreat from environments that seem chaotic. However, the area between predictability and randomness (the so-called "edge of chaos") is suspected to be a prime area for learning (see, for example, (Langton 1991)). The exact nature of such a mechanism has yet to be explored.

Summary and Conclusions

We are just beginning a trip to explore a developmental approach to intelligent systems. We believe that a cyclic architecture in combination with a driving model of self-motivation will create an emergent system that has the ability of continued learning. There is, no doubt, a long road ahead.

Acknowledgments

We would like to thank Paul Grobstein for interesting discussions in these topics.

References

[Blank 2001] Blank, D. S. 2001. Radical artificial intelligence: A postmodern approach. In Moore., ed., *Proceedings of the 12th Midwest Artificial Intelligence and Cognitive Science Conference*.

[Brooks 14 23] Brooks, R. A. 14–23. A robust layered control system for a mobile robot. *IEEE Journal of Robotics and Automation*.

[Cycorp 2002] Cycorp, I. 2002. http://www.cyc.com/overview.html.

[Elman *et al.* 1996] Elman, J.; Bates, E.; Johnson, M.; Karmiloff-Smith, A.; Parisi, D.; and Plunkett, K. 1996. *Rethinking Innateness: A Connectionist Perspective on Development*. MIT Press.

[Elman 1990] Elman, J. L. 1990. Finding structure in time. *Cognitive Science* 14:179–211.

[Harnad 1990] Harnad, S. 1990. The symbol grounding problem. *Physica D* 42:335–346.

[Heikkonen & Koikkalainen 1997] Heikkonen, J., and Koikkalainen, P. 1997. Self-Organization and Autonomous Robots. In Omid Omidvar & Patrick van der Smagt., ed., *Neural Systems for Robotics*. Academic Press. chapter 10, 297–337.

[J. del R. Millán 1997] J. del R. Millán. 1997. Incremental Acquisition of Local Networks for the Control of Autonomous Robots. *Seventh International Conference on Artificial Neural Networks*.

[Juillé & Pollack 1996] Juillé, H., and Pollack, J. B. 1996. Dynamics of co-evolutionary learning. In *Proceedings of the Fourth International Conference on Simulation of Adaptive Behavior*, 526–534. MIT Press.

[Kohonen 2001] Kohonen, T. 2001. *Self-Organizing Maps*. Springer, Third Edition edition.

[Kolata 1982] Kolata, G. 1982. How can computers get common sense? *Science* 217:1237.

[Konolige & Myers 1998] Konolige, K., and Myers, K. 1998. The saphira architecture for autonomous mobile robots. In Kortenkamp, D.; Bonasso, R. P.; and Murphy, R., eds., *Artificial Intelligence and Mobile Robots*. MIT Press.

[Kuipers & Beeson 2001] Kuipers, B., and Beeson, P. 2001. Toward bootstrap learning for place recognition. In Coradeschi, S., and Saffiotti, A., eds., *Papers from the 2001 AAAI Fall Symposium on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, number 01-01 in FS. AAAI Press.

[Kuipers 2000] Kuipers, B. 2000. The spatial semantic hierarchy. *Artificial Intelligence* 119:191–233.

[Kumar & Meeden 1998] Kumar, D., and Meeden, L. 1998. A Hybrid BDI Architecture for Modeling Embedded Rational Agents. In *Proc. of the AAAI Symposium on Cognitive Robotics*, 84–90. AAAI Press.

[Kumar 1996] Kumar, D. 1996. The SNePS BDI architecture. *Journal of Decision Support Systems* 16:3–19.

[Langton 1991] Langton, C. 1991. Computation at the edge of chaos: Phase transitions and emergent computation. In Forest, S., ed., *Emergent Computation*. MIT Press. 12–37.

[Lenat 1998] Lenat, D. B. 1998. From 2001 to 2001: Common sense and the mind of hal. In Stork, D. G., ed., *Hal's Legacy: 2001's Computer as Dream and Reality*. MIT Press.

[Meeden 1994] Meeden, L. 1994. *Towards planning: Incremental investigations into adaptive robot control*. Ph.D. Dissertation, Indiana University.

[Meeden 1996] Meeden, L. 1996. An incremental approach to developing intelligent neural network controllers for robots. *IEEE Transactions on Systems, Man, and Cybernetics, Part B: Cybernetics* 26(3):474–485.

[Minsky 1967] Minsky, M. 1967. *Computation: Finite and Infinite Machines*. Prentice-Hall.

[Mondada, Franzi, & Ienne 1993] Mondada, F.; Franzi, E.; and Ienne, P. 1993. Mobile robot miniaturization: A tool for investigation of control algorithms. In *Proceedings of ISER93*.

[Simon & Newell 1958] Simon, H., and Newell, A. 1958. Heuristic problem solving: The next advance in operations research. *Operations Research* 6.

[Thrun 1995] Thrun, S. 1995. A lifelong learning perspective for mobile robot control. In Graefe, V., ed., *Intelligent Robots and Systems*. Elsevier.

[Weng *et al.* 1999] Weng, J.; Hwang, W. S.; Zhang, Y.; and Evans, C. H. 1999. Developmental robots: Theory, method and experimental results. In *Proc. of the 2nd Int. Symp. on Humanoid Robots*.