2002

A Developmental Approach to Anchoring

Doug Blank
Bryn Mawr College, dblank@brynmawr.edu

Deepak Kumar
Bryn Mawr College, dkumar@brynmawr.edu

Lisa Meeden

A Developmental Approach to Anchoring

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 towards anchoring of symbols in robotic systems. Rather than viewing the anchoring problem as a maintenance of connections between symbols and corresponding perceptual objects, we present a bottom-up approach in which "anchored symbols" are generated via a developmental process. Thus, in a sense, the symbols are inherently anchored to the physical objects they designate. We motivate this approach by presenting an architecture that develops conceptual representations through its interactions with the environment. We argue that a general theory of anchoring that takes a top-down perspective will be inherently incomplete.

Keywords
developmental robotics, self-organization, symbol grounding

Corresponding Author
Lisa Meeden
Phone: (610) 328-8565
Fax: (610) 328-8606

1 The anchoring problem

Anchoring has been defined as, "the process of creating and maintaining the correspondence between symbols and percepts that refer to the same physical objects" [3]. It is further identified as a necessary property of any physically embedded system that is also endowed by a symbolic reasoning system. In [3], an attempt is made to provide a domain-independent definition of anchoring. The definition, as well as a prescribed anchoring process, is described in terms of finding, reacquiring, and tracking anchors for given symbols in real time. We argue that this methodology is weak for various theoretical as well as pragmatic reasons. Therefore, any attempt to derive a general theory of anchoring based on this formulation will result in an incomplete theory.

Any robot demonstrating the capabilities outlined above is required to have at least 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. If there were no symbols to manipulate, there would be no anchoring problem.

It is important to outline the properties above since any solution to the so-called anchoring problem has to coordinate these in some manner.

1.1 Approaches

The anchoring problem is a hard problem. 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 suggest that, as described in [3], the anchoring problem may be considered AI-Complete [20]. That is, solving the anchoring problem is equivalent to solving the entire *AI problem*. This is easy to see, since computational vision is considered AI-Complete, and any system dealing with the perceptual system inherently includes vision. It is important to explore and analyze the issue of anchoring from many perspectives.

Current approaches to the anchoring problem can be broadly categorized into top-down and bottom-up approaches. In a top-down approach, the anchoring problem becomes that of aligning symbols to feature sets in the robot's perceptual stream [1, 2, 4, 8, 9, 22]. That is, at any given time, the robot has all the relevant symbols which then need to be correctly associated with physical objects in the world, based

on its perceptions. Typically, the perceptual stream is extremely rich, and of very high bandwidth, thereby rendering the anchoring problem hard to do in real time. Most demonstrations of top-down solutions to the anchoring problem pertain to simplified domains, with a small number of well characterized (easy to recognize) objects.

Bottom-up approaches employ some model(s) of sensor data interpretation in order to identify relevant properties of the world [12, 13, 19]. In some experiments, attempts have been made at deriving symbolic representations resulting from interpreted sensor data.

In both approaches there has been varying degrees of success. In what follows, we present our analysis and perspectives on the anchoring problem. We will subsequently present a proposal for an extendable architecture for a mobile robot's control system. Our methodology takes a bottom-up approach, though ultimately, it is our goal to incorporate a full symbolic-level capability.

1.2 Problems with anchoring

All descriptions of the anchoring problem begin by assuming that the system/robot already has *all* the symbols that it needs to carry out purposeful behavior. When dealing with real percepts, a robot is bound to perceive objects, and situations that have not yet been conceptualized. Hence, there is a need for a bottom-up facility for recognizing new objects and novel situations.

In most top-down approaches, it is assumed that the anchoring problem is *only* about anchoring symbols representing objects to actual physical objects. This assumption is based on the strategy that actions of a robot manipulate objects, and hence those are the entities that should be anchored. Anchoring can and should be done not only on objects, but also on situations, and, perhaps most importantly, on actions. After all, based on the capabilities outlined in Section 1 above, a robot is physically embodied with a three-dimensional structure in space. It is also typically capable of moving about in the world. Therefore, it makes sense to anchor the symbolic concepts of a robot's motor actions to actual physical movements of the robot. That is, a simple action such as *turnleft*, should have an anchoring in the motor movements of the robot.

A robot's symbolic processing capabilities should be compatible with its physical attributes. Most physical robots used in anchoring experiments have limited sensorimotor capabilities. Yet, the symbolic 'brains' driving these robots tend to have models with much more sophisticated capabilities.

1.3 The context of anchoring

We would like to identify two kinds of anchoring: *context-free anchoring (CFA)* and *context-sensitive anchoring (CSA)*. A typical example of CFA is a robot tracking an object in a visual scene. For example, the object might be a colored ball rolling

on the floor. In this situation, the surrounding scene is not important (i.e., context-free), and should be ignored by the tracking system. The problem is then reduced to that of perceptual processing. Thus, the symbol "ball3" can be anchored via a tight coupling with the perceptual correlate of the ball.

Context-sensitive anchoring takes into account the current context of a situation. The perceptual correlate is only part of the picture, so to speak. The goals of the robot, its current intentions, beliefs, etc., might also play a part in the task. This is a harder problem, one that encompasses the entire symbol grounding problem [7].

Even context-free anchoring poses many problems. If the ball, in the example above, becomes occluded, then the coupling is broken, and *ad hoc* methods must be introduced in order to reestablish perceptual contact. Also, if a second ball enters the scene, it must be viewed as a separate symbol and tracked. As more objects enter the scene, the problem becomes intractable.

Anchoring should be considered as a way of creating correspondences between concepts and perceptual correlates, but also between concepts and behaviors. For example, "Going down a hall" should be anchored in a similar manner as "the blue ball". Any attempt to do only the latter does not need a general solution and consequently will not lead to a general theory. We will conclude this discussion on anchoring by posing the following *Turing-like* thought experiment.

> The goal of anchoring could be described as attempting to get a robot to appreciate a simple magic trick. Consider taking a coin and placing it in the palm of your hand, and then closing your hand. A robot observing these actions, and relying on context-free anchoring, might conclude that because it can no longer see the coin that it has disappeared and begin applauding. However, a robot relying on context-sensitive anchoring would assume the coin was in the closed hand. This robot would not applaud until the hand was opened and the coin was seen to be gone.

2 Developmental robotics

A number of cognitive science researchers have recently argued that understanding the developmental processes in the brain is crucial to understanding intelligence [5]. 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 [12]. The bottom-up approach to self-learning, learning about its environment, and the

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

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 for symbol anchoring. 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.

3 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 [21]. His project, called SAIL (Self-organizing, Autonomous, Incremental Learner), relys 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 [8]. 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 [9]. 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 huamn advice may not be very helpful. Our approach allows the robot to discover appropriate categories based on its own sensory abilities.

3.1 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[10]. 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 [10].

The simple recurrent network (SRN) was pioneered by Elman in the late 1980's [6]. 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.

3.2 Overview of a developmental architecture

Figure 1 shows an overview of the architecture of our system. The architecture is hierarchical and consists of four distinct levels. Each level builds abstractions from the representations formed at the previous level until concept-like entities are developed at the top level. 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.

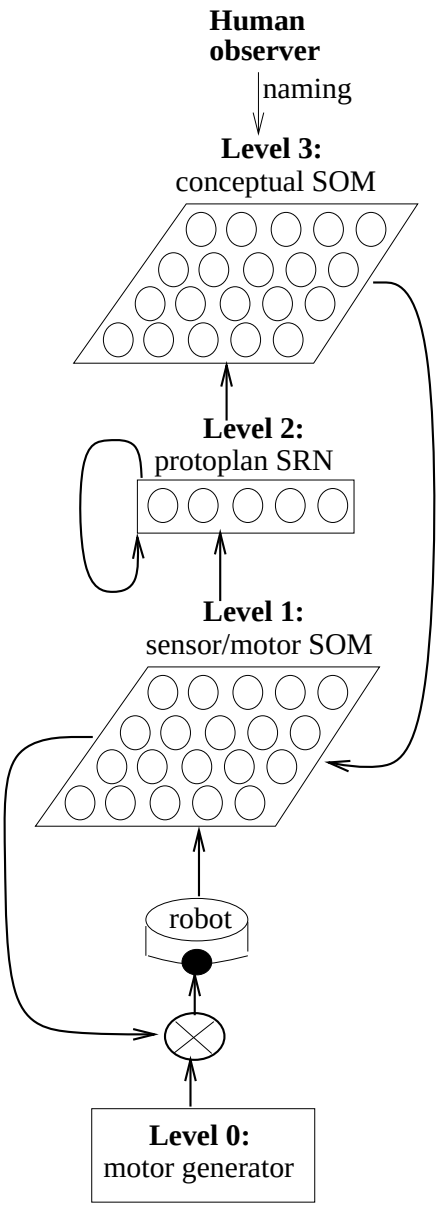

Figure 1: Architecture of our system

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.

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 [16, 17] 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 [14]. 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 [15]. 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. This provides the facilities needed for context-sensitive anchoring. 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 leavel learning mechanisms.

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

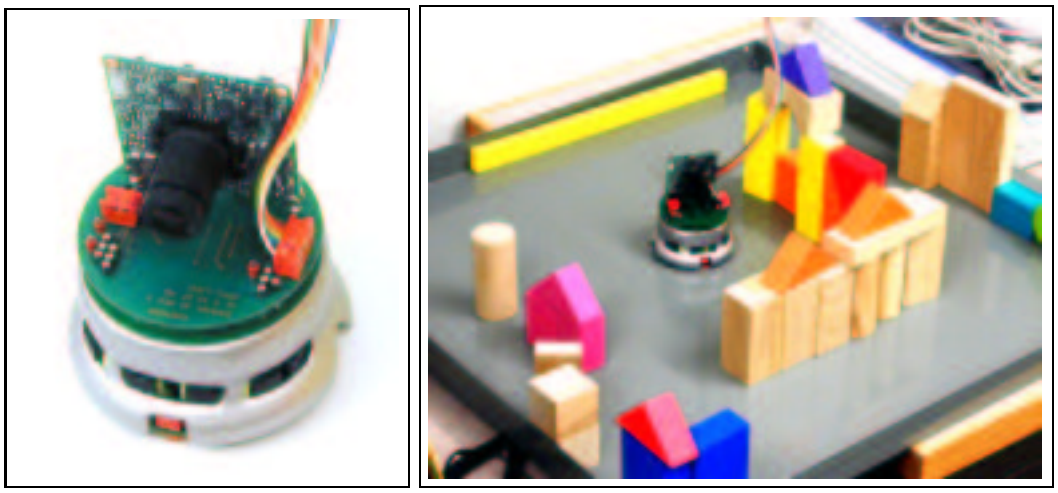

Figure 2: A Khepera-II Robot with a 2-D camera (left) and its world (right)

4 Preliminary experimental results

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 [18]. 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.

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 [11]) 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

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

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 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 70% of the performance of the original Level 0 controller. This performance should be further increased by giving equal weighting to the motor components as compared to the visual components when building the map. This weighting technique has proved successful in previous work [8].

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

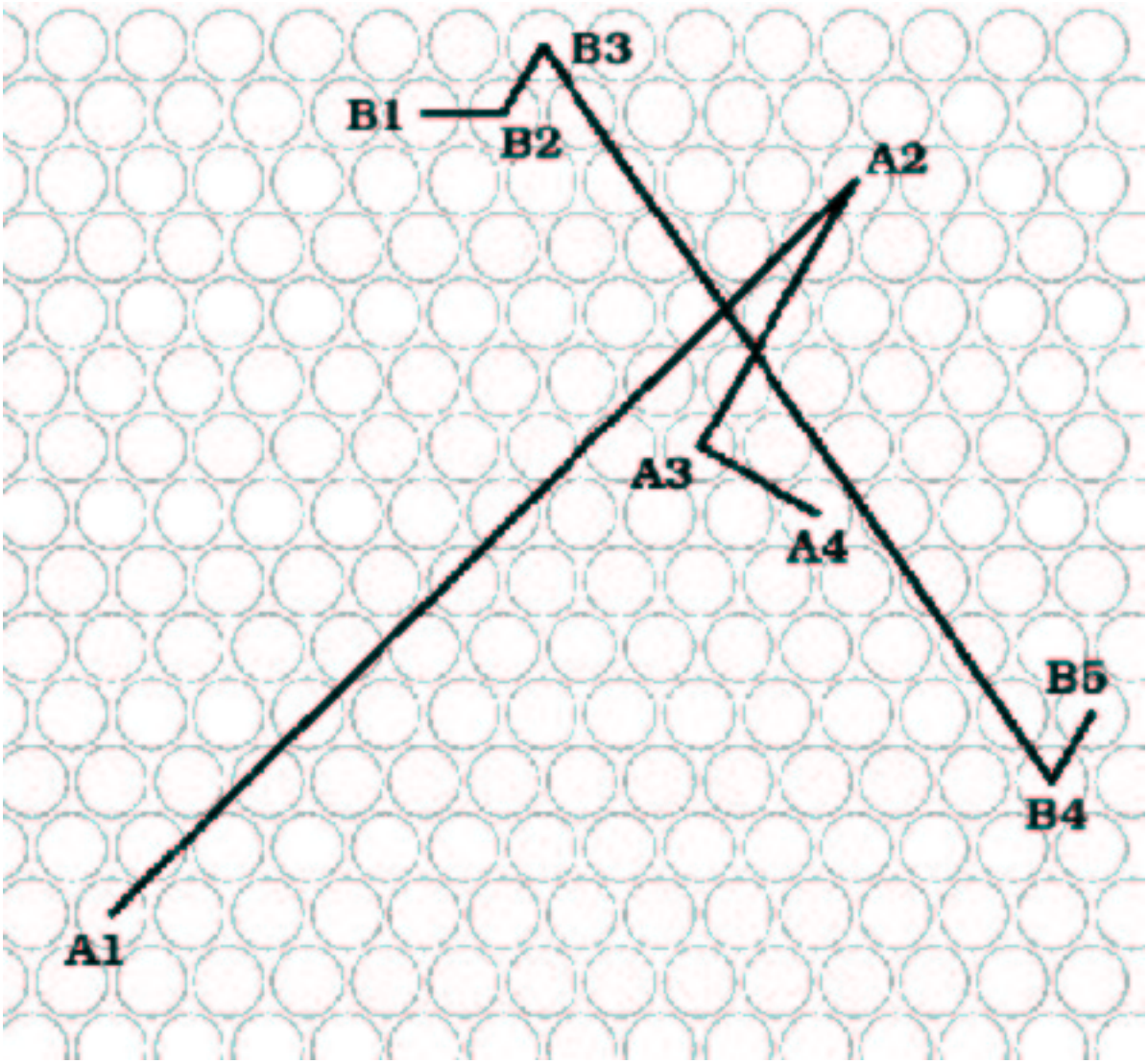

Figure 4: Two trajectories through a portion of Level 1's senor/motor map. Sequence A1-A4 occurred during a sharp right turn. Sequence B1-B5 occurred during a quick left turn. See Table 1 for the associated motor values.

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.

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.

5 Summary and Conclusions

In this paper we have argued that defining anchoring as a context-free problem may make the problem easier, but does nothing to advance a general theory. Rather, we embrace the full context-sensitive anchoring problem, although admitting that it is as hard as they get in artificial intelligence.

Anchoring is not simply a matter of finding and maintaining correspondences between *a priori* defined symbols and their perceptual correlates. The anchoring problem *is* the symbol grounding problem.

No doubt context-free anchoring is a useful task, just like context-free parsing has its uses. However, it may be impossible to build a general theory in such terms. To build a general theory capable of spanning across domains requires a serious rethinking of the problem. We propose such a radical departure. In our vision, anchoring is a process that surrounds a developing system. In this manner, the system can anchor objects, but also beliefs, actions, and behaviors as well.

Our model is in its initial stages. But with further research, we hope to better develop the idea of how developmental robotics can shed light on the anchoring problem.

Acknowledgments

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

References

[1] I. Bloch and A. Saffiotti. Some similarities between anchoring and pattern recognition concepts. In S. Coradeschi and A. Saffiotti, editors, *Papers from the 2001 AAAI Fall Symposium on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, number FS-01-01. AAAI Press, 2001.

[2] A. Chella, M. Frixione, and S. Gaglio. Interpreting symbols on conceptual spaces: The case of dynamic scenarios. In S. Coradeschi and A. Saffiotti, editors, *Papers from the 2001 AAAI Fall Symposium on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, number FS-01-01. AAAI Press, 2001.

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

[4] S. Coradeschi and A. Saffiotti. Perceptual anchoring of symbols for action. In *Proc. of the 17th IJCAI Conf.*, pages 407–412, Seattle, WA, 2001. AAAI Press.

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

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

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

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

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

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

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

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

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

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

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

[16] L. Meeden. *Towards planning: Incremental investigations into adaptive robot control.* PhD thesis, Indiana University, 1994.

[17] L. Meeden. 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, June 1996.

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

[19] F. Schonherr, M. Cistelecan, J. Hertzberg, and T. Christaller. Using behavior activation value histories for updating symbolic facts. In S. Coradeschi and A. Saffiotti, editors, *Papers from the 2001 AAAI Fall Symposium on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, number FS-01-01. AAAI Press, 2001.

[20] S. C. Shapiro. Artificial Intelligence. In A. Ralston, E. D. Reilly, and D. Hemmendinger, editors, *Encyclopedia of Computer Science*, pages 89–93. Grove's Dictionaries, Inc., NY, fourth edition, 2000.

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

[22] M. Witkowski, D. Randell, and M. Shanahan. Deriving fluents from sensor data for mobile robots. In S. Coradeschi and A. Saffiotti, editors, *Papers from the 2001 AAAI Fall Symposium on Anchoring Symbols to Sensor Data in Single and Multiple Robot Systems*, number FS-01-01. AAAI Press, 2001.

Author Biographies

Douglas Blank is currently working on designing robot behaviors using a developmental approach. He has previously worked on connectionist models of analogy. He is currently an Assistant Professor of Computer Science at Bryn Mawr College.

Deepak Kumar works on the connections between symbolic, top-down approaches to modeling behavior and the bottom-up learning based approaches. He has worked previously on rational agent architectures using belief-desire-intention models. He is currently an Associate Professor of Computer Science at Bryn Mawr College.

Lisa Meeden works on robot learning using connectionist and evolutionary computation approaches. Her interests include bottom-up, developmental approaches to robot learning. She is currently an Associate Professor of Computer Science at Swarthmore College.