A Framework for the Development of Robot Behavior

Roderic A. Grupen
Dept. of Computer Science
University of Massachusetts, Amherst
grupen@cs.umass.edu

Manfred Huber
Dept. of Computer Science and Engineering
University of Texas at Arlington
huber@cse.uta.edu

Abstract

Biological organisms display an astonishing capability to learn new skills and adapt to dynamic environments that far outperforms any computer or robot system. This paper presents an approach to robot skill acquisition that takes concepts from developmental theory to structure the learning problem and provides a mechanism to generate developmental schedules for a robot systems. The approach uses a developmental assembler to construct reusable and temporally extended actions in a sequence. All behavior is initially constructed from a set of innate control laws and events that delineate control decisions are derived from the pattern of (dis)equilibria on a working subset of sensorimotor policies. We show how this architecture can be used to accomplish sequential knowledge gathering and representation tasks and provide examples of developmental learning using a quadrupedal walking robot.

Introduction

Biological systems exhibit capabilities to acquire new skills and address novel tasks in complex environments that far surpass existing computer and robot technologies. We propose that part of this success is their use of innate structures and developmental mechanisms to guide learning while interacting with the environment. In particular, we propose that kinematic, dynamic, and neurological properties are exploited to simplify and structure learning. Developmental processes construct increasingly complex representations from a sequence of tractable learning tasks driven by a set of internal and environmental reinforcers. In this paper we present an approach to developmental organization in robotic systems that is aimed at providing similar learning and skill acquisition capabilities.

Behavior in biological systems is frequently learned in stages. By Piaget's account the sensorimotor stage in human infants, for example, lasts roughly 24 months (Piaget 1952). In the first four months, reflexive responses begin to organize into coherent motor strategies, and attentional mechanisms begin to emerge. From four to six months, primary circular reactions are practiced. Between six and eighteen months, these primary circular reactions lead to behavioral models of the world that apply to "classes" of interactions. A cornerstone to the theory describing such observations is the proposition that control knowledge can be represented in a manner that supports generalization. This paper explores if a commitment to such *figurative* schemata can lead to the acquisition of hierarchical control knowledge that can be used to similar advantage in the organization of robot behavior.

In this paper we present an approach to developmental organization in robotic systems that uses a developmental assembler to construct and re-use behavioral schemata (Lakoff 1984; Mandler 1992). Starting from an initial set of *figurative* schemata, corresponding loosely to innate reflexes, this approach acquires new schemata through interaction with the world under the guidance of a developmental strategy. We show how this approach can yield not only improvements in learning capabilities along a developmental trajectory but also leads to the acquisition of control knowledge and abstract knowledge representations grounded in behavioral skills. The operation of this approach and its potential benefits are illustrated with a sequence of experiments on a quadruped robot platform.

Structures for Learning and Development - Lessons from Developmental Theory

Investigating development in biology reveals a number of concepts that are important for its success and that, if captured in an appropriate computational framework, can also be used to construct robot control systems. In particular, studies in biology and psychology show how the structure of the organism and developmental mechanisms are used effectively to reduce the complexity of skill acquisition.

Considering, for example, an infant as an adaptive system in an open environment, the problem of establishing a monolithic control system is truly daunting. However, studies of development show that complex sensorimotor processes can temporarily compromise expressive power to reduce complexity. Managing this tradeoff effectively can lead to computational tractability in the short term and growth toward optimal behavior in the long term. We advocate the relatively optimistic position that traditions in robotics, control theory, AI, and learning are adequate computational accounts of some aspects of behavioral development and can thus form a basis for a developmental robot control systems.

Developmental Theory

Epigenetic developmental theory proposes that primitive reflexes, expressed as neuro-anatomical structures, are the ba-

sic building blocks of behavior. In this model, behavior is constructed from combinations of reflexes in response to reinforcement. Ontogenetic developmental theory suggests that coordinated behavior appears and subsequently disappears in order to serve a developmental function.

We contend that reflexes serve as an epigenetic computational basis and that some are short-lived and serve an ontogenetic knowledge formation role. For example, the stepping reflex is likely the antecedent of walking, but no simple reflexive precursor has been identified for reaching tasks which require multiple coordinated reflexes (Asymmetric Tonic Neck Reflex (ATNR), palmar grasp reflex, distal-curl reflex, Moro (clasp) reflex, startle, etc.). To understand developmental processes we need therefore to understand how knowledge and structure interact over time to acquire skills.

Biological observations and developmental theories suggest a number of essential components of developmental structure, namely the mechanism which modulates physical behavior, reflexes as the building blocks of behavior, maturational mechanisms that guide development, and a learning system that encapsulates and re-uses control knowledge.

Kinematic and Dynamic Structure In humans and other biological organisms, kinematic properties of the skeleton and the dynamics of the musculature strongly influence development. For example, Bizzi et. al. suggest that muscle dynamics influence the suitability of motor control (Bizzi, Chapple, & Hogan 1982).

Roboticists have similarly used kinematics and dynamics to fashion mechanisms with appropriate properties to facilitate behavior. For instance, Salisbury (Salisbury 1982) designed the Stanford/JPL robot hand to be kinematically isotropic when grasping a 1 inch sphere. Similarly, intrinsic dynamics has been used to design passive walking mechanisms. However, our ability to address tasks through properties of the mechanism remains ad hoc. As a consequence, a developmental robot control system has to appropriately model and control the physical structure of the mechanism.

Reflexes and Composability The Central Nervous System (CNS) is organized not in terms of anatomic segments but according to movement patterns (Aronson 1981). The basic form of packaged movement pattern is the reflex which can reside in the central and peripheral nervous system and range from involuntary responses to cortically mediated visual reflexes. These processes contribute to the organization of behavior at the most basic level by constituting a sensorimotor instruction set for the developing organism. The so-called developmental reflexes serve ontogenetic goals by guiding skill acquisition and are not elicited in normal adults. In addition to providing sensorimotor function, these reflexes also exercise the musculature and focus learning on conditions underlying developmental milestones.

The composition of reflexes can lead to more comprehensive behavior. For example, there is evidence that a discrete number of individual force fields are superimposed in the frog's leg/spine to yield continuously controllable leg position (Mussa-Ivaldi, Bizzi, & Giszter 1991). Moreover, certain motor patterns repeat in a regular pattern. Some, like walking, swimming, or flying, are the result of Central Pattern Generators (CPGs). Wolff suggests methods for composing oscillators in order to address novel initial conditions and contexts (Wolff 1991).

Similar ideas have also been used in the design of robotic systems. For example, Williamson has demonstrated the use of simple oscillators for periodic manipulation tasks (Williamson 1999). In addition, a range of control approaches have been developed which construct behavior from a set of basic actions, including a range of behavior-based robot control techniques (see (Arkin 1998) for an overview), and Burridge et. al.'s juggling robot (Burridge, Rizzi, & Koditschek 1999).

Developmental Schedules and Maturation

Behavior and knowledge acquisition in biological systems usually occurs in stages following a developmental schedule. The schedule is here enforced largely by maturational mechanisms that limit the set of available physical and sensory resources. As a consequence, behavior development tends to initially focus on a limited set of degrees of freedom and then extends to finally incorporate all the kinematic structures. For example, Berthier et. al. (Berthier, Clifton, McCall, & Robin 1999) published consistent findings of longitudinal studies of infants (6-30 weeks) during the onset of visually- and acoustically-guided reaching tasks. Initially, reaching movements appear to be focused primarily in the shoulder and torso. Large proximal degrees-of-freedom are engaged first while the intrinsic muscles of the forearm and hand stiffened via co-contraction.

Developmental Milestones The maturational processes described above lead to a developmental trajectory that occurs in a number of stages. Fiorentino presents a coarse description of the developmental process during the first year of an infant's life (Fiorentino 1981). A sequence of postural stability tasks is identified that starts with the infant acquiring the ability to control its head. In this task, information is assumed to be heavily weighted toward vestibular, proprioceptive, and (later) vision organs. Figure 1 illustrates an early sequence in which a child learns to raise its head off the floor. The infant uses optical- and labyrinthine-righting reflexes that develop over the first few weeks. These mechanisms, from the prone position, interact with the symmetric tonic neck reflex to develop a quadrupedal position. A proprioceptive reflex called the "body-on-head" reflex helps to rotate the trunk in response to a head angle. The infant thus acquires policies for rotating the trunk and head about the body axis to pan the head and eyes. All of this leads toward stabilizing the infant in sitting and later standing postures.

As shown throughout this section, biological systems rely considerably on reflexive structures that not only generate behavior, but shape the acquisition of control knowledge. In the following, we propose a computational mechanism to implement a similar form of staged knowledge formation and learning. This approach has already be applied successfully to a number of robot platforms, including multi-fingered hands, mobile robots, and walking platforms, to acquire control schemata. Some of these schemata and their potential place in an "infant-like" developmental sequence are indicated in capital names in Figure 1.

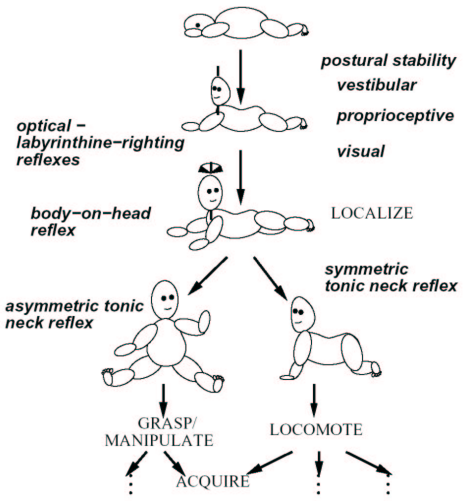

Figure 1: Superimposing a developmental sequence observed in human infants with schemata developed on robotic platforms. schemata are shown in capital letters.

A Model for Development in Robots - A Developmental Assembler

Figure 2 outlines a computational framework for robot systems that addresses learning and development and that incorporates some of the principles of structure discussed earlier. This framework constructs behavior from a compact set

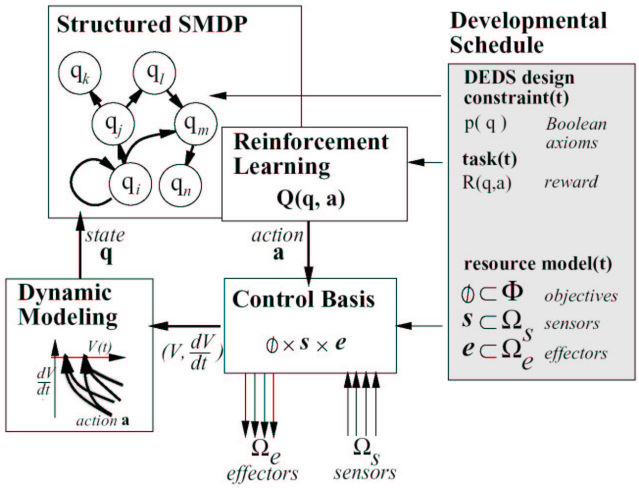

Figure 2: Native structure, learning, and behavior in an integrated developmental assembler.

of figurative schemata in the control basis under the guidance of a developmental schedule. During this process, the system also learns models of the interaction dynamics. Schemata are learned using a reinforcement learning component in a Semi-Markov Decision Process framework and can be re-used as additional elements in the control basis. These schemata, together with their associated dynamic models, serve as control knowledge for subsequent tasks. During learning, one dimension of development is viewed as a scheduling problem in which a strategy for engaging sensor, motor and computational resources is sought to satisfy a task. Each stage of this process is characterized by developmental parameters; the tasks, the participating control objectives, the sensor and effector resources allocated, and axioms that define legal combinations of behavior. The overall objective is to progress through a sequence of such designs to assemble new behaviors.

Action

The framework presented here is designed to learn a behavior hierarchy by composing more primitive actions. The Control Basis (Huber, MacDonald, & Grupen 1996; Coelho Jr. & Grupen 1997) in Figure 2 is designed to provide a combinatoric basis for control that supports the representation of declarative and procedural control knowledge. The most primitive actions, loosely corresponding to reflexes, are closed-loop control processes constructed by combining an artificial potential (or objective), $\phi \in \Phi$, sensory abstractions, $s \in \Omega_s$, and groups of effectors, $e \in \Omega_e$. The effect of an action plays out over time as the controller acts to optimize the action's control objective. Modeling primitive actions as controllers here implies that i) actions are asymptotically stable and generate trajectories toward locally optimal conditions with respect to the objective function, ii) controllers suppress local perturbations, iii) the dynamics of the controlled system provides useful discrete abstractions of the underlying continuous state space, and iv) time is metered by discrete observable events in the transient response of the controlled system rather than by an arbitrary clock.

Concurrent Control Composition To further increase the expressiveness, the Control Basis framework permits to activate multiple controllers concurrently. To obtain a predictable behavior the composition used here utilizes the hierarchical *subject-to* operator, $\lhd$, which, similar to the Moore-Penrose pseudoinverse (Yoshikawa 1990), limits actions of the subordinate controller to steps which do not counteract the objectives of the dominant controller. For example, a pair of controllers, $\phi_{sub} \lhd \phi_{sup}$, will descend the potential of the superior controller, ϕ_{sup}, and will superimpose only those action components from the subordinate controller, ϕ_{sub}, that do not increase the value of the superior potential. Examples of this approach to multi-objective control include posture optimization while reaching for a goal.

Learning, in this framework, is focused on finding combinations of controllers that create favorable dynamics. New policies can be found in terms of existing controllers.

State and System Modeling

The Dynamic Modeling component of Figure 2 is responsible for modeling the signature dynamics of the controlled process. For example, Figure 3 plots the potential, $\phi(t)$, against its rate of change for a grasp controller as it positions fingers on an unknown object (Coelho Jr. & Grupen 1997). As can be seen, the controller has multiple equilibria because there are many control contexts, i.e. many different objects and grasp solutions. When policy π_i is engaged, the pattern of membership in these empirical models, $A-F$,

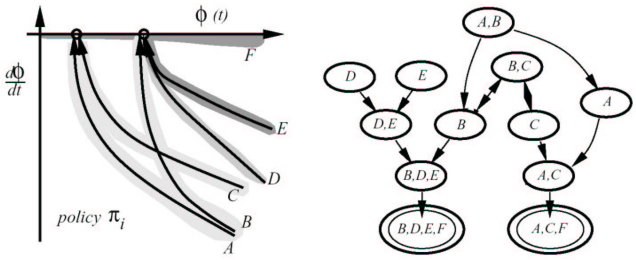

Figure 3: The pattern of membership in governing dynamic models serves to identify a discrete state for the policy.

changes over time in a manner that identifies the current control context. Model F is a special model signifying convergence, i.e. $\frac{d\phi}{dt} \approx 0$. Model F is the only model native to every controller - all other models are controller-specific.

This perspective has roots in methods like Hidden Markov Models (HMM) where categories are found by parsing a sequence of events. Likewise, Takens's theorem describes how patterns in nonlinear dynamical systems are related to hidden states. In our architecture, closed-loop controllers produce mechanical artifacts that distinguish control contexts. Assertions about the system's stability have been used to form the state space for such systems as in the attractors proposed by Huber et. al. (Huber & Grupen 1997) or the limit cycles proposed by Schaal et. al. (Schaal & Sternad 1998).

Controllers are distinguished by their sensory and motor resource allocations. The dynamic state of the controller can be expressed by a predicate vector $q_i \in Z^k$ that describes the status of ϕ_i by identifying the subset of empirical models M_{ij} , $j = 1...k$ that are consistent with the run-time observations. For instance, an element of q_i can represent convergence, implying that, for example, a particular stance of a walking machine is stable. We found that a small set of such models is sufficient to recover a wide variety of control contexts (Coelho 2001).

In general, an agent's predicate state q reflects the current status of several active controllers. We denote by $p(M_{ij}|\phi_i(q))$ the probability that model M_{ij} explains the observed time history when controller ϕ_i is engaged in state q. The system identification task is to learn $p(M_{ij}|\phi_i(q))$ for all predicate states q.

Reinforcement Learning

Reinforcement Learning (RL) is a natural paradigm for programming these systems since it does not require external supervision and learns from potentially delayed rewards (Barto, Bradtke, & Singh 1993). Here, RL is used to solve the temporal credit assignment problem for an optimal policy with respect to a given reinforcer.

Q-learning is used to compute the discounted sum of future rewards for each state-action pair, $Q(s, a)$. The control policy specifies which action, a, is to be selected from every state, s. Initially, actions are chosen randomly to explore the consequences of control decisions. Over time the rewards obtained are consolidated by updating the values of $Q(s, a)$ as the system transitions from state s_t to state s_{t+1}:

$$Q(s_t, a_t) = (1-\alpha)Q(s_t, a_t) + \alpha(r_{t+1} + \gamma \max_b Q(s_{t+1}, b))$$

where r_t is the reward received at time t, α is a learning rate, and γ is the discounting factor. As the learning process progresses, the control policy becomes increasingly focused on exploiting high-quality actions.

One of the major drawbacks of reinforcement learning methods is the large number of trials required to find a given policy. A second problem in exploration-based learning is the need to take random actions which can lead to catastrophic failures. The developmental mechanism described in the following section is designed to address these shortcomings and lead to high performance learning systems.

As policies are constructed, schemata that capture rewarding behavior are extracted and incorporated into the control basis. Subsequent policies may explore re-using these schemata which provide a temporal abstraction of the problem domain. This hierarchical approach is formalized here as a Semi-Markov Decision Process (SMDP).

Developmental Schedule

The set of primitive actions from a given control basis, $\Phi \times 2^{\Omega_s} \times 2^{\Omega_e}$, is quite large. This is good from the perspective of expressive power but bad for computational complexity. Therefore, aspects of developmental structure have been implemented to bias exploration toward computationally tractable subsets of the action and state sets in order to accumulate critical control knowledge sequentially.

The resource model expresses constraints on the sensors, effectors, and potential functions/policies that may be considered when generating actions. As such it models the effect of the maturational mechanisms discussed previously. To incorporate the "maturational" constraints into the control system, the approach presented here uses the Discrete Event Dynamic Systems (DEDS) formalism (Sobh *et al.* 1994) to constrain the range of legal interactions to those that i) satisfy real-time computing constraints, ii) guarantee safety specifications, and iii) are consistent with kinematic and dynamic limitations. In this formalism the state of the system is assumed to evolve with the occurrence of discrete events and a supervisor takes the form of a nondeterministic finite state automaton in which states are patterns of membership in dynamic models and transitions represent concurrent control situations. Logical conditions on the predicate vector influence the range of control options.

Example: Learning Quadrupedal Gaits

To demonstrate the presented control approach and to illustrate its benefits, this section presents a sequence of experiments using the walking platform"Thing" (Figure 4).

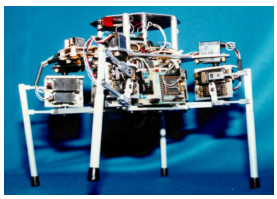

Figure 4: *Thing*, a 12 degree of freedom quadruped designed to learn walking gaits using the developmental assembler.

Thing is a small, 12 degree of freedom quadruped that was "born" with three primitive control objectives, $\phi \in \Phi$, in the control basis; namely, force, position, and kinematic conditioning objectives. Controllers are constructed by associating objectives with resources and concurrent controllers are constructed using *subject-to* compositions (Huber & Grupen 1997). A developmental sequence was implemented in which Thing learns simple policies and then uses them as abstract actions in a behavioral hierarchy.

Developmental Constraints

The developmental sequence was implemented in the developmental assembler as time-varying constraints which represent "maturational" processes and domain requirements.

"Maturational" Resource Constraints The simplest gait in Thing's repertoire achieves reward by accumulating a heading change. The resource model for this ROTATE schema considers recruiting three-legged tripod stances into controllers. Objective type ϕ_1 is a Zero Moment Point (ZMP) controller parametrized by the sensors and effectors with which it is implemented. Sensors designate the position of three foot placements and one of these three legs will be controlled to minimized the net moment around the platform's center of mass.

$$\phi_1 : ZMP\ controller \left|{}^{input\ tripod}_{active\ leg}\right.$$

There are four unique tripod stances for a quadruped, each of which can elect to apply ZMP control to one of the legs. This recruitment model yields 12 unique controllers.

Further, the resource model includes a single kinematic conditioning controller that looks at the configuration of all 4 legs and executes movements to rotate the robot's body while leaving the foot placement fixed. Objective ϕ_2 is thus a kinematic conditioning (KC) controller that optimizes the condition of the legs by rotating the robot's heading, φ.

$$\phi_2 : KC\ controller \left|{}^{input\ tetrapod}_{heading}\right.$$

The resource model, therefore, provides a first layer of developmental structure, organizing a set of 13 unique primitive controllers for the rotate task.

Quasistatic Constraints If there are k models of control dynamics for each controller, then there can be at most 2^{13*k} unique membership patterns. However, many of these states are unreachable. Moreover, Bernstein suggested that much can be learned in a quasistatically stable approximation of the unconstrained state space. In this case, we require that the robot always maintains at least one stable stance. If we define the convergence condition (model F in Figure 3) to be satisfied when the tripod is stable then no additional models are necessary. Moreover, for each ZMP controller, the stability assertion is independent of which leg is assigned as the effector. Therefore, the status of the 12 ZMP controllers can be captured in 4 binary predicates, p_i, each indicating the convergence of a ZMP controller to a stable equilibrium. With the kinematic conditioning control, this leads to 5 bits of state information for the quasistatic condition.

$$state = \begin{bmatrix} p_0\left(\phi_1\left|{}^{012}_{*}\right.\right)\ p_1\left(\phi_1\left|{}^{023}_{*}\right.\right)\ p_2\left(\phi_1\left|{}^{123}_{*}\right.\right) \\ p_3\left(\phi_1\left|{}^{013}_{*}\right.\right)\ p_4\left(\phi_2\left|{}^{0123}_{\varphi}\right.\right) \end{bmatrix}$$

In the experiments reported, we allowed up to three objectives to be addressed simultaneously, leading to 1885 actions. To guarantee that the robot will not fall, it is necessary that at least one ZMP controller is near equilibrium at all times. This specification is expressed as a logical disjunction, $p_0 \vee p_1 \vee p_2 \vee p_3$. This structural axiom is used as a filter during exploration, reducing the average number of legal actions to just 157.

Compiling Control Knowledge

Using the developmental mechanism described here, a first experiment was used to learn two basic walking schemata, namely a ROTATE schema for rotation in place and a STEP schema corresponding to a simple stepping pattern.

The ROTATE Schema In the first learning task, the developmental constraints introduced in the previous sections were applied and a reward structure was provided to reward control sequences that accumulate angular rotations:

$$r = \Delta\varphi = \varphi_k - \varphi_{k-1}$$

Index k here designates consecutive convergence events and φ_k is the heading following convergence of action a_k . The rotation gait illustrated in Figure 6, where the bit vectors in the states indicate the values of the five convergence predicates, was acquired reliably in about 11 minutes, on-line, in a single trial. Figure 5 shows the average learning curve over 10 learning trials for the ROTATE Schema.

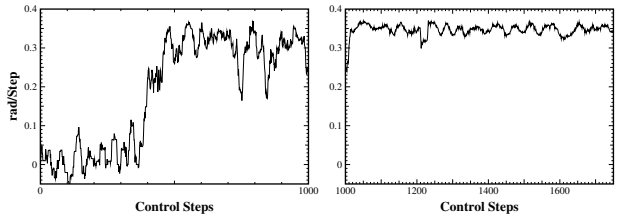

Figure 5: Performance of the ROTATE gait during learning (left) and during execution (right).

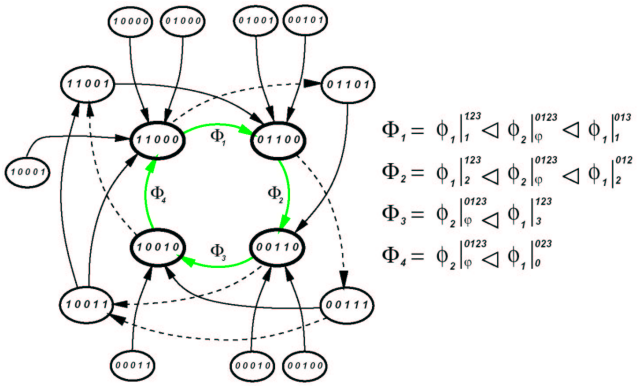

Figure 6: The ROTATE policy with contingencies for a variety of run-time contexts. The central cycle has transition probabilities greater that 95%.

The STEP Schema After the ROTATE gait was learned, the resource model was elaborated to support resource engagements that could translate the robot's center of mass.

The new design contains 10 state predicates and 175 actions on average per state. A reward signal was provided that was proportional to the forward motion of the robot, resulting in a STEP schema that represents a simple forward stepping pattern. Figure 7 shows the corresponding learning curve.

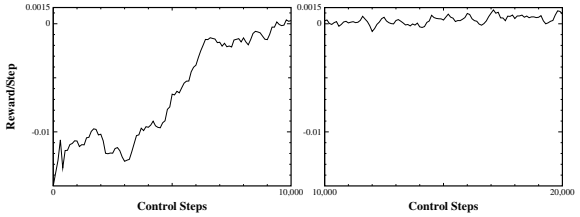

Figure 7: Performance of the STEP gait during learning (left) and during execution (right).

Re-Using Schemata - The TRANSLATE Schema

Once the ROTATE and STEP schemata are captured, they can be included in the control basis, making them available for re-use as temporally extended actions.

To evaluate the relative impact of these behavioral abstractions and the associated control knowledge in the context of a new task, an additional series of experiments was performed. For these experiments, the resource model was first further enriched to include the position controller, yielding 12 state predicates and an average of 231 actions per state. Then a reward signal proportional to the reduction in distance to the goal was provided.

$$r_k = d_{k-1} - d_k$$

where d_k is the robot's distance from the goal after event k.

Using this setup, a baseline experiment was performed in which none of the schemata was available. Subsequently, two more experiments were performed, the first using only the ROTATE schema and the second using both the ROTATE and the STEP schemata. Figure 8 compares the performance of the three TRANSLATE designs.

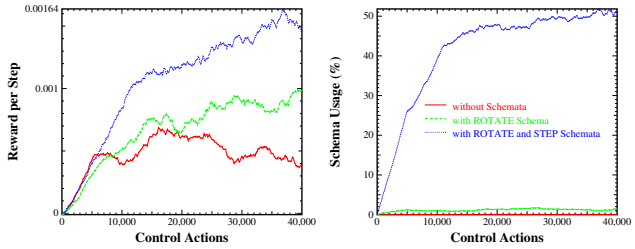

Figure 8: Performance of the TRANSLATE schema. The left panel compares learning performance without any schema, with the ROTATE schema, and with the ROTATE and the STEP schemata. The right panel shows the percentage of executed steps from a schema.

Each 10,000 control actions here required about 2 hours of run-time in a single trial. After roughly 2 hours, the performance of both larger problem designs exceed the more economical base system. Moreover, it can be seen that the system with both schemata easily outperforms the one with only the ROTATE schema (although the gaits learned in both cases reach the same asymptotic performance after approximately 110,000 steps).

The graph on the right of Figure 8 shows the percentage of times that an action executed in the TRANSLATE schema was either from the ROTATE or the STEP schema. These curves show that in the case of the TRANSLATE gait with only the ROTATE schema the final gait uses the schema only approximately 2% of the time. However, even this limited use permits the system to successfully orient itself with respect to the goal and thus indirectly focuses the learning process on acquiring a walking pattern without having to worry about alignment. In the case of the TRANSLATE gait with both, the ROTATE and the STEP schemata, schema usage increases to more than 50%, indicating a significant re-use of the basic stepping pattern encoded in the STEP schema. The learning process can here focus on the acquisition of transition gaits and on the improvement of the basic step pattern into a robust TRANSLATE gait.

Hierarchy - The MAZE-SOLVE Schema

Once schemata for rotating and translating are in place, navigating in a cluttered environment can be formulated as a policy for deciding when to engage these temporally extended actions, one at a time in response to observed obstacles. We demonstrated that Thing can find a path from point A to point B with no prior knowledge of the intervening obstacles using a forward-looking IR proximity detector to observe obstacles enroute and map them into its configuration space. The locomotion plan follows a streamline in a harmonic function path controller by selecting one of two temporally extended actions (ROTATE, or TRANSLATE) in a 4 state finite state automaton. The state is derived from a 2 bit "interaction-based" state descriptor. One bit describes the convergence status of the rotate controller, and the other describes the convergence status of the translate controller. Figure 9 shows an example run of the robot.

Conclusions and Future Work

This paper presents an approach to robot control that utilizes developmental mechanisms to automatically generate control knowledge in terms of behavioral schemata that can be re-used in subsequent tasks. Taking guidance from developmental theory in biological systems, this approach builds behavior from a set of closed-loop controllers, corresponding loosely to reflexes in biological systems. Skill learning is guided within the developmental assembler using a developmental schedule that imposes constraints in a DEDS framework to simulate the effects of maturational mechanisms in biological systems. In particular, it imposes time-varying constraints on the set of sensor and effector resources that can be recruited by the control elements.

A sequence of experiments was performed that illustrated the potential of the proposed approach in the context of a developmental trajectory for a quadruped robot. These experiments clearly demonstrate the benefit of incorporating learned control knowledge in the form of behavioral schemata and illustrates the potential for reductions in state space complexity once competent schemata are learned. We

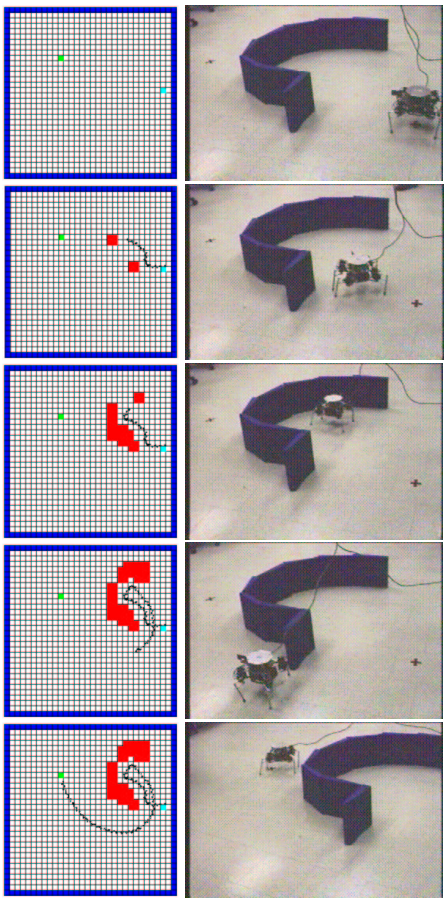

Figure 9: The MAZE-SOLVE schema has 3 actions - ROTATE, TRANSLATE, and a harmonic function path controller. The MAZE-SOLVE schema descends the harmonic potential using TRANSLATE subject to ROTATE and solves all mazes for which there exists a path at the resolution of the configuration space (illustrated on the left).

are currently in the process of investigating techniques to further generalize learned schemata into figurative forms which can be instantiated with different resource assignments. Such capabilities to predict other instances of a schema, in turn, could lead to significantly more advanced representational abstractions and potentially to metaphorical extensions whereby schemata are extended to other physical examples of that phenomenon.

Acknowledgments

This work was supported in part by NSF CDA 9703217 and ITR-0121297, DARPA-MARS DABT63-99-1-0004, and NASA NAG9-1445#1.

References

Arkin, R. 1998. Behavior-Based Robotics. MITY Press.

Aronson, A. 1981. Clinical Examinations in Neurology. W.B. Saunders Co., Philadelphia, PA.

Barto, A. G.; Bradtke, S. J.; and Singh, S. P. 1993. Learning to act using real-time dynamic programming. Technical Report 93-02, University of Massachusetts, Amherst, MA.

Bizzi, E., Chapple, W., and Hogan, N. 1982. Mechanical properties of muscles: Implications for motor control. *Trends in Neuroscience* 5:395–398.

Berthier, N. E., Clifton, R. E., McCall, D. D., and Robin, D. J.. 1999. Proximodistal structure of early reaching in human infants. *Experimental Brain Research* 127:259–269.

Burridge, R. R., Rizzi, A. A., and Koditschek, D. E. 1999. Sequential composition of dynamically dexterous robot behaviors. *Robotics Research* 18(6):534–555.

Coelho Jr., J. A., and Grupen, R. A. 1997. A control basis for learning multifingered grasps. *J. Robotic Sys.* 14(7):545–557.

Coelho, J. A. Jr. 2001. Multifingered Grasping: Haptic Reflexes and Control Context. PhD thesis, University of Massachusetts, Amherst, MA.

Fiorentino, M. R. 1981. A Basis for Sensorimotor Development - Normal and Abnormal. Charles C. Thomas.

Huber, M., and Grupen, R. A. 1997. A feedback control structure for on-line learning tasks. *Robotics and Autonomous Systems* 22(3-4):303–315.

Huber, M.; MacDonald, W. S.; and Grupen, R. A. 1996. A control basis for multilegged walking. In *Proc. IEEE Int. Conf. Robot. Automat.*, 2988–2993. Minneapolis, MN.

Lakoff, G. 1994. Women, Fire, and Dangerous Things. University of Chicago Press.

Mandler, J. M. 1992. How to build a baby: II. conceptual primitives. *Psychological Review* 99(4):587–604.

Mussa-Ivaldi, F., Bizzi, E., and Giszter, S. 1991. Transforming plans into action by tuning passive behavior: A field approximation approach. In *Proc. Int. Symp. on Intelligent Control* pp. 101–109. Arlington, VA.

Piaget, J. 1952. The Origins of Intelligence in Childhood. International Universities Press.

Ravindran, B. and Barto A. 2001. Symmetries and model minimization in markov decision processes. CMPSCI Technical Report 01-43, Computer Science Department, University of Massachusetts, Amherst, MA.

Salisbury, J. K. 1982. Kinematic and Force Analysis of Articulated Hands. PhD thesis, Stanford University.

Sobh, M.; Owen, J.; Valvanis, K.; and Gracani, D. 1994. A subject-indexed bibliography of discrete event dynamic systems. *Robotics & Automation Magazine* 1(2):14–20.

Schaal, S. and Sternad, D. 1998. Programmable pattern generators. In *Proceedings of the International Conference on Computational Intelligence in Neuroscience* pp. 48–51.

Vukobratovic, M. and Borovac, B. 2004. Zero-moment point - thirty five years of its life. *Journal of Humanoid Robotics* 1(1):157–173.

Williamson, M. M. 1999. Neural control of rhythmic arm movements. *Neural Networks* 11(7-8):1379–1394.

Wolff, P. 1991. Endogenous motor rythms in young infants. The Development of Timing Control and Temporal Organization in Coordinated Action. Elsevier Science.

Yoshikawa, T. 1990. *Foundations of Robotics : Analysis and Control*. Cambridge, MA: MIT Press.