Improving Grasp Skills Using Schema Structured Learning

Robert Platt
Dexterous Robotics Laboratory
Johnson Space Center, NASA
robert.platt-1@nasa.gov

Roderic A. Grupen
Laboratory for Perceptual Robotics
Department of Computer Science
University of Massachusetts, Amherst
grupen@cs.umass.edu

Andrew H. Fagg
Symbiotic Computing Lab
School of Computer Science
University of Oklahoma
fagg@ou.edu

***Abstract*— In the control-based approach to robotics, complex behavior is created by sequencing and combining control primitives. While it is desirable for the robot to autonomously learn the correct control sequence, searching through the large number of potential solutions can be time consuming. This paper constrains this search to variations of a generalized solution encoded in a framework known as an *action schema*. A new algorithm, *schema structured learning*, is proposed that repeatedly executes variations of the generalized solution in search of instantiations that satisfy action schema objectives. This approach is tested in a grasping task where Dexter, the UMass humanoid robot, learns which reaching and grasping controllers maximize the probability of grasp success.**

I. Introduction

In contrast to the sense-think-act paradigm, control-based and behavior-based approaches to robotics realize desired behavior by sequencing and combining primitive controllers or behaviors. These approaches depend on a higher-level decision-making mechanism that selects the correct primitives to execute. One way to automatically learn the correct sequence of primitives is to encode the problem as a Markov Decision Process, and solve it using Reinforcement Learning [1], [2]. However, in the absence of an *a priori* model of controller performance, this approach requires the robot to explore the effects of executing every action in every state. Although the system designer can manually constrain the potential action choices [1], the need to explore a large number of actions can slow down learning. The current paper addresses this problem by encoding a generalized solution as an *action schema*. Learning speed is increased by constraining the system to consider only variations of this generalized solution. A new algorithm called *schema structured learning* discovers which instantiations of the action schema are appropriate in different problem contexts. This paper explores this approach in the context or reaching and grasping. This paper is an expansion of our earlier work reported in [3]. The current paper better defines optimality in the context of the action schema and proposes a sample-based version of the algorithm.

Reinforcement learning (RL) is an attractive approach to robot learning because it allows the robot to autonomously learn to solve different problems using a single underlying set of control actions. For example, Huber and Grupen use RL to learn autonomously a rotation gait for a quadrupedal robot [1]. Martinson, Stoytchev, and Arkin use RL to solve a tank intercept problem using a small discrete set of states and control actions [4]. Rosenstein and Barto use a version of RL to learn to parameterize controllers that participate in a robot weightlifting task [5]. Unfortunately, the solutions to many practical robot problems are far too complex to learn autonomously starting from a low-level and general set of control primitives. In each of the above approaches, this complexity is managed by manually supplying significant structure to simplify learning. Huber prunes the number of potential actions that the robot must explore. Martinson, *et al.* handcraft specific states and actions for the task. Rosenstein and Barto manually specify a control structure and autonomously learn parameters that optimize its performance. The current paper acknowledges that such structure may be necessary in practical robotics problems and proposes a new framework that is inspired by the Piagetian notion of a schema for specifying this structure.

Piaget loosely defines a schema to be a mental representation of an action or perception [6]. Through the process of *assimilation*, the child adapts an existing schema to incorporate new experiences, encoding these experiences as variations on the same general structure. Piaget proposes that infants posses a basic tendency to exercise existing schemas, especially when those structures are not "well formed."

Two important previous approaches to incorporating the notion of a schema into a computational framework are that of Arbib and Drescher [7], [8]. Arbib's *schema theory* proposes two major types of schemas: the perceptual schema and the motor schema [7]. A perceptual schema is a process that responds to only a specific, task-relevant concept. A motor schema is a generalized program that describes how to accomplish a task. Schema theory proposes that a large number of perceptual schemas and motor schemas can interact in the context of a *coordinated control program*, thereby generating intelligent behavior [7]. Arkin applies some of these ideas to behavior-based robotics [2]. Gary Drescher also develops a schema-based approach to intelligent behavior based on Piagetian ideas [8]. Learning starts with a few schemas and primitive concepts that represent basic motor activities and perceptions. By executing schemas, the system discovers new concepts and proposes new schemas for interacting with these

concepts.

This paper proposes a new approach to robot learning based on a generalized representation of robot behavior known as the action schema. The action schema may be *instantiated* by specific implementations of the generalized behavior. An instantiation is considered to *succeed* or *fail* depending upon whether it results in desired state transitions specified by the action schema. A new on-line learning algorithm, schema structured learning, is proposed that explores different instantiations and discovers through a process of trial-and-error which instantiations are most likely to succeed. This paper explores the action schema approach in the context of robotic reaching and grasping. Schema structured learning discovers how to select appropriate reach and grasp control actions based on coarse visual information including object location, orientation, eccentricity, and length. A series of experiments are reported where Dexter, the UMass bimanual humanoid robot, attempts to grasp objects using various different reach and grasp primitives. The robot learns to select reach and grasp primitives that optimize the probability of a successful grasp.

Section II gives a brief overview of the *control basis* approach to robot behavior used in this paper and describes controllers used for localizing, reaching, and grasping. Sections III and IV describe the action schema framework, define the notion of the optimal policy instantiation, and give an algorithm, schema structured learning, for autonomously discovering these optimal instantiations. Finally, Section V presents experimental results.

II. Control-Based Reaching and Grasping

This paper's development of schema structured learning uses Huber and Grupen's *control basis* framework [1]. This framework systematically defines a set of control primitives and provides a robust and general way of representing system state. This section describes the control basis framework and details controllers that are used for localizing, reaching, and grasping.

The control basis can systematically specify an arbitrary closed-loop controller by matching an *artificial potential function* with a *sensor transform* and *effector transform* [1]. The artificial potential specifies controller objectives, the effector transform specifies what degrees of freedom the controller uses, and the sensor transform implements the controller feedback loop. For example, a REACH controller is defined by a REACH artificial potential, a sensor transform that specifies the goal configuration of the end-effector, and an effector transform that specifies what degrees of freedom are used to accomplish the task.

In general, the control basis realizes a complete controller by selecting one artificial potential from a set $\Phi = \{\phi_1, \phi_2, \ldots\}$, one sensor transform from a set $\Sigma = \{\sigma_1, \sigma_2, \ldots\}$, and one effector transform from a set $\Upsilon = \{\tau_1, \tau_2, \ldots\}$. Given Φ, Σ, and Υ, the set of controllers that may be generated is $\Pi \subseteq \Phi \times \Sigma \times \Upsilon$. When specifying a fully-instantiated controller, the notation $\phi_i|_\tau^\sigma$ denotes the controller constructed by parameterizing potential function ϕ_i with sensor transform σ and effector transform τ.

The control basis framework also allows composite controllers to be constructed that execute multiple constituent controllers concurrently. Each constituent controller is assigned a priority, and controllers with lower priority are executed in the nullspace of controllers with higher priority. Composite controllers are denoted, $\phi_b|_\tau^\sigma \triangleleft \phi_a|_\tau^\sigma$, where $\phi_b|_\tau^\sigma$ is said to execute "subject-to" (i.e. in the nullspace of) $\phi_a|_\tau^\sigma$.

System state is measured in terms of controller dynamics. At any point in time, the instantaneous error and the instantaneous gradient of error can be evaluated. Although the more general system dynamics can be treated [9], in this paper we will consider only controller convergence to establish system state. For example, the state of having grasped an object with some effector is represented by the convergence status of a grasp controller parameterized by that effector transform.

This paper's ideas are illustrated in the context of a LOCALIZE-REACH-GRASP action schema that implements generalized grasping behavior. The LOCALIZE controller in LOCALIZE-REACH-GRASP, $\phi_l|_\tau^\sigma$, segments the object and characterizes the resulting blob in terms of its three-dimensional Cartesian position, orientation, length, and eccentricity.

In this paper, LOCALIZE-REACH-GRASP uses two REACH control primitives: a reach-to-position primitive, $\phi_{rp}|_\tau^{\sigma(x)}$, and a reach-to-orientation primitive, $\phi_{ro}|_\tau^{\sigma(\theta)}$. When $\phi_{rp}|_\tau^{\sigma(x)}$ executes on its own, the robot reaches to an offset of x along the object major axis without regard for orientation. When reach-to-orientation executes subject to reach-to-position, $\phi_{ro}|_\tau^{\sigma(\theta)} \triangleleft \phi_{rp}|_\tau^{\sigma(x)}$, then the robot attempts to achieve a desired orientation while reaching to a position. Orientation is measured with respect to the lines running from the contact set centroid through each contact. $\phi_{ro}|_\tau^{\sigma(\theta)}$ orients the manipulator so that the average angle between these lines and the object major axis is θ.

GRASP controllers displace contacts toward good grasp configurations using feedback control [10], [11]. This approach uses tactile feedback to calculate an error gradient and displace grasp contacts on the object surface without using a geometric object model. After making light contact with the object using sensitive tactile load cells, the controller displaces contacts toward minima in the grasp error function using discrete probes [10] or a continuous sliding motion [12]. This paper uses two GRASP controllers: $\phi_g|_{\tau_{123}}^{\sigma_{123}}$ and $\phi_g|_{\tau_{12}}^{\sigma_{12}}$. $\phi_g|_{\tau_{123}}^{\sigma_{123}}$ uses three physical contacts to synthesize a grasp, while $\phi_g|_{\tau_{12}}^{\sigma_{12}}$ combines two physical contacts into a *virtual finger* [13] that is considered to apply a single force that opposes a third physical contact.

III. The Action Schema

Although the control basis gives the robot access to a general-purpose set of control primitives, the large number of action choices can make autonomous trial-and-error learning computationally complex. The action schema framework is a systematic way of structuring a trial-and-error robot learning

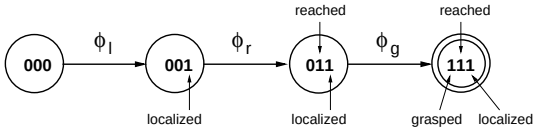


Fig. 1. The localize-reach-grasp action schema. The circles with binary numbers in them represent abstract states. The arrows represent abstract actions and transitions.

problem by encoding a generalized robot behavior that constrains the number of potential solutions that learning must consider. The generalized behavior is represented by a policy defined over an abstract state and action space. A one-to-many mapping is defined between the abstract state and action space and an underlying state and action space. The underlying space is assumed to represent system state and action with the finest granularity available to the robot. This allows the action schema's abstract policy to be translated into a number of *policy instantiations* that define a set of potential solutions. The action schema also specifies an abstract transition function that defines desired transition behavior in the underlying space. The objective of schema structured learning is to discover which instantiations of an action schema's abstract policy are most likely to result in transitions that are consistent with the abstract transition function.

Let $S' \times A'$ be the abstract state-action space defined by the action schema, and let $S \times A$ be the underlying state-action space that can represent possible robot behavior. The abstract policy is a mapping from abstract states to abstract actions,

$$\pi' : S' \rightarrow A'. \tag{1}$$

This function deterministically specifies which abstract action the system should take in any given abstract state. When the control basis is used with the action schema, abstract states and actions can be defined in terms of artificial potentials. For example, Figure 1 illustrates the LOCALIZE-REACH-GRASP action schema. The three abstract actions, ϕ_l (LOCALIZE), ϕ_{rp} (REACH), and ϕ_g (GRASP), correspond to controllers stripped of their parameterizations (leaving only artificial potentials). The circles illustrate the four abstract states, also derived from artificial potentials. The abstract policy, π', defines which abstract actions are to be taken from abstract states:

$$\begin{aligned} \pi'(000) &= \phi_l \\ \pi'(001) &= \phi_{rp} \\ \pi'(011) &= \phi_g. \end{aligned} \tag{2}$$

For example, if the robot is in abstract state (000), then π' executes abstract action ϕ_l.

The abstract policy is mapped to policy instantiations that implement the same qualitative type of behavior. This policy mapping is derived from state and action mappings that group together similar states and actions as follows. Let $f : S \rightarrow S'$ and $g : A \rightarrow A'$ be state and action functions that uniquely assign each underlying state and action to an abstract state and action. The inverses of these functions are defined to be $f^{-1}(s') = \{s \in S | f(s) = s'\}$ and $g^{-1}(a') = \{a \in A | g(a) = a'\}$. g^{-1} maps each abstract action to an equivalence class of actions that perform the same function in different ways. Similarly, f^{-1} maps each abstract state to an equivalence class of states.

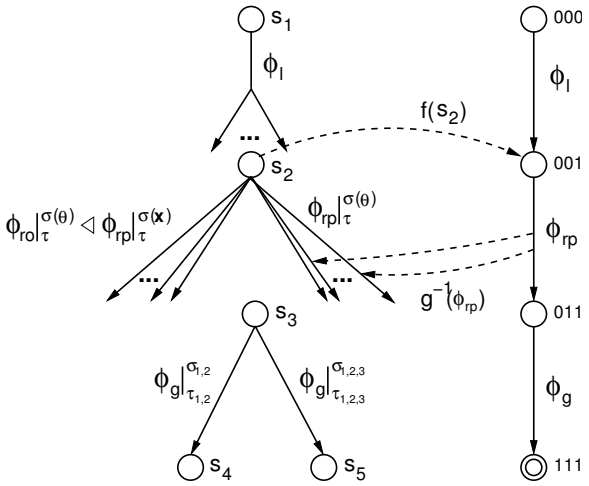


Fig. 2. Possible instantiations of the LOCALIZE-REACH-GRASP action schema. The inverse action mapping, g^{-1}, projects the abstract REACH action onto the set of possible REACH actions.

When the control basis is used, the abstract actions are represented by artificial potentials that can be instantiated by different choices for sensor and effector transform. In addition, when composite controllers are possible, arbitrary subordinate control primitives may be added to create even more instantiations. Controllers related to an artificial potential (the abstract action) in this way are guaranteed to share certain characteristic functionality. For example, in LOCALIZE-REACH-GRASP, the three abstract actions, ϕ_l, ϕ_{rp}, and ϕ_g map to underlying controllers as follows:

$$\begin{aligned} g^{-1}(\phi_l) &= \{\phi_l|_\tau^\sigma\}, \\ g^{-1}(\phi_{rp}) &= \{\phi_{rp}|_\tau^{\sigma(x)} | x \in [0,1]\} \cup \\ &\quad \{\phi_{ro}|_\tau^{\sigma(\theta)} \triangleleft \phi_{rp}|_\tau^{\sigma(x)} | x \in [0,1], \theta \in [0, \frac{\pi}{2}]\}, \\ g^{-1}(\phi_g) &= \{\phi_g|_{\tau_{123}}^{\sigma_{123}}, \phi_g|_{\tau_{12}}^{\sigma_{12}}\}. \end{aligned} \tag{3}$$

In this example, the abstract action, ϕ_g, can be instantiated by $\phi_g|_{\tau_{123}}^{\sigma_{123}}$ or $\phi_g|_{\tau_{12}}^{\sigma_{12}}$ (either a three-fingered grasp or a two-fingered grasp).

The state and action mappings, f and g, allow the abstract policy to be translated into a set of potential policy instantiations. This is accomplished by determining the set of actions that are consistent with the abstract policy in a given state. Assume that the system is in abstract state, s'_t. The abstract action specified by $\pi'(s'_t)$ can be projected onto a set of equivalent underlying actions using the inverse action mapping, $g^{-1}(\pi'(s'_t))$. Therefore, given the state and action mapping, the abstract policy, π', can be mapped onto any policy, π, such that,

$$\forall s_t \in S, \;\; \pi(s_t) \in B(s_t), \tag{4}$$

where

$$B(s_t) = g^{-1}(\pi'(f(s_t))). \quad (5)$$

These policies are called *policy instantiations* of the abstract policy.

This is illustrated in Figure 2. Suppose that the robot is in state $s_2 \in S$. The state mapping, f, projects this state onto $(001) \in S'$. Since $\pi'(001) = \phi_{rp}$, the abstract policy takes ϕ_{rp} from (001). Finally, the inverse action mapping, g^{-1}, projects this onto the reach choices, $\{\phi_{rp}|_\tau^{\sigma(x)} | x \in [0,1]\} \cup \{\phi_{ro}|_\tau^{\sigma(\theta)} \triangleleft \phi_{rp}|_\tau^{\sigma(x)} | x \in [0,1], \theta \in [0, \frac{\pi}{2}]\}$.

The action schema also defines transition constraints that specify how the robot is to behave while executing a policy instantiation. The desired behavior of the action schema is deterministically characterized by the abstract transition function,

$$T' : S' \times A' \rightarrow S'. \quad (6)$$

This specifies how the system must transition in response to executing the action. When taken from state $s_t \in S$, action $a \in A$ must deliver the system to

$$s_{t+1} \in f^{-1}(T'(f(s_t), g(a))). \quad (7)$$

As long as action $a \in A$ causes the robot to transition to one of these next states, the action is said to *succeed*. If the action causes a different transition, then the action *fails*. The goal of schema structured learning is to discover which policy instantiation maximizes the probability of meeting these transition constraints.

For example, in the case of the LOCALIZE-REACH-GRASP action schema, the abstract transition function is defined to be:

$$\begin{aligned} T'(000, \phi_l) &= 001 \quad (8) \\ T'(001, \phi_{rp}) &= 011 \\ T'(011, \phi_g) &= 111. \end{aligned}$$

Suppose that the robot is in state $s_2 \in S$, and executes $\phi_{rp}|_\tau^{\sigma(0.5)}$. If the system does not transition to a state, $s_{t+1} \in S$, that maps to $(011) \in S'$, $f(s_{t+1}) = (011)$, then $\phi_{rp}|_\tau^{\sigma(0.5)}$ fails.

In schema structured learning, the robot continues to execute actions in accordance with the abstract policy until an action fails or an absorbing state is reached. In the LOCALIZE-REACH-GRASP action schema, (111) is an absorbing state.

Bringing these pieces together, an action schema is a represented as a tuple,

$$\mathcal{S} = \langle S', A', \pi', T' \rangle, \quad (9)$$

where S' is the abstract state set, A' is the abstract action set, π' defines the abstract policy, and T' defines the abstract transition function. When defining an action schema, we will require that the path implicitly specified by π' and T' does not contain cycles.

IV. Optimal Policy Instantiations

The abstract policy encoded by the action schema maps to many different policy instantiations. The goal of schema structured learning is to discover which policy instantiations maximize the probability of satisfying the action schema's transition constraints. This is called the *optimal* policy instantiation. This section defines the optimal policy instantiation and introduces the schema structured learning algorithm given in Table I.

A. Definition of the Optimal Policy Instantiation

Recall that the abstract transition function defines how the robot system must transition after executing actions. An action succeeds when the resulting transition is consistent with these constraints and fails otherwise. A state-action trajectory will be said to succeed when each component action succeeds. An *optimal* policy instantiation, π^*, is one that maximizes the probability of a successful trajectory. Let $P^\pi(a|s_t)$ be the probability of a successful trajectory, given that the system takes action $a \in A$, starting in state $s_t \in S$, and follows policy instantiation π after that. If Π is defined to be the set of all possible policies, then

$$P^*(a|s_t) = \max_{\pi \in \Pi} P^\pi(a|s_t) \quad (10)$$

is the maximum probability of a successful trajectory taken over all possible policies. This allows the optimal policy to be calculated using,

$$\pi^*(s_t) = \arg \max_{a \in B(s_t)} P^*(a|s_t), \quad (11)$$

where $B(s_t) = g^{-1}(\pi'(f(s_t)))$ (Equation 5) is the set of actions that are consistent with the abstract transition function when the system is in state $s_t \in S$. The optimal policy always selects the action that maximizes the probability of satisfying action schema transition constraints.

B. Schema Structured Learning

Schema structured learning autonomously discovers the optimal policy instantiations. The robot explores and models the expected success of different policy instantiations in order to estimate $P^*(a|s_t)$. However, instead of directly using Equation 10, it is possible to use a dynamic programming approach to estimate $P^*(a|s_t)$. For convenience, we assume that the outcome of a successful action is known *a priori*,

$$T_s : S \times A \rightarrow S, \;\; T_s(s_t, a) = s_{t+1}. \quad (12)$$

Given this assumption, the maximum probability of a successful trajectory can be calculated recursively,

$$P^*(a_t|s_t) = P(\text{succ}|s_t, a_t) \max_{a \in B(T_s(s_t, a_t))} P^*(a|T_s(s_t, a_t)), \quad (13)$$

where $P(\text{succ}|s_t, a_t)$ is the probability that action a_t succeeds from state s_t. Assuming that T_s is modeled *a priori* is a reasonable assumption in many robot learning problems because the human designer frequently knows the desired state of the robot after a particular controller executes. However, in cases

TABLE I

SCHEMA STRUCTURED LEARNING ALGORITHM

Function SCHEMA STRUCTURED LEARNING
1. While not in an absorbing state
2. Get current state $s_t \in S$
3. Let $B(s_t) = g^{-1}(s_t)(\pi'(f(s_t)))$
4. Execute $\pi^*(s_t) = \arg\max_{a \in B(s_t)} P^*(a|s_t)$
5. If action failed, break from loop.
6. Get next state $s_{t+1} \in S$
7. Update transition model $P(\text{succ}|s_t, a)$
8. Repeat

where this is not true, a more complex version of Equation 13 can be used that is based on estimates of $P(s_{t+1}|s_t, a)$ rather than $P(\text{succ}|s_t, a)$.

The schema structured learning algorithm is illustrated in Table I. First, the robot assesses its current state and determines which actions instantiate the abstract policy at that point (steps two and three). Next, the robot executes an action that is estimated to be most likely to be part of a successful trajectory (step four). Finally, the transition model is updated and the process repeats. Note that as long as the algorithm's probability estimates are correct, then Equation 13 gives the optimal policy instantiation. Therefore, as long as schema structured learning's probability estimates converge to their true values, the algorithm can be expected to converge to a set of optimal policies.

Schema structured learning based on the action schema framework can be expected to learn faster than learning algorithms that solve Markov Decision Processes (MDPs) for at least two reasons. First, the action schema's abstract policy reduces the search space of learning. Second, schema structured learning only estimates the probability of success - not the probability of arriving in every possible next state ($P(\text{succ}|s_t, a)$ - not $\forall s_{t+1} \in S\ P(s_{t+1}|s_t, a)$. Therefore, the algorithm must estimate fewer probabilities and learning can be expected to be faster.

C. Large Action Spaces

Another key feature of schema structured learning is that a sample-based version of the algorithm exists. This sample-based version is effective in large or real-valued action spaces where it may be difficult to use other function approximation techniques to estimate the optimal action. In the case of large or real-valued action spaces, it is difficult or impossible to evaluate Equation 11 for all possible values of $a \in B(s_t)$. However, notice that actions with a high probability of success are more likely to be part of a successful trajectory. In particular, when the optimal policy instantiation has a high probability of success, the action that maximizes $P^*(a|s_t)$ is also likely to be near the maximum of $P(\text{succ}|s_t, a)$. This enables a sample-based version of schema structured learning to use its estimate of $P(\text{succ}|s_t, a)$ (derived from previous experience) to bias its action sampling in step three (Table I).

The above observation leads to a sample-based version of schema structured learning. Assume that the robot is in state, $s_t \in S$, and that $a' = \pi'(f(s_t))$. Instead of evaluating every instantiation of the abstract policy in step three, the sample-based version of the algorithm only evaluates a fixed number of samples, $D_{s_t}(a') \subseteq g^{-1}(a')$, drawn from the probability distribution $P(\text{succ}|s_t, a)$. This sample set is updated every time the estimate of $P(\text{succ}|s_t, a)$ improves.

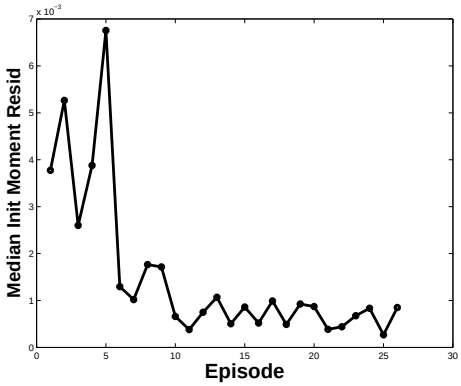


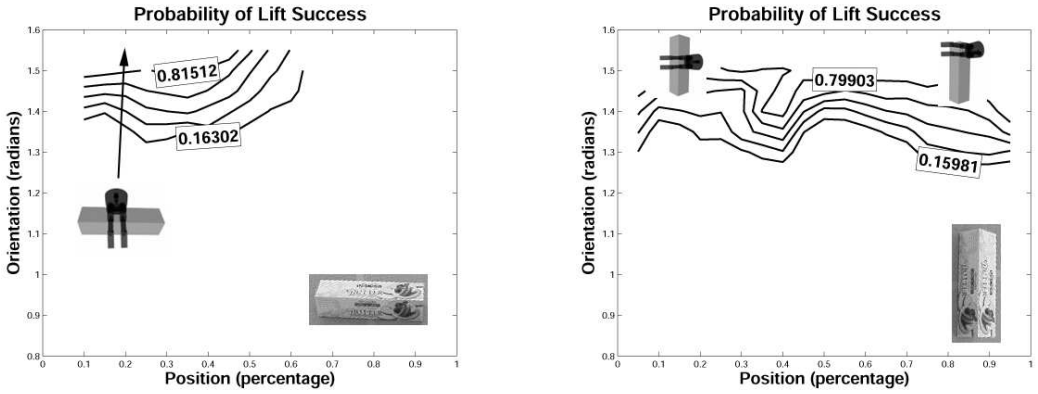
Fig. 3. Results from the first experiment (grasping the vertical cylinder) showing median initial grasp controller error (moment residual error) as a function of trial. A high error indicates a poor grasp and a low error indicates a good grasp. Note that median error reaches its lowest point after approximately 10 trials.

V. EXPERIMENTS

Two experiments were performed to characterize the learning performance of schema structured learning in the context of grasp synthesis. Both experiments were performed using Dexter, a bi-manual humanoid upper torso consisting of two Barrett arms, two Barrett hands, and a Bisight head. In the first experiment, Dexter learned to localize, reach, and grasp a towel roll measuring 20cm high and 10cm in diameter by attempting to grasp it 26 times. At the beginning of each trial, the towel roll was placed vertically in approximately the same tabletop location. Only three-fingered grasps, $\phi_g|_{\tau_{123}}^{\sigma_{123}}$, were allowed. On each trial, schema structured learning executed using the LOCALIZE-REACH-GRASP action schema until either the absorbing state was reached or an action failed. In either case, the trial was terminated, the system was reset, and a new trial was begun. This experiment was repeated eight times.

Figure 3 shows median grasp error as a function of trial number. This is the grasp error measured after reaching to the object, but before executing the GRASP controller. Before the 10th trial, the large median grasp errors indicate that schema structured learning had not yet discovered how to grasp the towel roll. This means that the GRASP controller must correct this poor configuration by displacing the contacts along the object surface toward a good grasp configuration. However, by the 10th or 15th trial, the robot has learned to select an instantiation of the REACH controller that minimizes moment residual errors.

In the second experiment, Dexter learned to adjust its grasp strategy based on the orientation of the object. A long box measuring 7x7x27cm was alternately presented to Dexter



(a) (b)

Fig. 4. Results from the second experiment showing the two grasp strategies learned for the different box orientations. The contour plot in (a) shows that the probability of success is maximized when the manipulator reaches to the center of the box and orients itself perpendicular to the box major axis. (b) shows that position matters less when the box is presented vertically.

horizontally and vertically. Dexter attempted to grasp and lift this object 40 times using a LOCALIZE-REACH-GRASP-HOLD-LIFT action schema. In addition to attempting to reach and grasp the object, this action schema also applied a grasping force and attempted to lift the object. The lift was only considered to succeed when the object did not exert a large moment about the contact points and it did not swing out of the grasp or drop. On each trial, the action schema executed until either the absorbing state was reached or an action failed. Only two-fingered grasps were allowed.

The contour plots in Figure 4 illustrate the results of the second experiment. Both figures show the probability of grasp success as a function of orientation (vertical axis) and position (horizontal axis) relative to the object. Position and orientation are measured in the same way as with the REACH controllers in Section II: position is the distance of the contact centroid between the center and one end of the major axis; orientation is the average angle formed by the major axis and the line between a contact and the contact centroid. Figure 4(a) shows the probability of lift success for the horizontally presented box, and Figure 4(b) shows the probability of success for the vertically presented box. Both graphs show that the robot learns to orient its hand perpendicular to the major axis of the box (in both graphs, the probability density is maximized near the top.) However, notice that when the box is presented horizontally, the robot also learns to grasp it near its center. This behavior improves the robot's chances of lift success because it is less likely to drop the object in this configuration.

VI. Conclusion

This paper expands on the action schema approach to robot learning proposed in [3]. In this approach, the search for desired robot behavior is constrained by a generalized policy encoded by the action schema. This simplifies learning: instead of considering all possible behaviors, the robot must only consider instantiations of the generalized action schema policy. This paper defines the optimal policy instantiation as that which maximizes the probability of satisfying action schema transition specifications. In an approach reminiscent of Piaget's process of assimilation, this paper proposes a sample-based algorithm, schema structured learning, whereby the robot repeatedly executes instantiations of the action schema policy in a search for optimal instantiations. The results show that the system quickly learns how best to reach and grasp an object and that the robot is able to adjust its strategy based on how an object is presented.

Acknowledgment

This work was completed while the first author was a student at the University of Massachusetts. This work was supported by NASA grant NNJ05HB61A, ARO grant DAAD 19-03-R-0017, and NASA GSRP Fellowship NNJ04jf76H.

References

[1] M. Huber and R. Grupen, "Learning to coordinate controllers - reinforcement learning on a control basis," in *Fifteenth Int'l Joint Conf. on Artificial Intelligence*, 1997.
[2] R. Arkin, *Behavior-Based Robotics*. MIT Press, 1998.
[3] R. Platt, A. H. Fagg, and R. A. Grupen, "Re-using schematic grasping policies," in *IEEE Int'l Conf. on Humanoid Robots*, 2005, pp. 141–147.
[4] E. Martinson, A. Stoytchev, and R. Arkin, "Robot behavioral selection using qlearning," in *IROS*, 2002.
[5] A. Rosenstein and A. Barto, "Robot weightlifting by direct policy search," in *Seventeenth Int'l Joint Conf. on Artificial Intelligence*, 2001.
[6] J. Piaget, *The Origins of Intelligence in Children*. Norton, NY, 1952.
[7] M. Arbib, "Schema theory," in *Encyclopedia of Artificial Intelligence (2nd Edition)*. Wiley-Interscience, 1992.
[8] G. Drescher, *Made-Up Minds: A Constructivist Approach to Artificial Intelligence*. MIT Press, 1991.
[9] J. Coelho, "Multifingered grasping: Grasp reflexes and control context," Ph.D. dissertation, University of Massachusetts, 2001.
[10] J. Coelho and R. Grupen, "A control basis for learning multifingered grasps," *Journal of Robotic Systems*, 1997.
[11] R. Platt, A. H. Fagg, and R. A. Grupen, "Nullspace composition of control laws for grasping," in *IEEE Int'l Conf. on Intelligent Robots and Systems*, 2002.
[12] ——, "Manipulation gaits: Sequences of grasp control tasks," in *IEEE Int'l Conf. Robotics Automation*, 2004.
[13] R. Platt, A. Fagg, and R. Grupen, "Extending fingertip grasping to whole body grasping," in *IEEE Int'l Conf. on Robotics and Automation*, 2003.